Received: 30 September 2022 | Revised: 6 April 2023 | Accepted: 30 May 2023

DOI: 10.1002/nme.7308

RESEARCH ARTICLE

WILEY

An optimization based multi-block-structured grid generation method

Marcel Sauer | **Christian Morsbach**

Institute of Propulsion Technology, Numerical Methods, German Aerospace Center (DLR), Linder Höhe, Cologne, Germany

Correspondence
Christian Morsbach, Institute of Propulsion Technology, Numerical Methods, German Aerospace Center (DLR), Linder Höhe, Cologne, 51147, Germany.
Email: Christian.Morsbach@dlr.de

Summary

A new optimization based meshing technique for block-structured meshes is presented. Given a topological block partition, block sizes, and distance constraints, meshes are generated automatically. The optimization effort is limited by using a BSpline based intermediate layer with reduced degrees of freedom, making it suitable for high resolution meshes. The optimization itself minimizes approximated grid criteria defined in the intermediate layer domain. Sampling the intermediate layer results in a mesh, which is projected in order to preserve geometric constraints. The capability of the new method is shown for 2D and 3D geometries.

KEYWORDS

block-structured, BSpline, grid generation, meshing, optimization

1 | INTRODUCTION

The majority of numerical simulation methods use meshes for the spatial discretization of the solution domain. Depending on the simulation method, high requirements on the mesh quality may be in place. For example, in the case of CFD simulations, wall normal oriented cells with an appropriate first cell size, near orthogonal cell angles, and a moderate expansion ratio into the interior of the domain are preferred.

A manual meshing process is highly iterative and time consuming. Automatic approaches, where parameters are adjusted for a small set of geometries and reapplied for a wide set of parametrized geometries, are necessary for automatic design optimizations. Especially for unstructured grids many automated methods exist for creating triangles, quads, tetrahedra, and hexahedra meshes.[1-4] In combination with marching front algorithms for resolving boundary layers, appropriate meshes can be produced automatically for a wide set of geometries.[5,6] In contrast to unstructured grids, fully automated methods for creating high-quality block-structured meshes for arbitrary geometries are not present. On the other hand, block-structured meshes are the preferred choice if it is possible to create them with a reasonable effort, as they have advantages in terms of memory usage, computational efficiency, numerical formulations, and required cell counts.[7-9] The dominating approach for automatic structured mesh generation is the elliptical grid generation.[10-12] To prescribe mesh constraints, moving points on geometries, and multi-block meshes, modifications are necessary.[13-15] The elliptical grid generation does extend to 3D, but is often only applied in 2D and generates 3D meshes by stitching several slices together.[16]

The proposed method is designed for classes of geometry, for which a suitable topological block partition is known. Several 2D applications for turbomachinery passages were published with an early version of the method.[17]

The proposed method is a new combination of optimization techniques and the usage of modified BSpline objects to represent block structured meshes. A coarse block-structured mesh must be provided, which is used to define geometry associations and distance settings. With additional vectors of scalar sampling positions, a BSpline based intermediate layer with reduced degrees of freedom is constructed, which can be optimized and sampled to generate a mesh. This intermediate layer representation offers good scaling with the final grid resolution. With the usage of the intermediate layer, the degrees of freedom of a mesh can be drastically reduced compared to a naive approach, where every mesh point contributes two or three degrees of freedom. In real world 2D cases, the degrees of freedom can be reduced by about two orders of magnitude compared to the naive approach. About three orders of magnitude can be expected for real world 3D cases. The optimization uses different criteria types defined on the intermediate layer, which approximate the criteria in the sampled mesh. The sampled mesh must be fitted to the geometry, as the sampling can achieve high-quality meshes in the interior, but not boundary conforming meshes. With a reasonable propagation algorithm or small deviations to start with, the mesh quality can be preserved within the adaption.

The method settings allow the user to alter the behavior of the meshing. As such, boundary orthogonal meshes with a specified first distance can be created without specifying the final node positions. Targets for expansion ratios can be defined and local weight adjustments are possible. With adjusted general weights for points, a trade off can be made, which prioritizes local grid criteria over the overall grid criteria. Similarly, the criteria specific weights allow for a local adjustment of the balance between criteria types. Compared to elliptical grid generation methods, this flexibility allows the user to tweak the mesh on a high level. As the settings are bound to the block topology, the geometry can be replaced and a mesh with similar features can be produced.

The last intermediate representation can also be used to resample a mesh with changed distance prescriptions and sampling resolutions while preserving the original topology and spatial block partition. This feature assists the user in creating comparable meshes within grid convergence studies.

In this article, the intermediate layer will be presented. With this layer, a mesh can be generated based on a limited parameter set. Furthermore, optimization steps and automatic refinements of the intermediate layer are described. Implementation details and simplifications regarding the optimization fitness function are presented. The method is applied to an academic 2D example with multiple intermediate steps shown. This example is extended to a 3D geometry and also a 3D mesh is generated. Targeting real-world applications, a 2D example grid for a turbine cascade passage is presented and notes for the method's extension for 3D applications are provided.

2 | INTERMEDIATE LAYER

An intermediate layer is used to limit the degrees of freedom and as such the complexity of the optimization steps. The intermediate layer uses BSpline objects, for which modifications are discussed later, to represent continuous surfaces or volumes with the desired block topology. Each BSpline object, a surface or a volume, has a control grid with control points to describe the continuous object. The mesh is generated by sampling the intermediate layer. The degrees of freedom of the intermediate layer are defined by the sampling positions and the parametrization of the BSpline control points. These degrees of freedom will be the parameters of the upcoming optimization and the intermediate layer is used by the optimization fitness function to evaluate the mesh quality. Each control point of a BSpline object can be assigned to follow curves or surfaces, be completely fixed in space, or follow some other constraints. Within a BSpline object, the sampled nodes are generated by the Cartesian product of vectors for the sampling positions along the parametric axis. The resolution of the control grids is expected to be constant and independent of the resulting mesh resolution, while being coarser. By using vectors along parametric axes for sampling positions, the quadratic or cubic scaling of degrees of freedom of the mesh representation with the final mesh resolution is reduced to less than linear. With this setup, it is estimated to reduce the degrees of freedom for the grid for a real-world 3D CFD simulation by two to three orders of magnitude compared to the degrees of freedom of the final mesh, which is estimated by the number of grid points multiplied by the spatial dimensions. Including control grids, sampling positions, connectivity, constraints, and local settings, the intermediate layer encapsulates all data for a single step of the method: setting up an optimization problem, deducing new intermediate layers with refined control grids, sampling a full resolution mesh, and providing projection targets for enforcing geometric constraints. The following sections will describe the construction of the intermediate layer, introduce modifications for multi-block applications, and list requirements for sampling valid grids.

2.1 | Construction details

Figure 1 shows an initial control grid, which is used by the intermediate layer, and the sampled output mesh. The initial control grid uses the desired block topology. For control grid of the intermediate layer, all blocks are split into multiple sub blocks in a way that every block border is associated with no or exactly one block border or geometry entity. The intermediate layer's control grid shares the block topology with the sampled mesh. Likewise, the final resulting mesh will be merged to restore the initial control grid's block topology.

In block-structured meshes, nodes or edges prohibiting a structured stencil are called singularities. In Figure 1, one singularity exists at the upper right corner of block A. Splitting block B at this singularity and generating the blocks B1 and B2 avoids the $j = 1$ border of B being shared with different borders of block A. Each block in the intermediate layer's block-structured control grid is used as the control grid for a second degree BSpline[18] surface or volume with equally spaced knot vectors. The equally spaced knot vector excludes the possibility of a local degree reduction and ensures an equal influence of the control points. Setting the second degree limits the number of influencing control points to three to the power of the parametric dimensions while preserving the ability to avoid kinks within a BSpline object. For each point to be sampled, a structured 3×3 ($\times$ 3) stencil of control points is used. With this degree, the BSpline follows the control points closely and the control points can be used to calculate approximated grid criteria for the fitness function for the later intermediate layer optimization.

The sampled grid of a BSpline object is generated by the Cartesian product of vectors which define the sampling positions for each parametric axis (see $\vec{a}$, $\vec{b}$, and $\vec{c}$ in Figure 1). Across block interfaces, matching grid points are generated by sharing the control points along the block border and using the same sampling positions. This is ensured by sharing the sampling positions along a parametric axis with blocks perpendicular to the parametric axis. This can be seen in Figure 1 where A is sampled by $(\vec{c} \times \vec{a})$ and B1, using the same vector $\vec{c}$, is sampled by $(\vec{c} \times \vec{b})$. For block B2 the vector $\vec{a}$ has to be reversed since the vector's direction of definition and the parametric direction of i do not align. The sampling vectors are expected to consist of monotonically increasing values where the first and the last values are equal to the definition range of the associated BSpline parameter. The parametrization consist of the parameters for each unique point in the control grid and the inner values of the sampling vectors. The first value of the sampling vector is set to the minimum value of the definition range of the BSpline object in the according direction. The same applies for the last values of the vector and the definition range. These values ensure that the whole BSpline objects are sampled and define the block borders.

With standard BSpline objects the continuous intermediate layer would produce an unsteadiness in the first derivative across block boundaries. In regular regions, the control grid of neighboring blocks can be merged to generate a steady derivative (see shared sampling in Figure 1). Wherever it is not possible to generate the structured stencil required to sample a point, a special sampling has to be applied (denoted as singularity sampling in Figure 1).

Further modifications are required in order to avoid artifacts in the mapping of an arc length dependent on the number of control points and limit the influence width of control points near a singularity. The sampling near singularities is then done by a non-linear combination of modified BSpline objects. Within the sampled mesh, the geometry surfaces will be approximated with the BSpline representation of the intermediate layer. In order to create geometry conforming meshes, a projection followed by a displacement is needed.

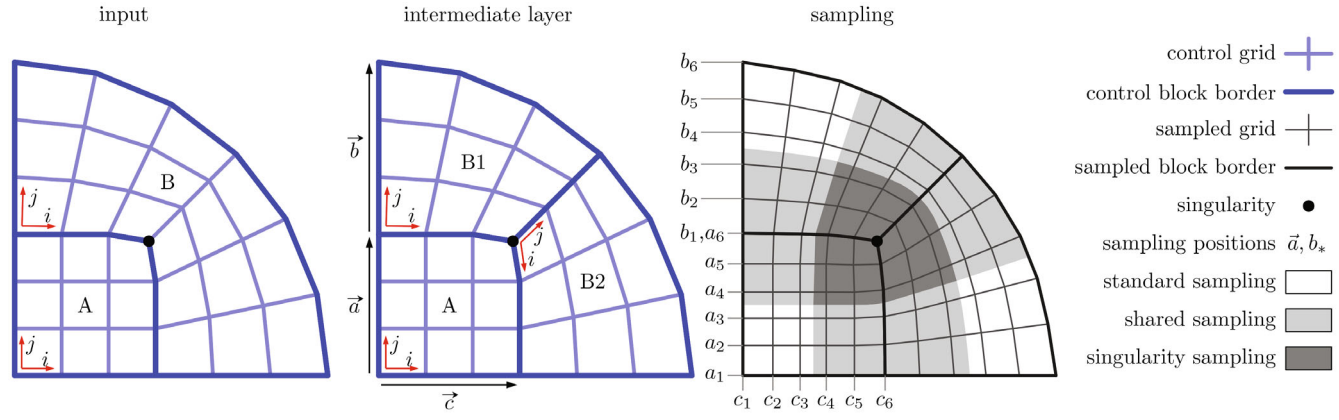

FIGURE 1 Intermediate layer and sampling.

2.2 | Length mapping

In a multi-block mesh, a similar behavior of the BSplines with different numbers of control points is necessary. For example, in Figure 1 the upper border of block A has four control nodes, as no structured stencil can be applied across the singularity. On the other hand, at the lower border a structured stencil is possible and a merged control grid with more points is used. At the same i-wise sampling position, the lower and upper borders should behave similarly. Reduced to a one-dimensional case, a BSpline curve with an equally spaced control polygon, the arc length up to the curve parameter u should be linear to u. Unfortunately, the standard formulation of a second degree BSpline with an equally spaced knot vector only produces a linear distance mapping with exactly three control points. With more control points the distance mapping has an acceleration part, followed by a linear section and ending with a deceleration. The Figure 2 visualizes this behavior. The left side shows the control polygon for different numbers of control points and sampled locations along the curve, which are equidistant in u. The sampling uses three points per line segment of the control polygon and has the same average distance for all cases. The term segment will always refer to a mesh edge, which connects two adjacent mesh nodes. This setup will only create equidistantly spaced sample positions for $n = 3$ and in the center region of $n \geq 5$. This phenomenon is highlighted by the derivative on the right side of Figure 2. For $n \geq 5$ one or more repetitive segments of basis function are present, which result in a partially constant derivative.

In order to avoid the non-linear parts, the BSpline parameter is not used directly when more than three control points are present. Instead, an auxiliary function is used to transform a sampling position to a BSpline parameter. This function nullifies the acceleration and deceleration parts while stretching the definition range by half of the knot distance Δk. It uses the number of control points n and the knot vector entries $k_1 \ \ldots \ k_{n+3}$ and is defined by the following formula:

$$u_{\mathrm{BS}}(u) = \begin{cases} k_5 - \sqrt{-2 \cdot \Delta k \cdot u + k_5^2 - k_4^2} & \text{for } u < k_4 \\ u & \text{for } k_4 \leq u \leq k_{n-3} \\ k_{n-4} + \sqrt{2 \cdot \Delta k \cdot u + k_{n-4}^2 - k_{n-3}^2} & \text{for } u > k_{n-3} \end{cases} \tag{1}$$

Figure 3 shows on the left the BSpline basis functions for an unmodified BSpline and a BSpline with the length correction applied. On the right side of Figure 3, comparisons are shown between sampling short curves with three control nodes, where no correction is necessary, and long curves without the correction applied (above the short curves) and with the correction applied (below the short curves). With the correction, the sampling positions of long and short curves match

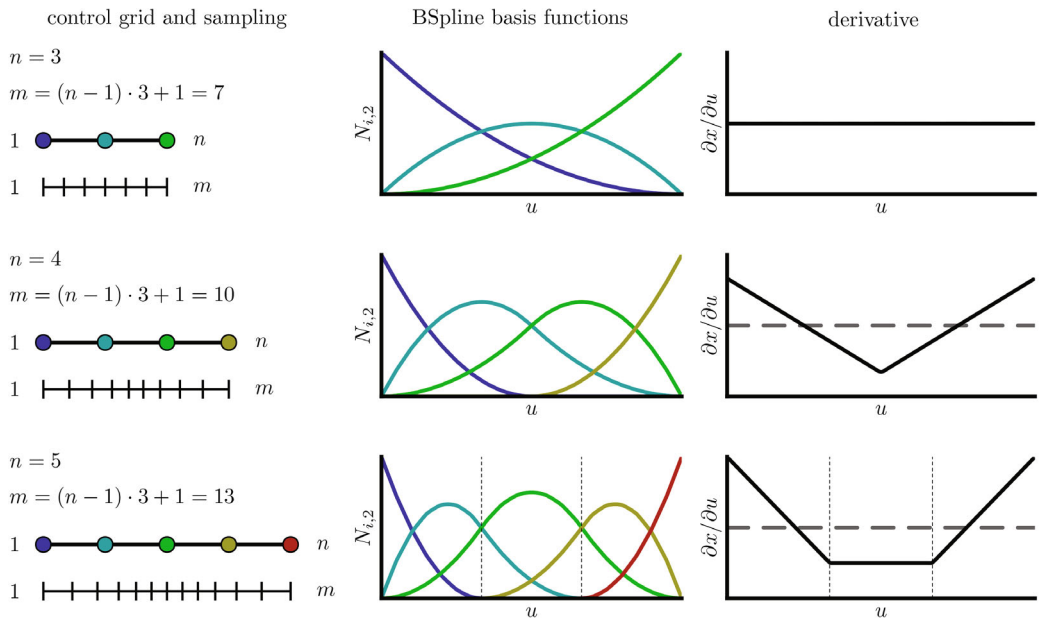

FIGURE 2 Standard second degree BSpline length mapping.

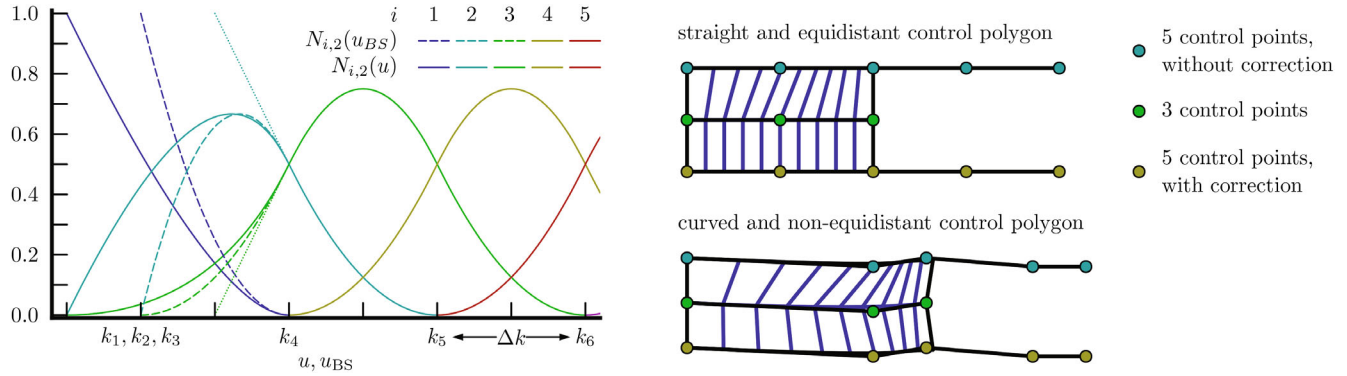

FIGURE 3 BSpline length correction; left: basis function with (solid) and without correction (dashed), right: sampling comparison.

exactly for straight equidistant control polygons and different numbers of control points. Also, the matching is better when using curved and non-equidistant control points, as can be seen by the skewness of the cells shown between the curves.

2.3 | Degree reduction

Contrary to the homogeneous behavior for general regions, a further limitation of influencing control points is beneficial for modeling singularities. For a curve starting at a singularity, a linear section allows to sample a distance within this region based on two control nodes. With this limitation, less restrictions are required for a homogeneous sampling in the proximity of a singularity. This can be achieved by replacing the starting area of the BSpline basis functions with linear segments attached to the first of the repetitive segments. Using the basis function plot of Figure 3, curves for $u < k_4$ are removed and the curves $N_{2,2}$ and $N_{3,2}$ are extended linearly and slope conserving (dotted line in figure) to $u = (k_3 + k_4)/2$ ending one and zero, respectively. As the original curve $N_{1,2}$ is not used anymore, one less control point is needed.

2.4 | Multi-block sampling

With multi-block topologies the structured stencil, which is required by the standard BSpline like sampling, cannot be created in the proximity of singularities and topology-wise inwards facing block corners and edges. With a singularity as the point or edge, which prevents a structured stencil, inwards facing corners and edges are also denoted as singularities.

The non-repetitive behavior at the start of the BSpline basis functions ends at $u = k_4$, where the second and third control point have the weight 0.5. Beyond this point the decision whether to share an additional control node does not influence the curve anymore and a standard stencil can be used to select relevant control points until the non-repetitive part at the end begins. Extended to 2D, regions in which a standard sampling, a standard sampling with shared nodes of neighboring blocks, and special singularity sampling has to be applied, can be separated. These sampling regions are highlighted in Figures 1 and 4. With topology-inwards facing block corners, which are treated in a similar way as singularities, and inner singularities in 2D two cases have to be handled, illustrated in Figure 4. The distinction between those is based on the existence of neighboring blocks denoted by the arrows in the figure. The first configuration for block A has a direct neighbor to the left and diagonally to the bottom left, but no block is present below A. The same configuration applies for block C when switching and reversing BSpline parameter. The second configuration for block D has direct neighbors below and to the left, but has no unique block to the bottom left. This may happen because the lower left point forms an inner singularity, and as such the lower edge of E is connected to the left edge of F or between E and F, there are ≥ 2 blocks present. Another option for this configuration is an inward facing block corner, where the lower edge of E and the left edge of F are borders of the topology. Therefore, the sampling method of block D will also apply to block B. The blocks E and F may be sampled by the procedure for block D in case of an inner singularity and by the procedure for block A otherwise.

For sampling these regions quadratic blending functions of multiple valid surface definitions are used. The BSpline parameter of all used surfaces are transformed to be $(0, 0)$ at the current blocks (A or D) lower left corner and 1 along each

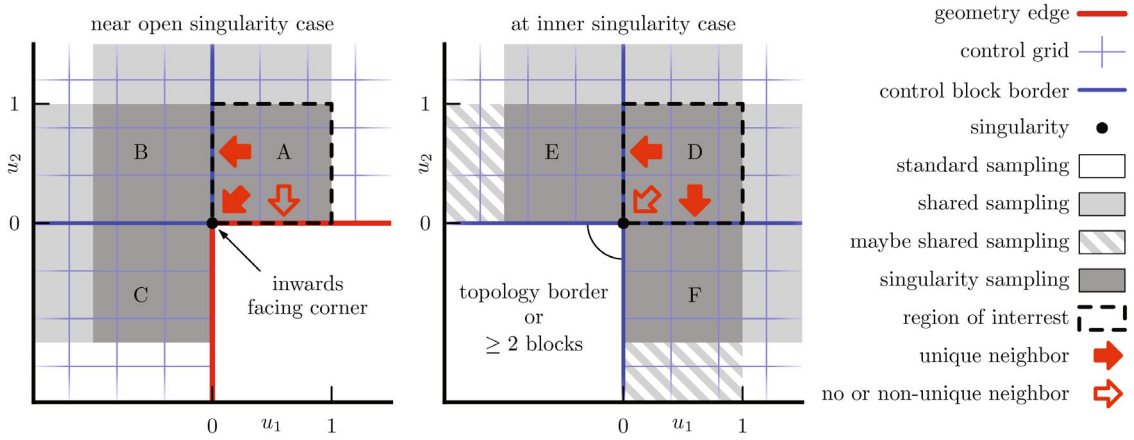

FIGURE 4 2D neighbor configurations in BSpline parameter space.

axis, where repetitive behavior of the basis function starts ($u = k_4$ without shared nodes). In order to limit the influence of the singularity or inwards facing corner, these surfaces are defined with a degree reduction towards the point $(0, 0)$.

In the case of block A, two surfaces can be defined. The surface S_{A} only uses control points of block A and has a degree loss at the left and the lower edge. The surface S_{BA} also uses the control points of block B and has a degree loss at the lower edge. Within the region $u_1 \in [0, 1), u_2 \in [0, 1)$ A is sampled as S_{A^*} by Equation (2) and, as such, the sampling along the block edge between A and B is defined.

$$\begin{aligned} f_{\mathrm{A}}(u_1, u_2) &= (1 - u_2)^2 \\ f_{\mathrm{BA}}(u_1, u_2) &= 1 - (1 - u_2)^2 \\ S_{A^*} &= f_{\mathrm{A}} \cdot S_{\mathrm{A}} + f_{\mathrm{BA}} \cdot S_{\mathrm{BA}} \end{aligned} \tag{2}$$

Regarding block D, as E and F may be sampled using the sampling method for block A, the sampling along the block edges must be preserved. This is achieved by reusing the sampling of block A with blocks (E and D) and (F and D). Another two-sided quadratic blending f_d along the diagonal of D leads to the sampling for block D as $S_{D^{**}}$ shown in Equation (3).

$$\begin{aligned} d(u_1, u_2) &= \begin{cases} 0 & \text{for } (u_1, u_2) \in (0, 0), (1, 1) \\ (-u_1 + u_2)/(u1 + u2) & \text{for } u_1 + u_2 < 1 \\ (-u_1 + u_2)/(2 - u1 - u2) & \text{for } u_1 + u_2 \geq 1 \end{cases} \\ f_d(u_1, u_2) &= \begin{cases} \frac{1}{2} \cdot \left(2 - (1 - d(u_1, u_2))^2\right) & \text{for } u_2 > u_1 \\ 1 - \frac{1}{2} \cdot \left(2 - (1 - d(u_1, u_2))^2\right) & \text{for } u_2 \leq u_1 \end{cases} \\ f_{\mathrm{D}}(u_1, u_2) &= (1 - u_1)^2 \cdot (1 - f_d) + (1 - u_2)^2 \cdot f_d \\ f_{\mathrm{ED}}(u_1, u_2) &= \left(1 - (1 - u_2)^2\right) \cdot f_d \\ f_{\mathrm{FD}}(u_1, u_2) &= \left(1 - (1 - u_1)^2\right) \cdot (1 - f_d) \\ S_{D^{**}} &= f_{\mathrm{D}} \cdot S_{\mathrm{D}} + f_{\mathrm{ED}} \cdot S_{\mathrm{ED}} + f_{\mathrm{FD}} \cdot S_{\mathrm{FD}} \end{aligned} \tag{3}$$

Sampling an inner singularity in 3D leads to two cases. If the edge along a singularity does not intersect with the edge of another singularity, the sampling is done similar to the surface singularity sampling. The parameters u_1 and u_2 are chosen to be perpendicular to the singularity edge to calculate the blending factors and the surfaces are generated by fixing the third parameter along the singularity edge in the BSpline volumes. If the singularity edges of two or more singularities intersect in a point, it is called 3D singularity. For these four valid volume definitions can be found and blended together similarly to 2D singularities (not presented here). At the boundaries of the influence range of the 3D singularity, these definitions must be consistent with the 2D singularity sampling.

2.5 | Displacement and refinement

BSplines are capable of describing a continuous surface/volume using a small set of control points. Within a mesh generation task, edges/surfaces of the mesh are bound to geometric entities. Matching an arbitrary geometric entity with a BSpline object is generally not possible with a limited set of control points.

Therefore, only the control points are directly assigned to geometric entities and a displacement step is performed to project the sampled mesh onto the geometric boundaries. This displacement must fade into the interior of the mesh to avoid mesh folding and preserve grid quality. With a coarse control grid a good spatial block partition can be found, but the required displacement deltas may be big. In order to keep the degrees of freedom limited, a refinement of the control grid is used to increase the resolution while keeping the location of all previous control points fixed. With an increased resolution the displacement deltas decrease and the displacement task can be locally constrained.

2.6 | Intermediate layer requirements

The most basic requirement is the generation of a non-folding grid. Within regular regions, a non-folding control grid and strictly monotonically increasing BSpline sampling positions are sufficient. In regions sampled by one of the singularity methods, this property does not exist. Nevertheless, in converged optimization results distances and angles are spatial homogeneous leading to valid meshes.

Requirements of the final mesh such as orthogonality to edges/surfaces and first distances can be translated to requirements on the intermediate layer. Orthogonality can be achieved by placing the first control point in the interior orthogonal to the edge/surface at the location of the control point on the boundary. Setting distances can be done approximately by defining the distance as an optimization parameter or fixing a distance in the control grid and fixing the sampling position. For setting an exact distance, three control points in the direction of the distance have to be fixed and the involved points must be sampled where only these fixed control points have an influence. As the displacement modifies the mesh and fades into the interior, distance constraints which have previously been set exactly will appear slightly distorted.

For passing a block border in a regular region without a jump in sampled cell sizes, the distances of the sampling positions across the border, normalized with the equidistant knot vector distance, must be similar. As a singularity decouples the involved blocks, the first cell size of the control grid also has to be of similar size at singularities.

2.7 | Control point parametrization

As shown above, a multi-block control grid can be used to sample a grid. Beside non-fixed sampling positions as parameters, each control point in the grid may contribute some degrees of freedom. Control points that appear as part of multiple blocks share the same parameters. Based on the topology, control points may be fixed, on a curve, on a surface, or not constrained at all. Further user settings can define additional constraints. In order to provide an orthogonal distance control, a base point is constrained to a curve or a surface and a second control point uses a fixed distance orthogonal shift to the base control point using its derivatives. Such a control point can be used to enforce orthogonality and first distance constraints. In order to preserve periodic conditions, only one side of a periodic boundary uses a parametrized control point. The other side is defined to be generated by a fixed transformation from the first boundary.

The constraints are defined hierarchically by narrowing down the degrees of freedom. Keeping track of the more general constraints is used to deduce projection targets and constraints of newly inserted control points during the refinement process. As an example, a control point on the intersection of two surfaces is free at the beginning. Then it is constraint to be on the first surface, to be on the second surface, and then to follow the intersection curve of the surfaces, consecutively. Lastly it may be set to be completely stationary.

3 | GRID GENERATION METHOD OVERVIEW

The generation of the final grid is divided in several steps. The steps consist of initialization, coarse optimization, refinement, sampling position optimization, and sampling and displacement. Each optimization step uses the same procedure to construct the fitness function, but differs in degrees of freedom. Figure 5 shows flow chart of the method with sketches

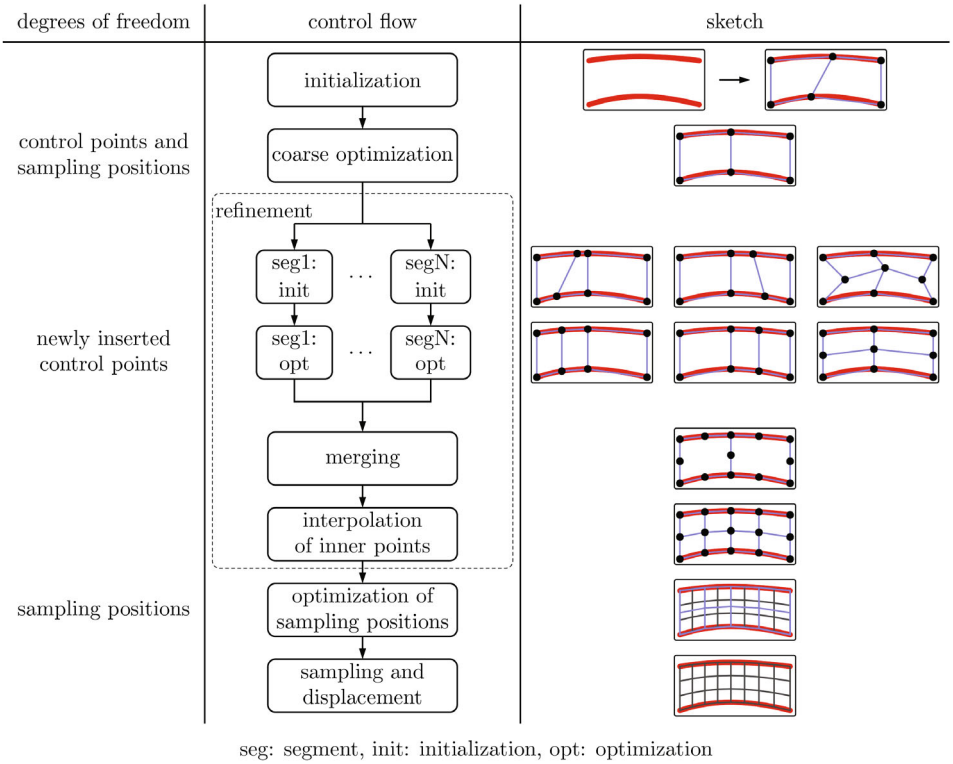

FIGURE 5 Grid generation steps.

for the intermediate steps. The following sections will describe the purpose of these steps, whereas an application of these steps is presented in Section 5.

3.1 | Initialization

In the first step an initial intermediate layer representation is created. The control points of this layer are also used to set constraints and local influence parameters for the mesh. Initializing the intermediate layer's control grid and sampling positions is a geometry dependent task while setting constraints and local influence parameters are options based on the topology. Typically, an application class encapsulates the control grid generation and sets constraints, refinements, and parameters for a desired topology. Examples for application classes are turbine and compressor passages, ducts, and blade FEM meshes.

The initial control mesh must be a non-folding, block-structured grid with the desired topological block partition. Due to the sampling at BSpline parameter values, the resolution of the control block is independent of the final resolution. Additionally, the initial mesh should be conforming with the restrictions applied to it later on. Points on curves/surfaces should be initialized on the curves/surfaces, orthogonal shifted points should match the curve/surface normal at the base point and periodic cases should have matching periodic boundaries. As no further constraints are set, such an initialization can often be achieved by basic algebraic grid generation techniques such as orthogonal offsets, equidistantly sampled lines, and transfinite interpolations.

The initial set of BSpline sampling positions is generated by setting the limits of each edge in the split control grid to match the start and the end of a control block. If a first or a last distance is set by other than by an additional optimization criterion, the corresponding second or second last BSpline sampling position is set appropriately. The remaining inner values are part of the parametrization of the intermediate layer and initialized equally spaced in parameter space.

Dependent on the constraint of a control point, it contributes zero to three parameters to the intermediate layer's parametrization vector. In the optimization, grid criteria will be defined, which utilize the control grid to be calculated. Besides setting constraints, these criteria can be influenced by setting general or criteria specific weights and targets for control points.

3.2 | Coarse optimization

The coarse optimization step is the first of three similar optimization steps. All optimization steps aim to minimize a set of criteria, which are defined to be zero at their optimum. The fitness function for the optimizations is defined by:

$$\epsilon = \sum_{\text{ct}} \sum_{i} \left(w_{\text{ct}} \cdot w_{\text{p}} \cdot w_{\text{m}} \cdot t_{\text{ct}}(c_{\text{ct},i}(\vec{p})) \right)^2 \tag{4}$$

Where ct is a criteria type, i a criterion instance and $c_{\text{ct},i}$ is the evaluation of a criteria instance. Each criteria type is calculated utilizing the intermediate layer and approximates the criteria type in the sampled mesh. Examples for criteria types are inner cell angles, which can be estimated directly by the cell angles of a control grid cell, and expansion ratios, which can be sampled along each grid line of the control grid. Some criteria require a non-folding control grid to be calculated correctly. In case of a folding control grid, ϵ is not evaluated.

The weight w_{ct} is specific for a criteria type and used to define the balance between different types. w_{p} is the local weight manipulation, which may be set with the initial control grid. The block-local weight w_{m} is used to account for differences in control grid resolution and final resolution by the number of occurrences for a given criteria type. For example, a cell-based criterion is evaluated for each cell in a block of the control grid. This block may produce a bigger number of cells in the sampling process resulting in a ratio of sampled cells to evaluated cells. These ratios can differ from block to block and, for some criteria, within a block for different combinations of index directions. The factor w_{m} normalizes the criterion's contribution for all ratios by the quantity of possible applications in the sampled result:

$$w_{\text{m}} = \sqrt{\frac{\text{occurrences sampling}}{\text{occurrences control grid}}} \tag{5}$$

The function t_{ct} is a criterion type specific transfer function for an optional nonlinear scaling.

Within the coarse optimization, control node parametrization and sampling positions are used as degrees of freedom. It is used to set the spatial block distribution and the general shape of the mesh. The resolution of the control grid must be sufficient to represent the desired mesh. Sampling the optimized intermediate layer after a converged optimization should result in high-quality mesh which roughly follows the shape of the geometry input. Without the definition of a refinement, the following steps are skipped and the mesh is sampled and projected to be conforming with the geometry. With a refinement definition, the following optimizations decrease the distance between the sampled mesh and the geometry in order to simplify the projection and displacement step while preserving the mesh quality.

3.3 | Refinement

Within the refinement, the resolution of the control grid is increased and, as a result, the sampled mesh will be closer to the geometry. For each edge of the initial control grid a refinement factor can be set. The refinement factors default values should be set by the application class with the ability for the user to adjust the values. When a refinement factor is set, all segments along the selected edge excluding segments near singularities and segments for distance prescriptions will be divided into multiple segments. In the block-structured manner, refining an edge segment also refines all edge segments perpendicular to the edge direction.

For each group of edge segments an initialization and an optimization will be performed. The initialization is done by placing points equidistantly along edge segments. Based on the constraints, the inserted points may be projected and their displacement be faded into the interior. In the refinement optimization all control points of the coarse control grid and sampling positions are treated as constant and only the newly inserted points provide degrees of freedom. Between groups of edge segments, no dependencies are present and these initializations and optimizations can run in parallel.

After inserting and optimizing points along different edges, a fine control grid is assembled. It includes all coarse points and all inserted points. Hereby uninitialized points occur when a single cell of the coarse control grid is refined in two or three directions (see sketches for merging and interpolation within the refinement box in Figure 5). These points are set using transfinite interpolations and are eventually projected onto the geometry.

3.4 | Sampling position optimization

During the refinement optimizations the BSpline sampling positions are fixed to ensure independent optimizations. With a changed control grid, there may be some potential left in adjusting these positions. Therefore, an additional optimization using the fine control grid is done. The degrees of freedom consist only of the non-fixed BSpline sampling positions. If no refinement was requested, this optimization step is skipped.

3.5 | Sampling and displacement

The optimized and refined intermediate layer is sampled using the BSpline sampling positions considering the modifications discussed in Section 2. The resulting grid is expected to have a high quality after the optimization steps. But, as described in Section 2.5, a projection step is needed to ensure geometric constraints on the sampled mesh. In order to preserve the grid quality all resulting shifts must be propagated into the interior of the grid.

The geometry fitted mesh is finally merged in order to reverse the splitting introduced in Section 2.1 and the grid is returned with the topology that was given by the initial control grid.

4 | IMPLEMENTATION DETAILS

In the previous sections, the concept of an optimization-based grid generation method was presented. This section lists choices that were made to generate the meshes in the following sections.

4.1 | Criteria types

In order to define a fitness function, a set of criteria types with a set of rules where they should be applied, is needed. Further criteria types, for example, curvature, scaled Jacobian and others, can be used but are not considered here as they are not required for the presented test cases and the concept of the method. All criteria instances are dependent on the position of some control nodes of the BSpline control grid and may be dependent on some BSpline sampling positions. Especially in a refinement optimization, all influencing control nodes and sampling positions may be fixed and the degrees of freedom of the optimization cannot affect a criterion instance. Such instances are removed from the fitness function.

4.1.1 | Inner cell angle

The inner cell angle criterion measures the deviation from a target angle at each node of a cell ($\vec{P}_2$) for all combination of two adjacent edge vectors ($\vec{P}_1 - \vec{P}_2$ and $\vec{P}_3 - \vec{P}_2$). An example for the influencing control points is shown on the left side of Figure 6.

$$c_w = \left| \arccos\left(\frac{\left(\vec{P}_1 - \vec{P}_2\right) \cdot \left(\vec{P}_3 - \vec{P}_2\right)}{\left|\vec{P}_1 - \vec{P}_2\right| \cdot \left|\vec{P}_3 - \vec{P}_2\right|} \right) - \alpha_0 \right| \tag{6}$$

The target angle α_0 is $\pi/2$ in regular regions and with $\vec{P}_2$ being part of a cell touching a singularity adjusted by the number of edges leading to the singularity. In 2D cases, four criteria instances are constructed per cell in the control grid. For 3D cases there are 24 inner cell angle criteria instances per control grid cell.

4.1.2 | Expansion ratio—control grid

The expansion ratio of the control grid is an auxiliary criterion targeting an equal distribution of control points. It uses three consecutive control points $\vec{P}_1, \vec{P}_2, \vec{P}_3$ following an edge of the control grid, as illustrated on the right side of Figure 6.

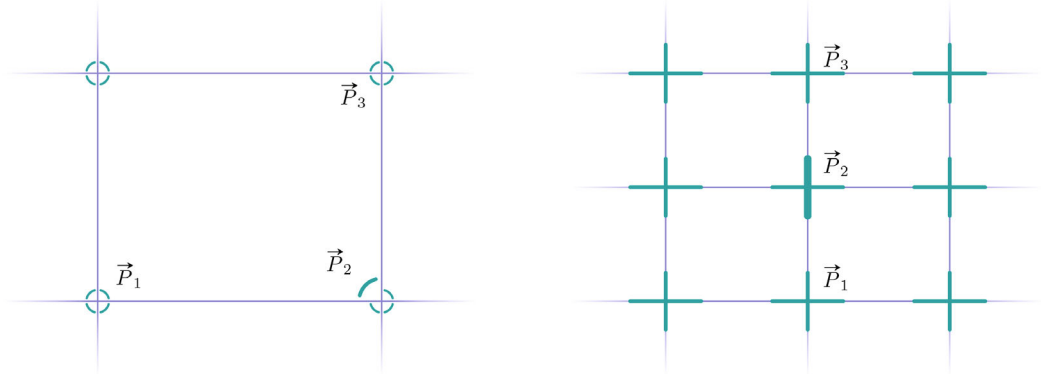

FIGURE 6 Criteria applications for inner cell angle (left) and expansion ratio of control grid (right).

The term s_0 denotes the desired expansion ratio and will be one for the application on the control grid.

$$c_s = \frac{\max\left(|\vec{P}_2 - \vec{P}_1| \cdot s_0, |\vec{P}_3 - \vec{P}_2|\right)}{\min\left(|\vec{P}_2 - \vec{P}_1| \cdot s_0, |\vec{P}_3 - \vec{P}_2|\right)} - 1 \tag{7}$$

For each control point which is within a block or on a regular block connectivity, two instances in 2D cases and three instances in 3D cases are considered. At block edges, which are not part of block connectivities, the stencil allows at maximum one instance per control point. At singularities the stencil is applied for all possible choices of points. The control grid cell sizes near singularities are decoupled from the rest of the control grid by locally disabling this criterion.

4.1.3 | Expansion ratio—sampling

The expansion ratio is estimated for the sampling positions along grid lines of the control grid. This is the only criterion that depends on the BSpline sampling positions. For its application, BSpline curves with the modifications presented in Section 2 are used. As only a curve is used to estimate the distances, the sampled points are not influenced by other control points of the surface or volume. Nevertheless, these distances approximate distances in the final grid sufficiently. The calculation is the same as for control grid expansion ratios with an option to prescribe a desired expansion ratio s_0. For each edge in each control block, the number of sampled points is defined by the number of corresponding sampling positions. At each of these sampled points this criterion is applied. Exceptions are the first and last point, if they are part of a connectivity or already considered by a previous block. At singularities, multiple expansion ratios for possible sets of neighboring curves are calculated.

4.2 | Weights and optimization scheme

The general fitness function is given by Equation (4). It is parametrized by the criteria type specific factors w_{ct} and transfer functions t_{ct}, the block resolution dependent factors w_m and the local factors w_p. For the following examples the factor w_m is disabled by setting it to one and all transfer functions t_{ct} are set to the identity. When the ratio between sampled points and control points show big differences between the split blocks of the control grid, the weight w_m gets relevant. With the transfer functions t the methods stability can be improved by increasing the penalty for nearly folding meshes. Both aspects are not relevant for the following examples. The resulting fitness function is reduced to:

$$\epsilon = \sum_{ct} \sum_{i} \left(w_{ct} \cdot w_p \cdot c_{ct,i}(\vec{p})\right)^2 \tag{8}$$

The weights for the reduced fitness function (8) were chosen according to Table 1.

Using the chain rule, all partial derivatives of weighted criteria evaluations with respect to the parametrization vector of the intermediate layer can be calculated. The optimization itself is performed by the Levenberg-Marquardt method[19,20] using the conjugated gradient method[21] to solve the linear system of equations. For the shown examples, a fixed number of iterations is prescribed. In an actual implementation a stopping criterion should be used to avoid unnecessary iterations. For the coarse optimization 300 iterations are done and other optimizations use 50 iterations. Typical optimizations converge in a fraction of the maximum iterations. Defining cancelation criteria is not in scope of this article. In the implementation one iteration refers to one test of a parameter vector.

5 | DETAILED EXAMPLE

This section shows a simple 2D test case for which the geometry is defined by the union of a circle and a rectangle. Aside from the meshing result, the initialization and configuration are addressed and intermediate steps will be visualized. The meshing objective is to create a grid with a boundary layer near the circle and the linear walls connected to the circle. In this layer the mesh should be oriented normal to the geometry and the first distance should be a constant. The parameters for this application are the number of points along four different edges in the topology and the first distance for the boundary layer. The geometry is shown in Figure 7 and the user parameters are labeled in the figure as s_{1-4} and d_1.

5.1 | Initialization

The initialization step needs a block-structured control grid and a set of BSpline sampling positions to build up an intermediate layer representation. For this application a boundary layer block is created and a center block is added. These

TABLE 1 Criteria type specific weights.

Criterion	w_{ct}
Inner cell angle	0.01
Inner cell angle (singularity)	0.015
Expansion ratio control grid	0.05
Expansion ratio control grid (singularity)	0.2
Expansion ratio sampling	1
Expansion ratio sampling (singularity)	1.5

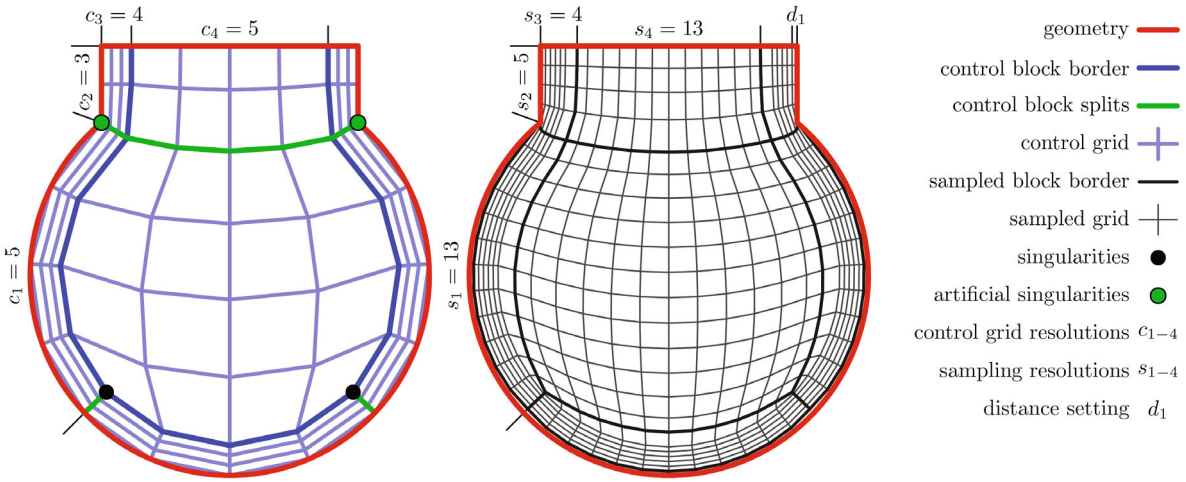

FIGURE 7 Initial control grid and sampling of the constructed intermediate layer without optimization.

blocks are visualized in Figure 7 on the left side. This combination creates two singularities in which three edges connect to one node. The boundary layer block is extruded with an equidistant distribution using $c_3 = 4$ points and a multiple of the prescribed first boundary layer distance d_1 as step width. The upper border of the center part is initialized linearly and equidistantly. The other borders of the center grid are copied from the boundary block's inner border and the remaining points are placed by the transfinite interpolation. With the four points in the extrusion of the boundary layer, one point is placed on the geometry and the second is used to control the normal and the mapping from BSpline sampling position distances to sampled distances. The last point defines the inner block edge and the second but last point is used to let the method adjust spacings near the two singularities. For the top region $c_2 = 3$ vertical control points are sufficient to give the method freedom to adjust distances near the intersections between circle and rectangle edges. The vertical and horizontal control block resolution of the circle are set to $c_1 = 5$ and $c_4 = 5$, respectively. Two points are used by the method to control the distances at the singularities and the intersection of the circle and the lines. The control grid distances at the singularities tend to get small and will not provide a considerable contribution to the spatial resolution of the control grid. Therefore, the remaining center point is used to provide an additional support station along the circle.

In order to preserve the intersection points and avoid an ambiguous geometry association during the refinement and projection steps, the intersection points are threaded as singularities (see artificial singularities in Figure 7, sampled by the method for block A in Section 2.4). The reciprocal of the factor to calculate the first control grid distance is used to fix the second position of the BSpline sampling vector for the boundary normal direction. All other sampling positions are distributed with equidistant spacings in terms of the BSpline parameter. For demonstration purposes the sampled grid is also shown in Figure 7. Especially with a less equally spaced distribution at the circle, the distances between the sampled grid and the geometry will increase. Therefore, a refinement with the factor two will be used later on for the horizontal and vertical distributions within the circle.

When the method is initialized with this control grid, it will split the two blocks into seven blocks at the real and artificial singularities. These splits are highlighted in Figure 7.

5.2 | Control point constraints and coarse optimization

In this 2D application the basis configuration for each control point is free movement in two spatial directions. Depending on the constraints, the control points may move in different directions and the non-fixed sampling positions are adjusted during the coarse optimization. In the following parts, constraints are added and a coarse optimization is performed to visualize the influence of these constraints. In a real application, the coarse optimization is only applied once with the final set of constraints.

Without any further constraints the geometry association is unknown to the method and the topology is the main constraint. Figure 8 shows the result of the coarse optimization when the required geometry constraints are not specified. Only two control points are fixed in this result to set the scale and rotation, as the method is rotation and scale invariant. For the spacings in the control grid equal distances in the proximity of singularities (including the artificial ones) are the main objective. Due to the decoupling of the control grid and the final grid with the sampling positions, theses inhomogeneous spacings do not transfer to the sampled mesh. Based on the criteria types the mesh is close to be an ideal mesh. The expansion ratio of the cells is close to one for the whole region and all inner cell angles are close to 90°.

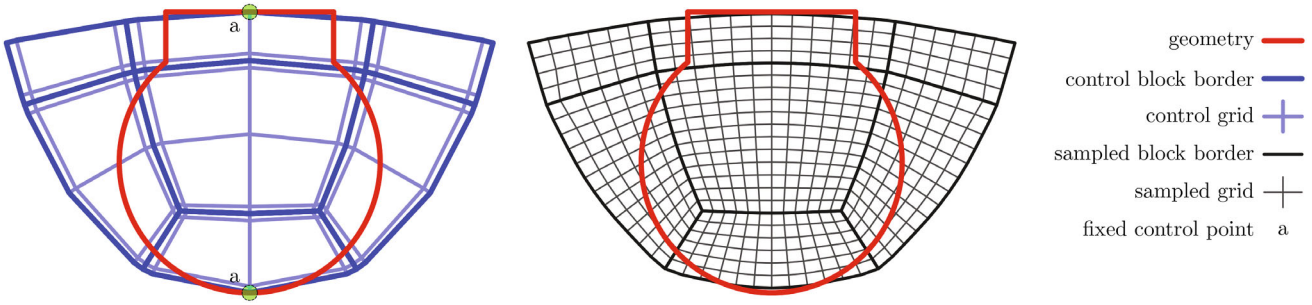

FIGURE 8 Optimization result without refinement and geometric constraints.

As placing points on the geometry is one of the requirements on a mesh, specifying the geometry constraints is a required step. For this example, control points on the circles may only move on circle and along the rectangle they must follow a coordinate axis. The intersection points of the circle with the lines and the corners of the rectangle are fixed points. These geometry constraints are transferred to refined points and used for the final projection. Applying the coarse optimization with these constraints leads to the control grid and the sampling shown in Figure 9. The sampled grid is only close to the real geometry. The geometry constraint is only applied to the control grid and a BSpline interpolates its control points only in special cases. Increasing the control grid resolution decreases the distance between the sampled mesh and the geometry. A final projection will always be necessary. As no refinement is defined yet, the projection would operate on the shown mesh. Without special constrains the wall normals are close to ideal due to the inner cell angle criterion. But the first distance is not set and as such the mesh has some variation in the distance and it is large compared to the target distance.

In order to set the first distance in the boundary layer and prescribe a wall normal edge direction, the second row of control points are constraint. Each second point is restricted to be orthogonal to the geometry at the location of the first point on the geometry while keeping the distance set in the initial control grid. The points of the second row on the upper edge and next to the intersection points are fixed as their base points are also fixed. Lastly the second sampling position for the boundary layer, which has been chosen create the desired first distance based on the control grid, must not be changed by the optimization. With these set of constraints applied, the coarse optimization result is shown in Figure 10.

While iterating to a suitable spatial block distribution and generating a desired boundary layer, the distance between the geometry and the sampled mesh is large compared to the cell sizes leading to the requirement of a finer control grid.

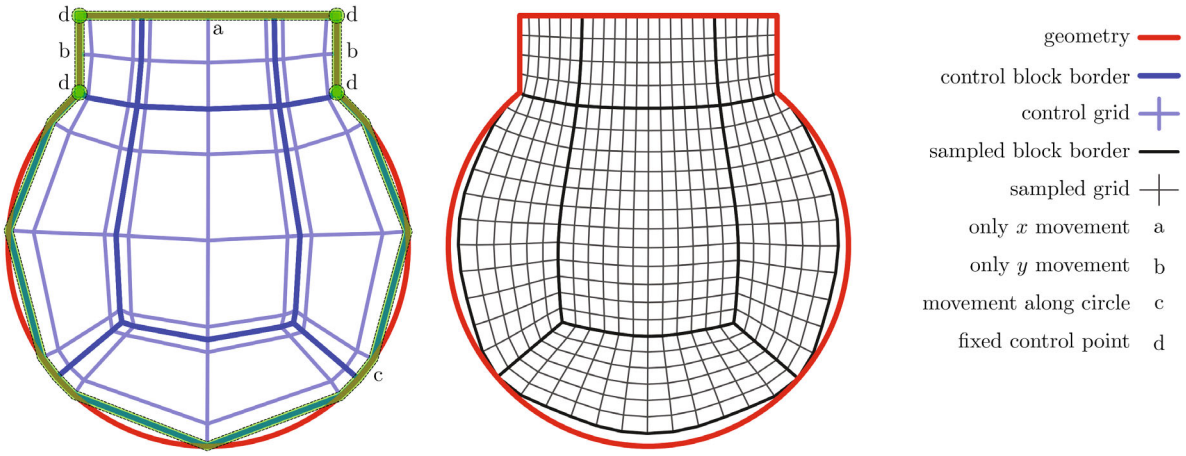

FIGURE 9 Optimization result without refinement using geometric constraints only.

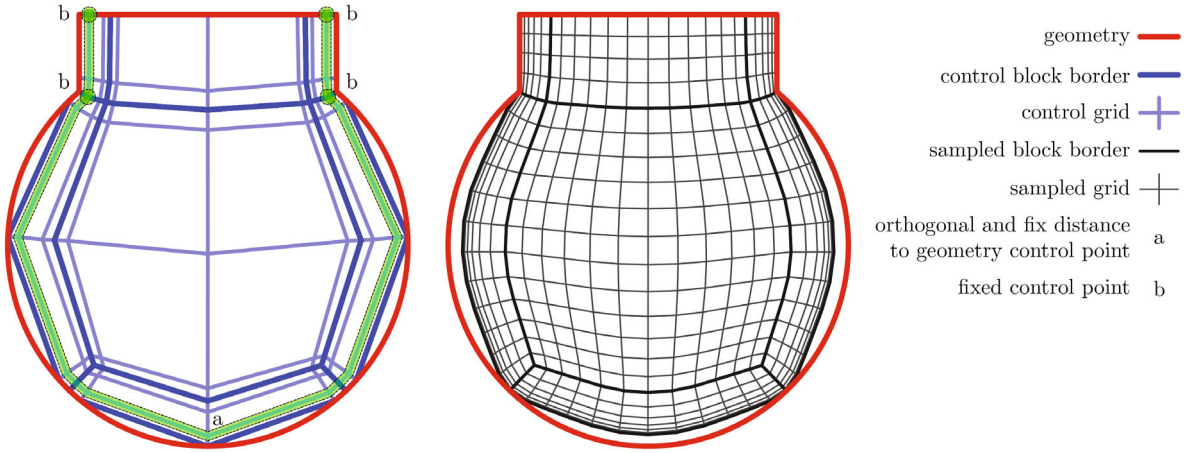

FIGURE 10 Optimization result without refinement with additional constraints based on Figure 9.

Introducing a refinement has an influence on the expansion ratios of the control grid and the sampling. For control grid expansion ratios all segment distances are divided by the remaining refinement factor. The expansion ratios of the sampling are estimated by inserting the refinement nodes as equidistant nodes along each segment. With the refinement of factor two for the horizontal and vertical segments the coarse optimization iterates to the control grid shown in Figure 11. Due to the changes in the calculation of the expansion ratios the control grid differs to the one shown in Figure 10. For the visualization of the sampling, all nodes, which will exist after the refinement, are created by linear combinations of the coarse control nodes. This leads to straight segments which will be curved when the points are inserted and optimized.

5.3 | Refinement

For this small example, the initial control grid could be created with a higher resolution and as such reduce the distance between sampling and geometry. In real-world applications this may not be an option, as the control grid resolution has a major influence on the complexity of the optimization steps. The alternative is using an automatic refinement. The refinement step introduces additional control points along edge segments, which can be optimized and increase the resolution based on the coarse result.

Within a refinement step all coarse control points and all sampling positions are fixed. Along an edge, where a refinement is defined, new control points are initialized linearly. Segments, which are close to a singularity or are used to prescribe a distance, are excluded. The constraints of the newly inserted control points are derived from the original coarse control points. With a new line in the control grid, the number of possible criteria instances increases. By fixing the control points from the coarse optimization and fixing the BSpline sampling positions, most of the criteria instances have no degrees of freedom and are removed from the fitness function. Figure 12 shows the optimization result of the vertical refinement. This refinement generates two horizontal groups a and b, which are optimized independently. In the refined region, the distance between the geometry and the sampled mesh is decreased noticeably.

After the refinement optimizations for the horizontal and vertical refinements a fine control grid is assembled. Where the groups of refinements intersect, missing control points are generated using the transfinite interpolation. The final control grid is shown in Figure 13 with the interpolated points highlighted.

5.4 | Sampling position optimization

When a refinement is used, the sampling positions were fixed within refinement optimizations, which is a requirement to make them independent from each other. With the newly inserted control points, the sampling positions may not be optimal anymore. In the sampling position optimization, only the sampling positions are parameters to the optimization step. With all control points fixed, the expansion ratio of sampling is the only criteria type with remaining degrees of

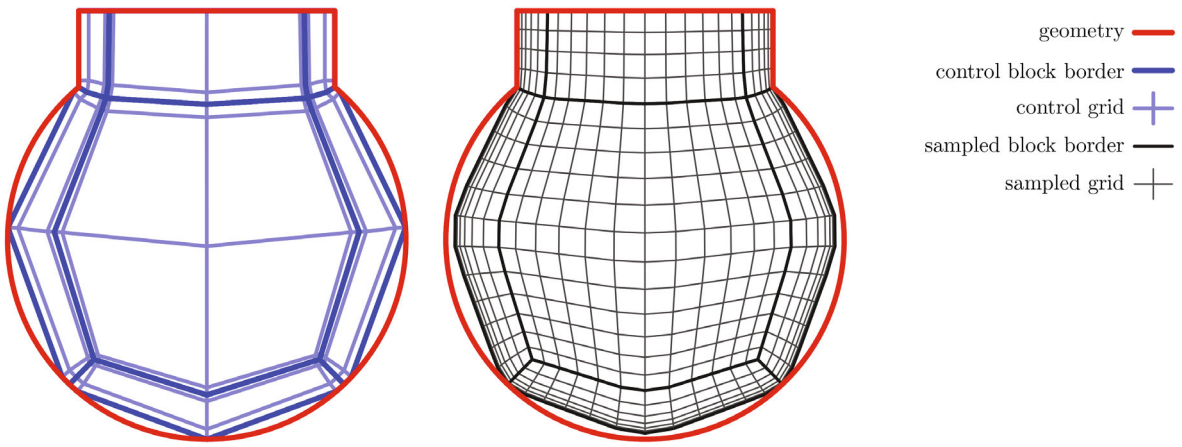

FIGURE 11 Coarse optimization result with refinement settings.

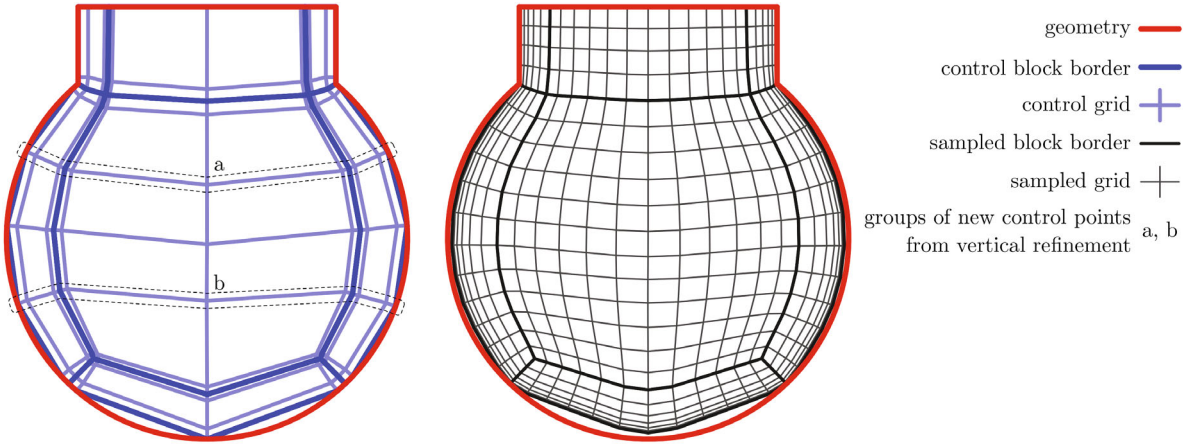

FIGURE 12 Optimization result of refinement in vertical direction with two independent groups of control points.

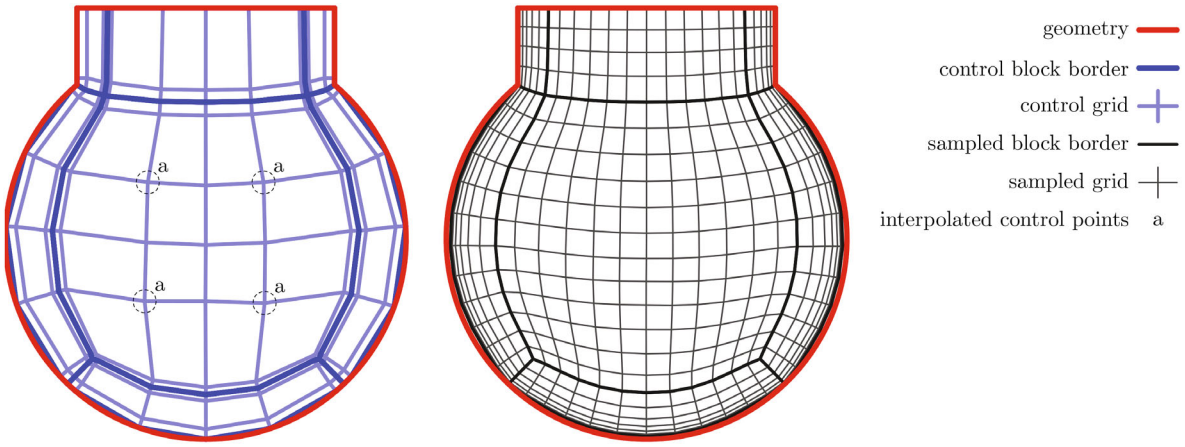

FIGURE 13 Merged control grid after refinements.

freedom. The number of instances of this criterion may be considerably larger compared to the coarse optimization step due to the refined control grid. Figure 14 compares the sampling from the previous step with the result of the optimization.

5.5 | Projection and displacement

With the optimization steps finished, a grid can be sampled. All previous steps only operate in the domain of the intermediate layer. As such all presented sampled representations are only for visualization purposes, but not used by the method. Before using the grid for calculations, geometry constraints must be fulfilled. Based on the intermediate layer's constraints, projection targets can be extracted. All geometry associated points are projected and the displacement vector is propagated based on inverse distance into the interior of the block. Figure 15 shows the sampled grid and the projected grid. In the zoomed part of the image the arrow shows the projection of one grid point, which displaces in the order of two times the first cell size.

5.6 | Grid quality metric

The quad skew quality metric[22] can be used to quantify the quality of the resulting mesh. Figure 16 shows this metric as contours and as histogram data. With more than 78%, most cells have a quality metric above 0.99 and are close to

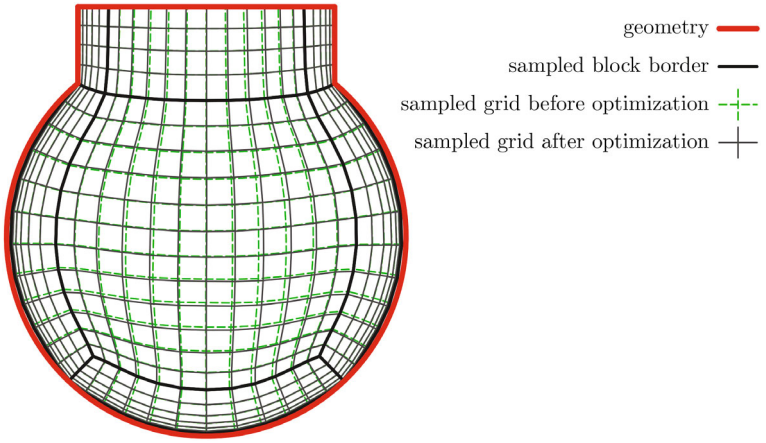

FIGURE 14 Sample position optimization: before and after.

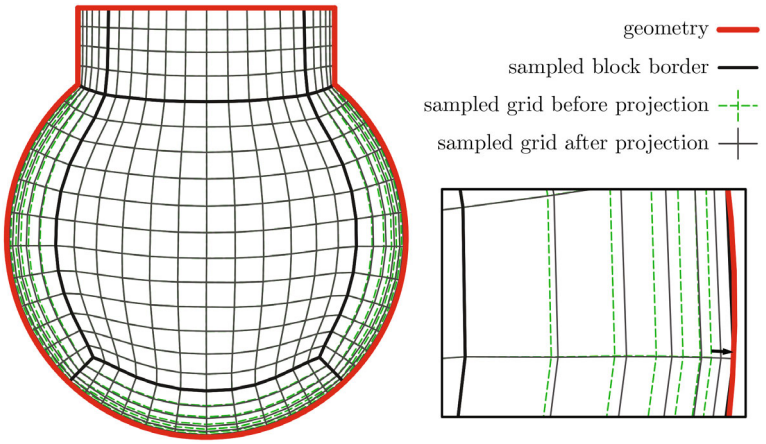

FIGURE 15 Projection and displacement: Before and after.

the optimum 1.0 of the metric. The four worst cells were rated between 0.77 and 0.78 and can be located close to the singularities. At singularities the skew quality cannot be optimal for all involved cells. Nevertheless, the usage of more points in the wall blocks would allow the edges near the singularities to spread out more evenly and reduce the skewness for the involved cells.

6 | FURTHER EXAMPLES

6.1 | Ball head

The test case shown in Section 5 can be extended to a 3D test case. For the geometry this is done by creating surfaces of revolution about the center vertical axis using the original curves. The block topology uses a butterfly configuration at the top of the cylinder, which ends near the sphere surface with a final block filling the remaining gap. This topology is reduced to a quarter by introducing periodic boundaries. Figure 17 shows the block topology and highlights the singularity edges, which intersect in the lower part of the sphere forming a 3D singularity.

Distances and constraints at the cylinder and the sphere are chosen similar to the 2D test case. The periodic constraint for the periodic faces and a line constraint along the rotation axis for points on it are added.

The methods application in 3D is performed the same way as in 2D. Within the merging of refined grid nodes three refinements can intersect in cell of the coarse control grid. For such a cell, transfinite interpolations are used to create the missing control points on the sides of the coarse cell. With these sides given another transfinite interpolation is used to

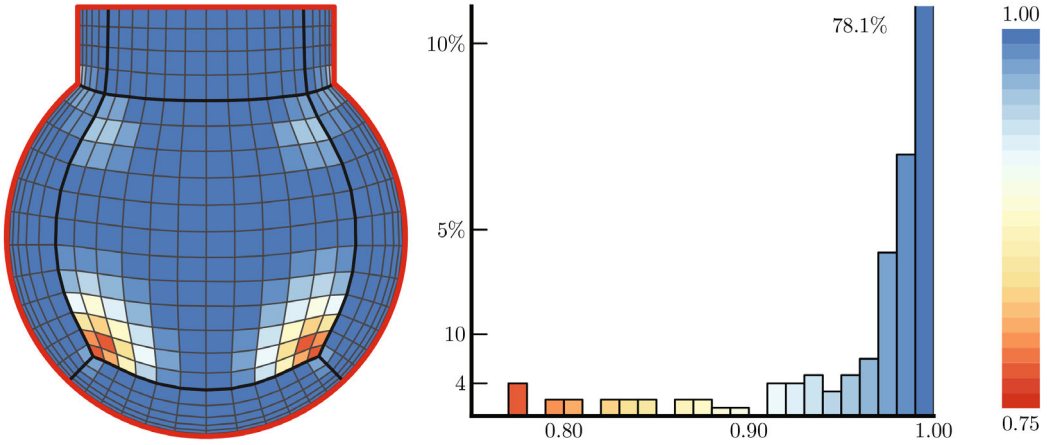

FIGURE 16 Skew quality metric.

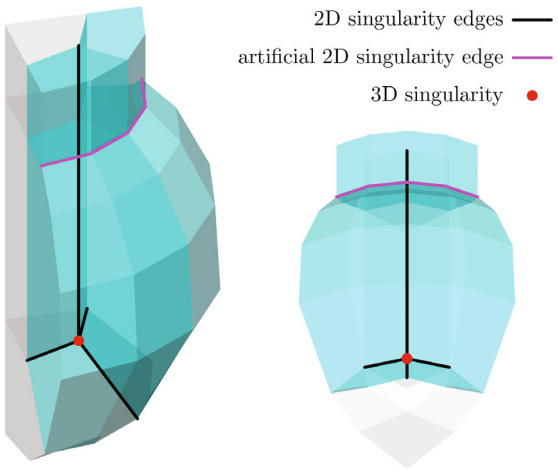

FIGURE 17 Ball head topology and singularities. Shaded faces originate from the initial control grid.

set the control points in the interior of the coarse cell. Applying the method results in a smooth mesh in the interior and across the periodic boundary with the desired wall resolution. Also, the resulting mesh at one of the periodic boundaries is similar to the result of the 2D test case. Figure 18 presents the 3D mesh and shows a comparison to the 2D test case.

6.2 | T106C cascade

The T106C cascade[23] is a turbine blade airfoil with a given linear blade periodicity. By specifying an inflow and an outflow position, a blade passage can be meshed, which includes one airfoil and has a periodic boundary to account for the blade periodicity. A topology for turbine blades is used. It sets default values for control block resolutions and refinements, which were not changed. The topology uses an O-Grid surrounding the airfoil for wall orthogonality and distance control. Also, it allows changes in distances and the number of points in the airfoil's orthogonal direction without influencing the rest of the topology. The other blocks fill up the remaining space and can be arranged to create near orthogonal cells in the passage between airfoils and in the inlet and outlet region.

The user specifies final block resolutions and desired distances normal to the blade and tangential to the blade at the leading and trailing edge. With the distances given, the initial control grid is generated automatically using algebraic methods, mostly orthogonal offsets, straight lines and transfinite interpolations. Figure 19 shows the initial control grid, the topology used, and the final control grid after the optimization steps. In the near zoom onto the trailing edge, fixed points for defining orthogonal and tangential distances and the trailing edge position are within the highlighted region.

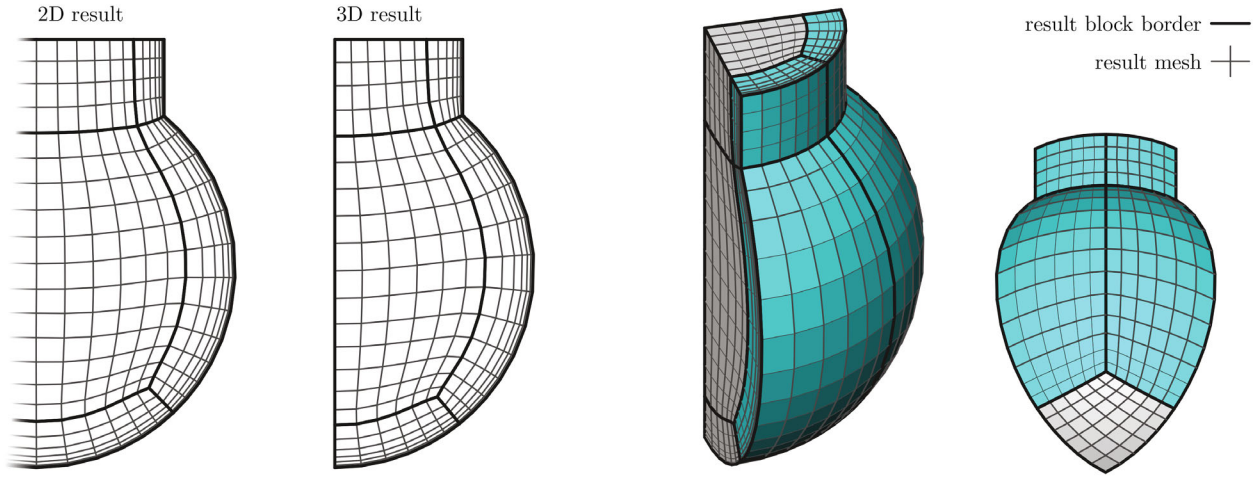

FIGURE 18 Ball head meshing result and comparison to previous 2D grid.

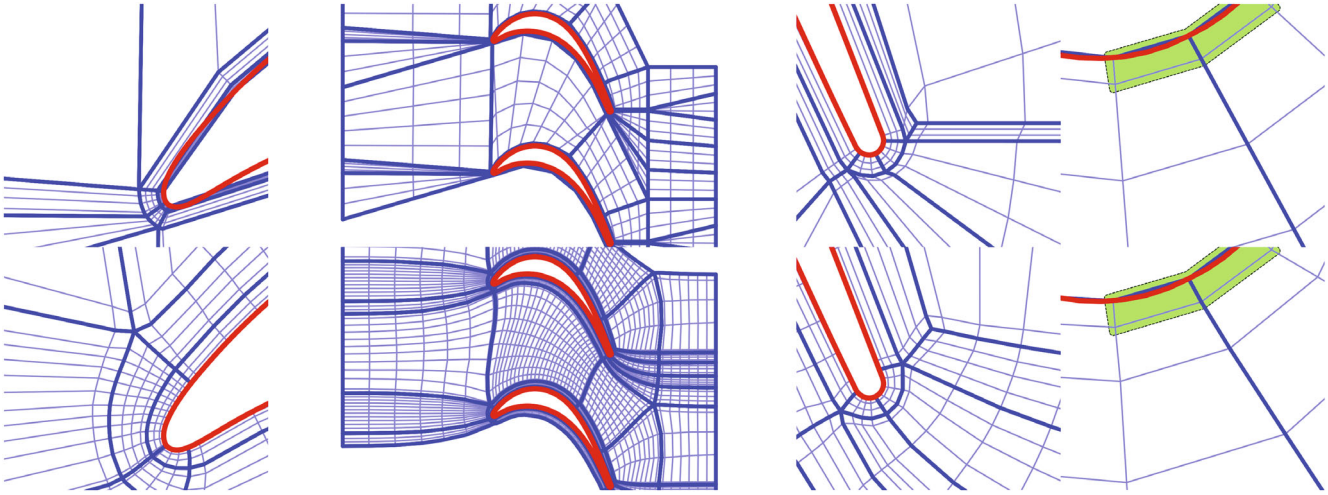

FIGURE 19 T106C: Control grid duplicated to two passages; top: initial, bottom: optimization result.

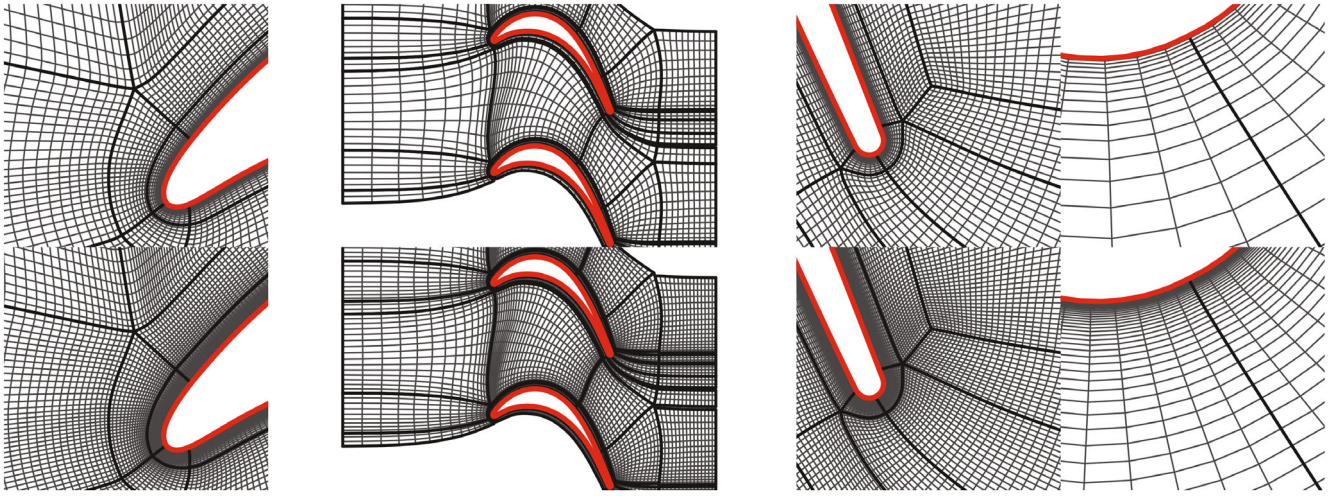

FIGURE 20 Base grid generation and finer resampled grid (the second column shows every fourth grid line).

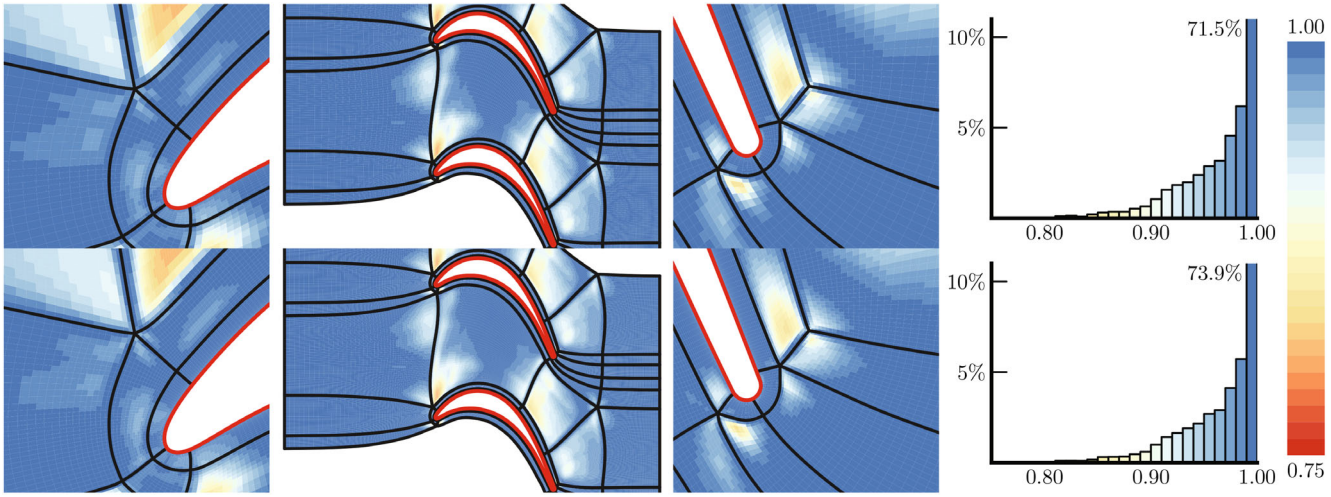

FIGURE 21 Skew quality metric for two samplings of the same control grid.

Based on the final control grid, multiple meshes can be sampled by changing desired distances and result block dimensions and repeating the sampling position optimization step of the method. Figure 20 shows details of the base result and a resampled result, where all block resolutions are multiplied by $\frac{3}{2}$ and distance settings are divided by $\frac{3}{2}$, except for the O-Grids normal direction, for which the double resolution is used while reducing the first distance to a quarter.

For both sampled results and similar viewports, Figure 21 illustrates the quad skew quality metric.[22] With values above 0.8 both cases result in high-quality meshes. The lower values for the skew quality metric are located close to singularities. The bottom row shows the results for the mesh sampled with a higher resolution with a higher refinement in the O-Grid. By placing more points in the O-Grid with a high quality metric, the overall portion of cells in the highest portion of the histogram rises by 2.4%

7 | SUMMARY AND CONCLUSIONS

A new method for generating block-structured meshes is presented in this article. The required input, which is usually abstracted from the user by application class, for example, compressor or turbine blade passages, consists of a coarse, non-folding control grid in the desired topology and constraints on control grid nodes. Internally, the mesh generation is performed by a multi-stage optimization process based on a degree of freedom reduced intermediate layer, which is built from the initial control grid. With a BSpline based sampling, the intermediate layer is used to create a fine mesh. After an adaption to become boundary conforming, the resulting mesh is finished.

With a suitable initial control grid generation, the user parameters can be reduced to setting desired point counts and distances. Additional influence on the meshing result can be achieved by locally adjusting criteria weights and targets, for example, setting a desired expansion ratio and increasing its influence for the wall normal direction within a block.

Additionally, the method allows for similar grid generation with different resolutions and distance settings. With this feature, sets of comparable meshes with a constant spatial block distribution can be created. These meshes provide a flexible basis for grid dependency studies with or without fixed first distances. In the context of Discontinuous Galerkin methods, higher order elements can be sampled with a similar post processing step.

The basic method shown in this article can be extended by other grid criteria and further improved by handling differences between control grid resolutions and final grid resolutions as well as providing nonlinear scaling functions to adjust convergence behavior. With these changes, a possible application for this method is the meshing of turbomachinery passages in compressor and turbine rows. In these applications, the method enables wide geometric variation while preserving a high grid quality within automatic design optimizations.

DATA AVAILABILITY STATEMENT

Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

ACKNOWLEDGMENT

Open Access funding enabled and organized by Projekt DEAL.

ORCID

Marcel Sauer https://orcid.org/0009-0004-2601-1434
Christian Morsbach https://orcid.org/0000-0002-6254-6979

REFERENCES

1. Delaunay B. Sur la sphère vide. A la mémoire de Georges Voronoï. *Bull Acad Sci URSS Classe Des Sci Math*. 1934;6:793-800.
2. Chew LP. Constrained delaunay triangulations. *Algorithm*. 1989;4(1):97-108.
3. Shewchuk JR. Tetrahedral mesh generation by Delaunay refinement. Proceedings of the Fourteenth Annual Symposium on Computational Geometry. 1998:86-95. doi:10.1145/276884.276894
4. Maréchal L. All hexahedral boundary layers generation. *Procedia Eng*. 2016;163:5-19. doi:10.1016/j.proeng.2016.11.007
5. Löhner R, Parikh PC. Generation of three-dimensional unstructured grids by the advancing-front method. *Int J Numer Methods Fluids*. 1988;8(10):1135-1149. doi:10.1002/fld.1650081003
6. Blacker TD, Meyers RJ. Seams and wedges in plastering: a 3-D hexahedral mesh generation algorithm. *Eng Comput*. 1993;9:83-93. doi:10.1007/BF01199047
7. Ali Z, Tucker P. Multiblock structured mesh generation for turbomachinery flows. Proceedings of the 22nd International Meshing Roundtable. 2014:165-182. doi:10.1007/978-3-319-02335-9_10
8. Allen CB. Towards automatic structured multiblock mesh generation using improved transfinite interpolation. *Int J Numer Methods Eng*. 2008;74:697-733. doi:10.1002/nme.2170
9. Armstrong CG, Fogg HJ, Tierney CM, Robinson TT. Common themes in multi-block structured quad/hex mesh generation. *Procedia Eng*. 2015;124:70-82. doi:10.1016/j.proeng.2015.10.123
10. Thompson JF, Thames FC, Mastin C. Automatic numerical generation of body-fitted curvilinear coordinate system for field containing any number of arbitrary two-dimensional bodies. *J Comput Phys*. 1974;15(3):299-319. doi:10.1016/0021-9991(74)90114-4
11. Steger J, Sorenson R. Automatic mesh-point clustering near a boundary in grid generation with elliptic partial differential equations. *J Comput Phys*. 1979;33(3):405-410. doi:10.1016/0021-9991(79)90165-7
12. Hasanzadeh K, Laurendeau E, Castonguay P, Leblond D. Framework of multiblock structured grid smoothing algorithms for aircraft configurations. *J Aircr*. 2018;55(4):1542-1550. doi:10.2514/1.C034581
13. Thomas PD, Middlecoff JF. Direct control of the grid point distribution in meshes generated by elliptic equations. *AIAA J*. 1980;18(6):652-656. doi:10.2514/3.50801
14. Zhang Y, Jia Y, Wang SS, Chan H. Boundary treatment for 2D elliptic mesh generation in complex geometries. *J Comput Phys*. 2008;227(16):7977-7997. doi:10.1016/j.jcp.2008.05.008
15. Wang F, Mare DL. Mesh generation for turbomachinery blade passages with three-dimensional Endwall features. *J Propul Power*. 2017;33(6):1459-1472. doi:10.2514/1.B36356
16. Cadence Design Systems Inc. IGG AutoGrid5 v16.1 User Guide. 2021 EN202103101547.
17. Sauer M. An optimization based approach to multi-block structured grid generation. Paper presented at: ECCOMAS–ECFD 2018–6th European Conference on Computational Mechanics (Solids, Structures and Coupled Problems) / 7th European Conference on Computational Fluid Dynamics. 2018 https://elib.dlr.de/120879/
18. Piegl L, Tiller W. *The NURBS Book*. 2nd ed. Springer-Verlag; 1997.
19. Levenberg K. A method for the solution of certain problems in least squares. *Q J Appl Math*. 1944;2(2):164-168.
20. Marquardt DW. An algorithm for least-squares estimation of nonlinear parameters. *J Soc Ind Appl Math*. 1963;11(2):431-441. doi:10.1137/0111030
21. Hestenes MR, Stiefel E. Methods of conjugate gradients for solving linear systems. *J Res Natl Bur Stand*. 1952;49(6):409-436. doi:10.6028/jres.049.044
22. Knupp PM. Algebraic mesh quality metrics for unstructured initial meshes. *Finite Elem Anal Des*. 2003;39(3):217-241. doi:10.1016/S0168-874X(02)00070-7
23. Michálek J, Monaldi M, Arts T. Aerodynamic performance of a very high lift low pressure turbine airfoil (T106C) at low Reynolds and high Mach number with effect of free stream turbulence intensity. *J Turbomach*. 2012;134(6):061009-1-0061009-10. doi:10.1115/1.4006291

How to cite this article: Sauer M, Morsbach C. An optimization based multi-block-structured grid generation method. *Int J Numer Methods Eng*. 2023;1-21. doi: 10.1002/nme.7308