Volumetric Geometry Reconstruction of Turbine Blades for Aircraft Engines

David Großmann[1] and Bert Jüttler[2]

[1] MTU Aero Engines GmbH, Munich, Germany
[2] Johannes Kepler University, Institute of Applied Geometry, Linz, Austria

Abstract. We present a framework for generating a trivariate B-spline parametrization of turbine blades from measurement data generated by optical scanners. This new representation replaces the standard patch-based representation of industrial blade designs. In a first step, the blade surface is represented by a smoothly varying family of B-spline curves. In a second step, the blade is parametrized by a trivariate B-spline volume. The resulting model is suitable for numerical simulation via isogeometric analysis, as well as for a fully automatic structured mesh generation with standard finite elements. We focus on the industrial applicability of the framework, by using standard turbine blade features throughout the process.

Keywords: volumetric geometry reconstruction, turbine blades, trivariate B-Splines, numerical simulation, isogeometric analysis

1 Introduction

The commercial activities of MTU Aero Engines focus on developing, manufacturing and repairing turbine engines for aircrafts. The volumetric B-spline parametrization – which is discussed in the present paper – enables us to explore new approaches to the challenging tasks of high-quality and efficient numerical simulations of turbine blades.

First, the new numerical simulation approach of isogeometric analysis (IGA), introduced by Hughes et al. [10], uses geometry mappings represented by volumetric NURBS parametrizations, and the finite-dimensional spaces needed for the Galerkin projection are defined with the help of these parametrizations. As a major advantage, only one representation of the blade geometry is required, which is used throughout the entire process of design, simulation, and manufacturing. Consequently, the geometrical errors introduced by approximating the blade geometry by finite element meshes are eliminated and the number of unknowns at the coarsest discretization level is significantly decreased. This new approach seems to be particularly well suited for blade geometries, since they are less complex than general CAD models.

Second, the volumetric description can be used for automatically partitioning the blade into several volumetric patches. Consequently, a fully automatic structured mesh generation for standard finite elements becomes possible.

B-spline volumes have been discussed in the classical literature in Computer Aided Design, e.g see [9]. The existing literature on volume parametrizations by B-splines concentrates mostly on applications to object modeling via free-from deformations. More recently, the construction of B-spline volume parametrizations for isogeometric analysis was discussed. The paper [2] designs B-spline volumes by sweeping and uses them for isogeometric analysis. In [14], the authors use harmonic functions to generate a spline parametrization of general cylinder-type objects. The regularization of B-spline volumes is addressed in [18].

With the focus on the industrial applicability, we present a process that generates a volumetric B-spline parametrization of a turbine blade. Our approach is fully compatible with the natural process flow in turbine blade engineering. Furthermore we can cover the industrial standards and requirements of a geometric blade model, and the intermediate results and methods are useful for performing standard CAD and CAE operations.

After presenting an outline of our approach, the paper describes the steps needed for the volumetric model generation. We conclude with some final remarks.

2 Problem Specification and Outline

We assume a triangular mesh of a turbine blade, generated by an optical measurement system or by sampling a standard (surface) CAD description of the blade. Therefore, our framework is usable for the integration of physical objects and geometric models.

Based on the mesh representation of the blade, we generate a volumetric tensor-product B-spline model of the blade by an almost automatic geometry reconstruction process, which requires only very little interaction with the user. The surface parametrization of the resulting model is suitable for a new parametrization of the blade surface which replaces the traditional blade parametrization with multiple trimmed surface patches. The volume parametrization of the model is suitable for high-quality numerical simulations using the isogeometric analysis or by a fully automatic structured mesh generation with standard finite elements.

Fig. 1 shows a disk with blades in a modern turbine, the standard position of a single blade and the typical shape of a blade. For the sake of brevity we shall denote both turbine blades and compressor blades as turbine blades or blades only.

The remainder of this section summarizes the proposed modeling framework. Starting with a triangular mesh $\mathcal{M}$ of a blade as input, we generate a volumetric tensor-product B-spline model in the following steps:

1. The user defines the number of slicing surfaces and the desired number of degrees of freedom for the tensor-product B-spline volume.
2. We generate two families of slicing surfaces $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$ which intersect the blade in well-behaved slices $\mathcal{C}_\omega$ between the blade boundary and the tip.

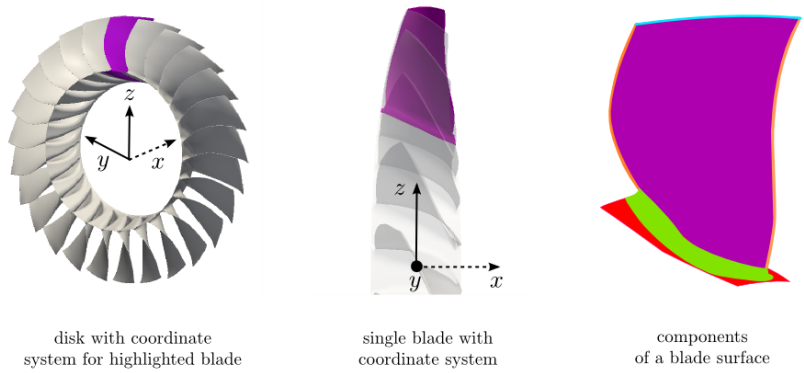

Fig. 1. From left to right: A disk with blades of a modern turbine engine. For our framework we assume the standard position of a disk, where the x-axis equals the turbine axis in the direction of the current. Considering a single blade, the z-axis equals the radial direction and the y-axis completes the right-handed orthonormal system. The shape of a blade consists of several parts, the endwall (red), the fillet (green), the airfoil with side parts (magenta) and edge parts (orange), and the tip (light blue).

3. We segment the slices $\mathcal{C}_\omega$ into edge and side parts, preparing them for the generation of a volumetric tensor-product B-spline model and considering the blade topology. For the airfoil part, the transitions between the four parts are called *wedge points*.
4. We fit the slices $\mathcal{C}_\omega$ by B-spline curves $\mathbf{c}_\omega(u)$ with identical degrees d and knot vectors $\mathcal{U}$. We use an iterative process to optimize the parametrization. The parameters associated with the wedge points (which we call *wedge knots*), however, are kept fixed.
5. We split the closed B-spline curves $\mathbf{c}_\omega(u)$ at the wedge knots into four boundary curves and generate surfaces $\mathbf{s}_\omega(u, v)$ by extending the boundary curves bilinearly to the inner part of the blade using Coons patches.
6. We interpolate the surfaces $\mathbf{s}_\omega(u, v)$ in sweeping direction w for a volumetric tensor-product B-spline model $\mathbf{v}(u, v, w)$ of the blade. An additional B-spline block at the bottom completes the model.

3 Slicing Surfaces

We represent the slicing surfaces $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$ as implicitly defined algebraic surfaces. This allows us to use their advantages compared to parametric surfaces, e.g. no data parametrization is needed for fitting processes and relatively simple algorithms for computing intersections with meshes and for blends are available.

An algebraic spline surface $\mathcal{F}$ is defined as the zero level set of a tensor-product spline function of (tri-) degree d ($d \geq 2$),

$$f(x, y, z) = \sum_{(i,j,k)\in\mathcal{J}} c_{i,j,k}\, N_{i,d}(x)\, N_{j,d}(y)\, N_{k,d}(z) \tag{1}$$

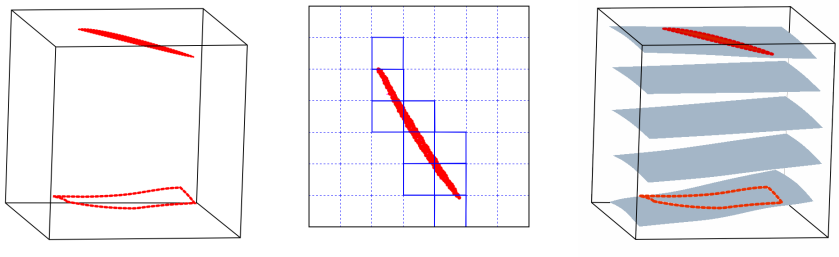

Fig. 2. From left to right: The blade boundary and the tip. The tensor-product splines are defined by three uniform knot sequences (top view). Several level sets $\mathcal{F}_\alpha$ of the scalar field.

with the real coefficients (control points) $c_{i,j,k}$, where $\mathcal{J}$ is the appropriate index set. The basis functions $(N_{i,d}(x)_{i=1,\dots,n_i})$, $(N_{j,d}(y)_{j=1,\dots,n_j})$ and $(N_{k,d}(z)_{k=1,\dots,n_k})$ are B-splines of degree d with respect to uniform knot sequences $\mathcal{X} = (\beta_i)_{i=1,\dots,n_i+1}$, $\mathcal{Y} = (\eta_j)_{j=1,\dots,n_j+1}$ and $\mathcal{Z} = (\zeta_k)_{k=1,\dots,n_k+1}$ for the three coordinate directions x, y and z. These knot sequences partition a bounding box Ω (domain of the tensor-product spline functions) into axis-aligned boxes. By using B-splines as basis functions we can use their advantageous properties such as local support and the increased flexibility compared to polynomials.

3.1 Slicing Surfaces for the Airfoil Part

We construct a family of slicing surfaces $\mathcal{F}_\alpha$ between the endwall ($\alpha = 0$) and the tip ($\alpha = 1$) of the blade, so that they generate well-behaved intersections (single connected components) for the airfoil part of the blade. The surfaces are designed in three steps, which are visualized in Fig. 2:

1. The tip $\mathcal{T}$ is represented by all tip points of the mesh, automatically detected by an adapted region-growing algorithm, while the endwall is represented by the blade boundary $\mathcal{C}_B$.
2. We fit these two data sets simultaneously using the techniques described in [12] (by minimizing the squared algebraic distances of a function f_α, constrained by some additional normal- and tension-terms) and by generalizing the term of minimizing the squared algebraic distances to
$$\sum_{\mathbf{v}\in\mathcal{C}_B} [f(\mathbf{v})]^2 + \sum_{\mathbf{v}\in\mathcal{T}} [f(\mathbf{v}) - 1]^2 \,. \tag{2}$$
The B-spline basis functions are defined on the bounding box with uniform knot sequences for all three dimensions.
3. Finally, a finite subset of the set of level sets of the scalar field f defines the airfoil slicing surfaces
$$\mathcal{F}_\alpha = \{(x,y,z) \in \Omega \;\mid\; f_\alpha(x,y,z) = \alpha\} \quad \text{for } \alpha \in [\bar{\alpha}, 1] \,, \tag{3}$$

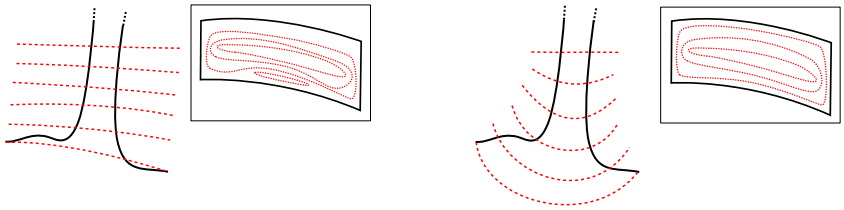

Fig. 3. The intersections of the slicing surfaces and the mesh have to be well-behaved and uniformly distributed in sweeping direction. To meet these conditions, we bend them up from horizontal to vertical positions.

where the value $\bar{\alpha}$ is chosen such that $\mathcal{F}_{\bar{\alpha}}$ $(0 < \bar{\alpha} < 1)$ is the lowest surface intersecting the airfoil part.

3.2 Slicing Surfaces for the Base Part

The family of (airfoil) slicing surfaces $\mathcal{F}_\alpha$ cannot be used for the base part because their intersections with the blade base are not well-behaved, see Fig. 3. Hence we define a second family of slicing surfaces $\mathcal{F}_\beta$ for the base part which we have to bend up from a horizontal position at the airfoil transition to a vertical position at the blade boundary.

In general, the transition curves between endwall and fillet are not useful to slice the base part, because there exists no clear segmentation between them in modern turbine blade geometries. Usually the fillet (and the endwall) is truncated at the front and rear of the endwall to save space and the endwall is designed with streamlined elements transitioned into the fillet, e.g. with a non-axisymmetric endwall contouring [8]. To address this problem we reconstruct the *fillet curve* $\mathcal{C}_F$ as a clear and significant feature for the base part of the blade.

The *fillet curve* $\mathcal{C}_F$ is the intersection of the airfoil and the endwall *before* the fillet has been generated. As a consequence, this curve $\mathcal{C}_F$ is a valuable indicator for the natural flow of the blade and for the transition between the endwall and the airfoil. Fig. 4 illustrates the reconstruction of the fillet curve.

Now, with the help of the fillet curve, we can define a family of ruled surfaces obtained by auxiliary slices and directions vectors. We define the *auxiliary slices* in four steps, see Fig. 5:

1. An algebraic spline surface is fitted to the airfoil data. The intersections of this surface with a family of airfoil slicing surfaces $\mathcal{F}_\alpha$ for $0 < \alpha < \bar{\alpha}$ defines the auxiliary airfoil slices.
2. The blade boundary curve $\mathcal{C}_B$ and the fillet curve $\mathcal{C}_F$ are projected on a cone whose axis equals the turbine axis.
3. A scalar field f is fitted to the curves $\mathcal{C}_B$ and $\mathcal{C}_F$ forced to $f = 0$ for $\mathcal{C}_B$ and $f = 1$ for $\mathcal{C}_F$. We use the same techniques as for the scalar field of the airfoil slicing surfaces.

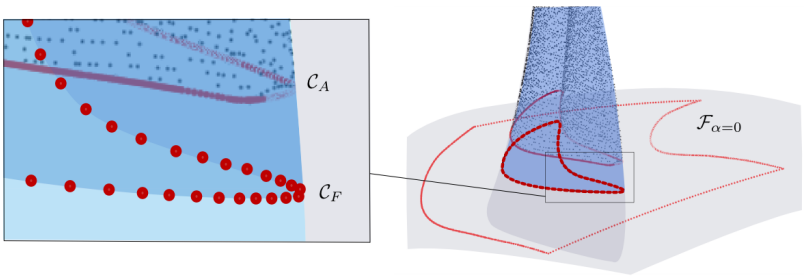

Fig. 4. To reconstruct the fillet curve $\mathcal{C}_F$, we fit an algebraic spline surface to all airfoil data points above the curve $\mathcal{C}_A$ and intersect this surface with the endwall surface $\mathcal{F}_{\alpha=0}$. $\mathcal{C}_A$ denotes the intersection of the lowest airfoil slicing surface $\mathcal{F}_{\alpha=\bar{\alpha}}$ with the blade.

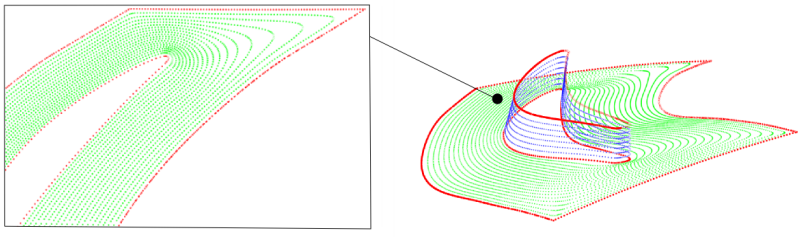

Fig. 5. The endwall surface and the (extended) airfoil surface are associated with fitted scalar fields. There level sets defines the auxiliary slices.

4. The level sets of the scalar field f defines the auxiliary endwall slices which we finally project back on the endwall surface $\mathcal{F}_{\alpha=0}$.

The corresponding *slicing directions* are defined in five steps, see Fig. 6:

1. The auxiliary endwall slices are divided into an inner and an outer part. The transition between them is called *transition slice* and their position is defined by the user.
2. A transition surface is defined as a ruled surface which is produced by the transition slice and directions which originate in the center of the turbine axis.
3. The intersection of this transition surface and the airfoil slicing surface $\mathcal{F}_{\alpha=\bar{\alpha}}$ is called the *bending curve*.
4. The points on the outer auxiliary endwall slices are associated with vertical direction vectors that originate in the center of the turbine axis.
5. The points on the inner auxiliary endwall slices and on the auxiliary airfoil slices are associated with direction vectors that point to the closest points on the bending curve.

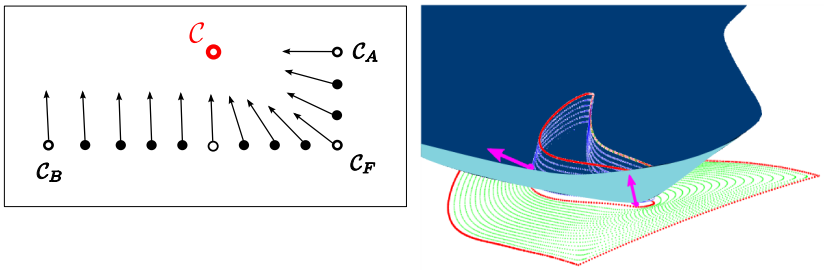

Fig. 6. Left: The slice directions are generated with the help of a bending curve $\mathcal{C}$. Right: A slicing surface.

Each of the auxiliary slices, along with the associated direction vectors, defines a ruled surface. We approximate these ruled surfaces by algebraic spline surfaces. This gives us the second family of slicing surfaces $\mathcal{F}_\beta$, see Fig. 6.

Remark In our framework, the intersection curve of the two algebraic surfaces in step 3 is traced by a predictor-corrector method with curvature-based stepsize control, cf. [3, 7]. This method generates a piecewise linear approximation of the bending curve.

4 Segmentation of the Slices

The (closed) piecewise linear slices $\mathcal{C}_\omega$ are generated by intersecting the mesh with the slicing surfaces $\mathcal{F}_\alpha$ and $\mathcal{F}_\beta$. In addition we have to consider two special cases: The lower blade boundary $\mathcal{C}_B$ can be taken directly from the mesh. For the uppermost slice, we intersect the slicing surface $\mathcal{F}_{\alpha=1}$ with an algebraic spline surface approximating the airfoil data. We cannot use the original mesh data here, since the mesh is slightly topped of and noisy on the transition between the tip and airfoil, due to errors introduced be the optical measurement process. Fig. 7 shows the resulting slices.

The slices $\mathcal{C}_\omega$ are now subdivided into four segments. This prepares them for the generation of the tensor-product spline volume. In addition, it complies with the standard turbine blade geometry, specially the classification of the airfoil into the edge and side parts.

4.1 Airfoil Part

In the airfoil part, the slices are called profiles (referring to the blade design process) and can be segmented by the wedge points into the two edge parts (trailing and leading edge) and the two side parts, see Fig. 8.

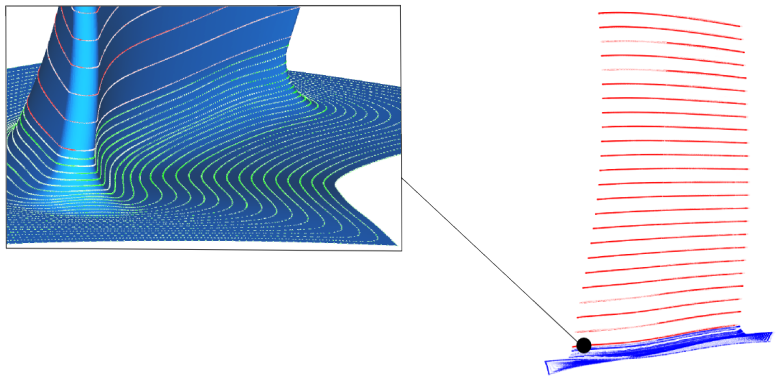

Fig. 7. Piecewise linear slices with (right) and without (left) the underlying mesh.

Edge points In each profile, the two points where the extrapolated medial axis intersects the profile are called the *edge points*. We need to locate them, in order to perform a robust approximation of the (symmetric) wedge points later. Considering only one profile, the following heuristic method constructs an extended medial axis in the edge part of the profile and generates the corresponding edge point:

1. As an initial edge point $\mathbf{p}_{i=0}$ we take the point of the profile with the minimal (respectively maximal) x-coordinate. In general, these points are situated between their associated wedge points.
2. Starting from the edge point $\mathbf{p}_i$, we generate four auxiliary points on each side of the profile. Each pair of auxiliary points has a fixed distance $d, \ldots, 4d$ to the edge point, where d is a certain fraction of the profile length in x-direction.
3. These four points are used to estimate the
4. medial axis of the blade in this slice.
5. An new edge point $\mathbf{p}_{i+1}$ is found by collecting the point of the profile with the minimal distance to the extrapolated medial axis.
6. We continue with step 2 until the edge point has converged to a stable position.

Fig. 8 shows two steps of the edge point iteration.

Wedge points For a robust approximation of the wedge points, we have to consider the following requirements, which are due to the needs of our industrial application.

1. In general, the curvature is not a distinctive feature for a wedge point of a profile; less for the leading edge than for the trailing edge and less for turbine blades than for compressor blades.

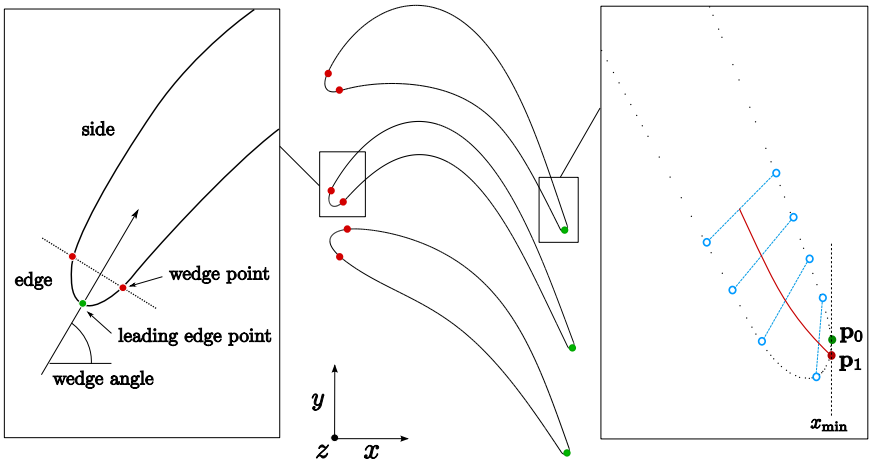

Fig. 8. From left to right: Some design parameters for the airfoil profiles. Three different types of profiles (two turbine blade profiles and one compressor blade profile at the bottom) with their wedge points for the leading edge and their trailing edge point for the trailing edge. Two steps of the edge point iteration process $\mathbf{p}_0 \to \mathbf{p}_1 \to \dots$ where a Bézier curve is generated by the center of four pairs of auxiliary points, giving a new approximation of the edge point.

2. The optical measurement process generates some remarkable data noise in regions of high curvature, e.g. in the trailing and leading edge. This leads to low data quality in the wedge point regions.
3. The consistency of the wedge points for all profiles is considerably more important than the exact detection of the wedge points for one profile.

As a consequence, we assume symmetric wedge points (in relation to the edge points) and approximate the edge parts of the profiles with spheres. More precisely, we start with the edge point, consider one neighboring curve point on each side of the profile and generate a sphere through these points, where we force the center of the sphere to be located in the plane spanned by the points. Then, we increase the number of (symmetric) neighboring profile points by a adding neighboring points on both ends and create fitting spheres (using the method described in [16]) with centers located in the plane of regression defined by them. This process of adding points is stopped when the approximation error starts to increase linearly with the number of points, i.e., when the profile segment starts to deviate substantially from a spherical curve. At this point, the sphere detaches itself from the edge part of the profile. This point defines the wedge points. In order to increase the consistency of the wedge points over all profiles we use standard data smoothing techniques. See Fig. 9 for some results.

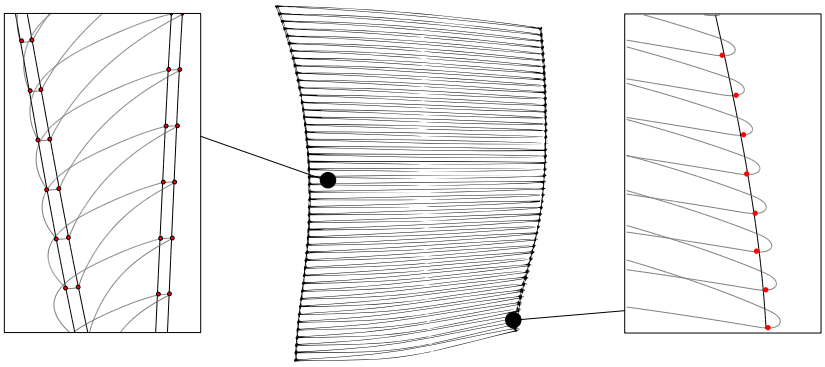

Fig. 9. Wedge points for all profiles. The plot on the right-hand side visualizes the smoothing process.

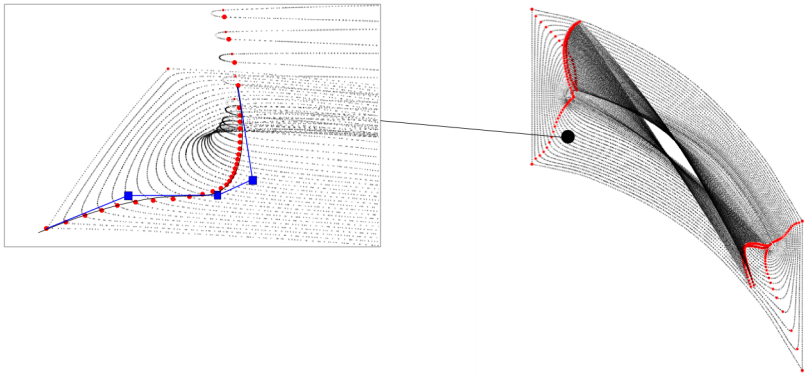

Fig. 10. Based on the tangential and a normal direction of the airfoil wedge curves and the bisector of the endwall surface at the corners, an auxiliary Bézier curve of degree four is defined. The slice points with minimal distances to this curve defines the wedge points for the base part.

4.2 Base Part

There exists no precise definition of edge points or wedge points for the slices of the base part. Therefore we extend the wedge points of the airfoil part smoothly into the base part, see Fig. 10.

5 Curve Fitting

In order to generate a volumetric tensor-product B-spline model, we have to represent the closed piecewise linear slices $\mathcal{C}_\omega$ by closed B-spline curves $\mathbf{c}_\omega(u)$

with identical degree and knot vector. We first generate initial B-spline curves and optimize them using an iterative fitting process.

5.1 Initial B-spline Curves

For each slice $\mathcal{C} \in \mathcal{C}_\omega$, the initial B-spline curve $\mathbf{c}(u)$,

$$\mathbf{c}(u) = \sum_{i=0}^{n} N_{i,d}(u)\, \mathbf{d}_i, \tag{4}$$

with B-spline basis functions $N_{i,d}$ of degree d with respect to a periodic knot sequence $\mathcal{U}$ and control points $\mathbf{d}_i \in \mathbb{R}^3$, is generated in four steps:

1. We develop the piecewise linear slice $\mathcal{C}$ to $\mathcal{U} = [0, 1]$ and relate the four wedge points to their parameters on $\mathcal{U}$, called wedge knots.
2. The remaining knots are distributed in the resulting four segments of $\mathcal{U}$, as specified by the wedge knots, by taking into account two requirements: First, the knots have to be distributed uniformly on each segment. Second, the length of the knot spans are identical for the edge parts and for the side parts.
3. We insert $d - 2$ additional wedge knots to relax the built-in smoothness to C^1 because the transitions between edges and sides of a profile are designed to be C^1 only.
4. Finally, we calculate the Greville abscissas of the knot sequence $\mathcal{U}$ and use the related points on the slice $\mathcal{C}$ as the initial positions of the control points $\mathbf{d}_i$.

Considering the whole blade, all initial B-spline curves $\mathbf{c}_\omega(u)$ have to be based on identical knot sequences, in order to be able to extend them to a tensor-product B-spline volume. Therefore we perform the first three steps for only one slice (representing the average geometry of the profiles) and use the obtained knot vector for all slices.

5.2 Fitting Process

We consider a slice $\mathcal{C} \in \mathcal{C}_\omega$, which is described by the points $\mathbf{p}_j \in \mathbb{R}^3$ obtained by intersecting the initial triangulated data with one of the slicing surfaces. In addition, we have an initial B-spline curve $\mathbf{c}(u)$. The four wedge knots $\hat{u}_0, \hat{u}_1, \hat{u}_2, \hat{u}_3$ are linked to the four wedge points $\hat{\mathbf{p}}_0, \hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, \hat{\mathbf{p}}_3$.

In order to obtain a better approximation, we minimize the objective function

$$\sum_k \|\mathbf{p}_k - \mathbf{c}(u_k)\|^2 + \sum_{j=1}^{4} \|\hat{\mathbf{p}}_j - \mathbf{c}(\hat{u}_j)\|^2 \to \min_{\mathbf{d}_i, u_k}. \tag{5}$$

This problem is solved numerically by a Gauss-Newton-type method, which can be interpreted geometrically as an evolution process, as described in [1, 13, 15]. An overview about some standard B-spline fitting techniques are given in [6, 17].

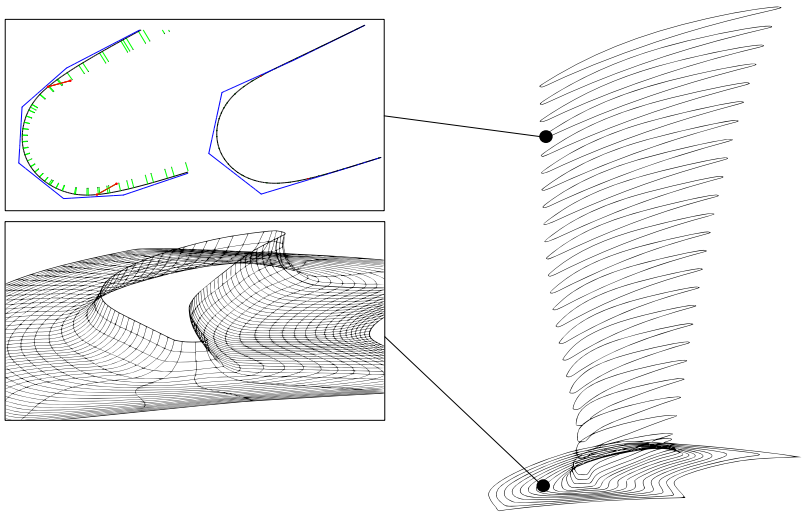

Fig. 11. Top left: One fitting step and the final B-spline curve of the evolution process. The red lines indicate the relation to the point data points with fixed parameters (wedge points, and possibly some additional points), and the green are associated with data points possessing floating parameters. Bottom left: A mesh obtained by piecewise linear interpolation of the B-splines curves. Right: The final set of B-spline curves that represent the blade surface.

In each iteration step, the data points $\mathbf{p}_j$ (with free parameters) are related to their closest points on the curve $\mathbf{c}(u_j)$ in a least-square sense. The wedge knots $\hat{u}_j$ are kept fixed. We use regularization terms (Tikhonov regularization combined with constraints on the tangential movement) in order to obtain a unique solution and a well-distributed parametric speed along the curve. Fig. 11 shows the resulting B-spline curves.

The profile curves are fitted independently, except for using identical degrees, knot sequences and wedge knots. This can lead to some unacceptable distortions of the curve parametrizations between the wedge knots by comparing neighboring curves. We suggest two approaches to overcome this problem. First, one may use more (uniformly distributed) points with fixed parameters between the wedge knots. Alternatively, one may apply a smoothing step (simultaneously across all profile curves) to the parametrization of the closest points of the data points after every iteration step. Fig. 11 shows the optimized curve parametrizations for the base part.

The used fitting framework is applicable to all manifolds of curves which are controlled by a certain system of shape parameters. Thus it is a highly valuable tool for turbine blade engineering. For example, in the aerodynamic optimization process the airfoil is designed by (lofted) streamlines represented as four C^1-connected Bézier curves of a fixed degree d, and the method can easily be adapted to this situation.

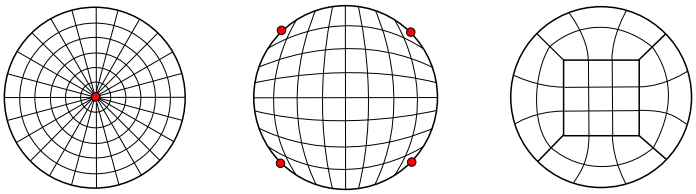

Fig. 12. Three possible parametrizations of a circular patch: A single-patch representation with a highly singular point in the center (left), a single-patch quadrilateral parametrization with four singular points on the boundary (center), and a regular five-patch representation (right).

6 Surface Generation

In this section we describe how to generate a B-spline surface for each slice. In the next section, we collect these slices to form a volume parametrization of the entire blade.

Any slice of the blade is topologically equivalent to a circle. A circular surface patch can be parametrized using different patch layouts, see Fig. 12 for three of them. In consideration of the blade proportions and the partition of the airfoil profiles by the wedge points, we use the quadrilateral representation with four singular points on the boundary which we relate to the wedge points. In the future, we plan to use the regular five-patch representation to generate a collection of tensor-product spline volumes ('multiple-patch volume') respecting the wedge point segmentation. Also, we plan to construct parametrizations that respect the interior structure of the blade, where cooling channels may be present.

The B-spline surfaces $\mathbf{s}_\omega(u, v)$ are generated by extending the B-spline curves $\mathbf{c}_\omega(u)$ in three steps, as follows:

1. The closed B-spline curve $\mathbf{c}(u)$ is subdivided at the wedge points by inserting one more knot on each wedge knot. This results in four B-spline curves of degree d and with one of the two different knot sequences $\mathcal{U}$ and $\mathcal{V}$.
2. The control points of the four curves are extended to the inner by a bilinearly blended discrete Coons patch $\mathbf{d}_{i,j}$, see [5].
3. For the base part, we move the inner control points smoothly in direction to the turbine axis to ensure a well-behaved (non-intersecting) parametrization there.

Finally, for each slice, we obtain a tensor-product B-spline surface

$$\mathbf{s}(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} N_{i,d_u}(u)\, N_{j,d_v}(v)\, \mathbf{d}_{i,j}, \tag{6}$$

with degrees $d_u = d_v = d$, knot sequences $\mathcal{U}, \mathcal{V}$ and control points $\mathbf{d}_{i,j}$. Fig. 13 shows several of the generated B-splines surfaces $\mathbf{s}_\omega(u, v)$.

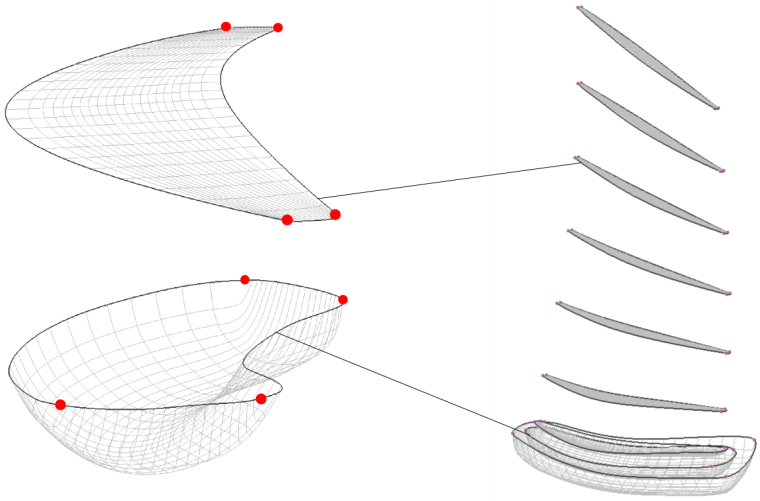

Fig. 13. Several B-spline surfaces representing the slices of the blade are shown. Two of them (left) are shown in detail. The highlighted surface points are the singularities of the patches, which correspond to the profile's wedge points.

7 Volume Generation

All B-spline surfaces $\mathbf{s}_\omega(u, v)$ have identical degrees and knot sequences. Thus we can generate a tensor-product B-spline volume $\mathbf{v}(u, v, w)$,

$$\mathbf{v}(u, v, w) = \sum_{i=0}^{n} \sum_{j=0}^{m} \sum_{k=0}^{l} N_{i,d_u}(u)\ N_{j,d_v}(v)\ N_{k,d_w}(w)\ \mathbf{d}_{i,j,k}, \tag{7}$$

by interpolating the surfaces in three steps:

1. The user defines a degree d_w.
2. A knot sequence $\mathcal{W}$ is defined by averaging the chord length parametrization of the surface points $\mathbf{s}_\omega(0.5, 0.5)$.
3. The control points $\mathbf{d}_{i,j,k}$ are generated by interpolating the control points of all surfaces with B-spline curves of degree d_w and knot sequence $\mathcal{W}$.

In order to cover the entire bade geometry, we add an appropriate trivariate B-spline block to the base part of the blade. Fig. 14 shows the final tensor-product B-spline volume.

8 Conclusions

The paper proposed a framework to model a single trivariate B-spline of a turbine blade from input triangle meshes of the blade. In order to be compatible with

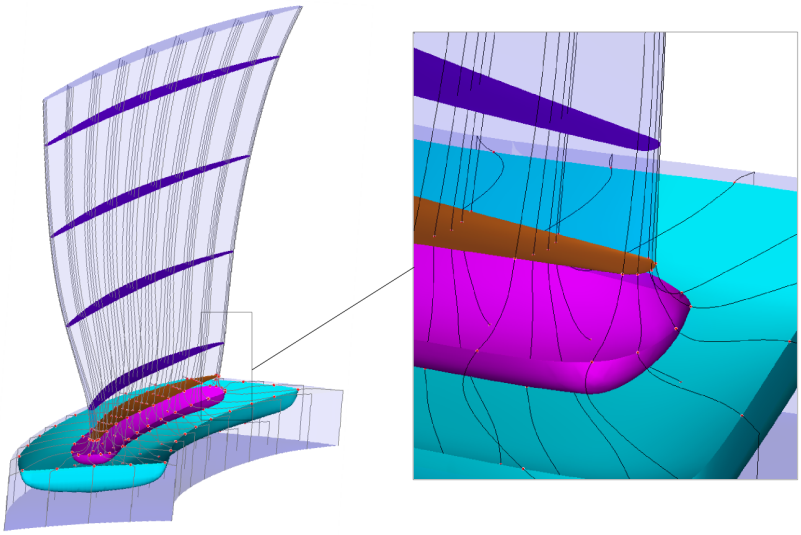

Fig. 14. The final trivariate B-spline model. The figure shows several parametric curves and surfaces.

the industrial process for blade design, we considered all standard blade features, such as wedge points, fillet curve and the natural flow of the blade at the fillet, when generating the parametrization.

The surface of the blade is represented by one surface patch with a smooth and well-behaved bivariate B-spline parametrization between the endwall boundary and the tip. It is ready for the (standard) commercial CAD and CAE applications. The volume is represented by a single trivariate B-spline parametrization and therefore usable both for isogeometric simulations and for a fully automatic structured mesh generation with standard finite elements.

To increase the quality of the model in the future, we want to investigate in the quality of the inner parametrization of the base part and in the generation of a multi-patch volumetric representation, respecting the blade features. Furthermore, local refinement techniques should be useful to avoid the restrictive tensor-product property of the B-splines geometries which may lead to distortions in the blade parametrization.

The framework is implemented in *Common LISP* and generates a volumetric blade model in a fully-automatic process using the user-defined degrees of freedom for the resulting model. For a triangle mesh with around 300.000 data points as input, the total computation time is around 20min, based on an Intel Core 2 duo processor with 3.0GHz and 8GByte memory. Nevertheless all steps of the process can be used separately in an object-oriented framework for blade modeling.

References

1. M. Aigner, B. Jüttler: Approximation Flows in Shape Manifolds. In P. Chenin, T. Lyche, L.L. Schumaker (eds.), Curve and Surface Design: Avignon 2006, Nashboro Press, 2007, 1–10.
2. M. Aigner, B. Jüttler, E. Pilgerstorfer, B. Simeon, A.-V. Vuong: Swept Volume Parameterization for Isogeometric Analysis. In E.R. Hancock, R.R. Martin (eds.): Mathematics of Surfaces XIII, Lecture Notes in Computer Science **5654** (2009), Springer, 19–44.
3. C. L. Bajaj, C. M. Hoffmann, R. E. Lynch, J. E. H. Hopcroft: Tracing surface intersections, Computer Aided Geometric Design, Volume **5** (1988), 285–307.
4. W. Boehm, H. Prautzsch: Numerical Methods. A. K. Peters, Wellesley, 1993.
5. G. Farin, D. Hansford: Discrete Coons patches, Computer Aided Geometric Design, Volume **16** (1999), 691–700.
6. M. S. Floater: Meshless parameterization and B-spline surface approximation, The Mathematics of Surfaces IX, R. Cipolla and R. Martin (eds.), Springer-Verlag (2000), 1–18.
7. R. Goldman: Curvature Formulas for Implicit Curves and Surfaces, Computer Aided Geometric Design, Volume **22** (2005), 632–658.
8. D. G. Gregory-Smith, G. Ingramnd, P. Jayaraman, N. W. Harvey, M. G. Rose: Non-axisymmetric turbine end wall profiling. Proceedings of the Institution of Mechanical Engineers, Part A: Journal of Power and Energy, **215** (2001), 721–734.
9. J. Hoschek, D. Lasser: Fundamentals of Computer Aided Geometric Design. A. K. Peters, Wellesley, Mass. (1993).
10. T. J. R. Hughes, J. A. Cottrell, Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Computer methods in applied mechanics and engineering **194** (2005), 4135-4195.
11. B. Jüttler: Least-squares fitting of algebraic spline curves via normal vector estimation. In R. Cipolla, R. Martin (eds.), The Mathematics of Surfaces IX, Springer 2000, 263–280.
12. B. Jüttler, A. Felis: Least-squares fitting of algebraic spline surfaces, Advances in Computational Mathematics **17** (2002), 135–152.
13. Y. Liu, W. Wang: A Revisit to Least Squares Orthogonal Distance Fitting of Parametric Curves and Surfaces. In F. Chen, B. Jüttler (eds.): Advances in Geometric Modeling and Processing, Lecture Notes in Computer Science **4975** (2008), Springer, 384–397.
14. T. Martin, E. Cohen, R. M. Kirby: Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Computer Aided Geometric Design, Volume **26** (2009), 648–664.
15. X. Song, M. Aigner, F. Chen, B. Jüttler: Circular spline fitting using an evolution process. J. of Computational and Applied Mathematics **231** (2009), 423–433.
16. R. Taubin: Estimation of Planar Curves, Surfaces, and Nonplanar Space Curves Defined by Implicit Equations with Applications to Edge and Range Image Segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence **13** (1991), 1115–1138.
17. W. Wang, H. Pottmann, and Y. Liu: Fitting B-spline curves to point clouds by squared distance minimization. ACM Trans. Graphics **25** (2006), 214–238.
18. G. Xu, C. Bajaj: Regularization of B-spline objects. Computer Aided Geometric Design, Volume **28** (2011), 38–49.