SPLINE TECHNIQUES FOR GENERATING AIRPLANE WINGS WITH PRACTICAL APPLICATIONS

Karl-Heinz BRAKHAGE
RWTH Aachen, Germany

ABSTRACT: In the present paper we describe the CAGD part of an effort that aimed at the unification of the whole geometric preprocessing that preceded the wind tunnel readings with a realistic air-plane wing model in a recent research project. This preprocessing includes the automated generation of the CAD models which were used for the manufacturing of the multi-parted wing-fuselage configuration and the generation of the numerical grids for the corresponding numerical simulations. Due to the constraints of the project it was decided to employ only exact, watertight, untrimmed B-Spline representations. From this process we describe the methods and algorithms for automated generation of multi-parted airplane wings from one or more cross-sections given by point clouds and the top view of the wing. A rounded wing tip an several types of winglets can be added. Furthermore we can compute and add fuselages to achieve realistic results near the root of the airfoil wing.

Keywords: B-splines, CAGD, approximation, fairing.

1 INTRODUCTION

In the Collaborative Research Center SFB 401, "Modulation of Flow and Fluid-Structure Interaction at Airplane Wings", the aerodynamics of high lift and cruise configurations and the interaction of structural dynamics and aerodynamics are presently being investigated. In the subproject "High Reynolds Number Aero-Structural Dynamics" stationary and unsteady wind tunnel readings with an elastic model have been carried out. The experiments were done in the European Transonic Wind-Tunnel (ETW) in December 2006. The wing corresponds to a cruise configuration of scale 1:28, whose supercritical cross-section (BAC 3-11) is described in two AGARD reports ([9]). The geometry of the BAC 3-11 aerofoil cross-section was numerically defined and the design ordinates were provided. The tolerances on the profile were given. The airfoil is modeled as a three parted back-swept wing with a rounded tip. To achieve realistic results a half-body is placed between the airfoil wing and the wind tunnel wall. For a future transfer project we are currently working on airfoils with winglets. The developed tools are based on B-spline representations. Approximation and fairing methods have to fulfill several constraints. These depend upon the manufacturing and the phenomena of adaptive flow solvers for the Navier-Stokes equations. It turned out that especially for numerical simulation fairing of the geometry is very important . We use a finite volume method as flow solver. Adaptation and error estimation is based on a multi-scale analysis.

In this article we describe some details on methods and algorithms for the automated generation of multi-parted airplane wings from one or more cross-sections given by point clouds and the top view of the wing with the following properties. The relative thickness of the wing (thickness/chord) can be varied from section to section. A rounded tip with design parameters and GC1-continuity at the crossing to the wing is automatically computed. Additionally several types of winglets can be added at the tip. They are de-

termined by a few user-defined parameters. Constrained approximation and fairing lead to a very smooth geometry especially suited for wind tunnel experiments and numerical simulation. A mounting unit with GC1 fillets to the wing and a simplified half of a fuselage are computed, too. The geometries of those can be modified by changing only a few significant parameters. Especially the fuselage part leads to more realistic results near the airfoil root. Overall, we had success in the effort to unify the whole geometric preprocessing related with this project. The aim was to produce geometry representations that are well suited for both the manufacturing process and the grid generation already in the modeling stage. The basic data exchange between the modeling, grid generation and manufacturing software was carried out by IGES files. Concretely the milling machine employed hyperCAD/hyperMill from OpenMind, the inner technical constructions were planed with CATIA, for the visualization we used Rhino and for the grid generation an in-house code has been developed that is part of the QUADFLOW project (see [3] for more details). A necuron (rigid foamed plastic) and a steel model for the ETW have already been manufactured. In December 2006 the wind tunnel readings have been carried out at the ETW in Cologne. The data is still under exploration. It can be accessed following the URL http://www.lufmech.rwth-aachen.de/ and the link "HIRENASD" on that web-page. The remainder of this article is organized as follows. To get an overview of the desired features of our algorithms we summarize the main properties of the given data and of the model to be achieved. Furthermore we give a brief outline of the construction process. Details on those steps that exhaustively use B-spline properties and algorithms are given in section 3 and 4. After that we describe an extension of the wing model with a winglet that is planned for future experiments. Our techniques for planar and volume meshing are based on the generation of curvature dependent offset-curves and -surfaces and B-splines, too (see [6] for details).

2 MODEL DESCRIPTION

First we have to clarify that in our context here there are two different meanings of the term chord length. In aviation the chord or chord length is the wing depth. In CAGD the chord length parameterization is an approximated arc-length parameterization. The concept of building a chord length knot spacing is motivated by the following idea. If a curve follows very closely to the data polygon of its interpolated points, then the length of the curve segment between two adjacent data points is very close to the length of the chord of these two data points and the length of the interpolating curve would be close to the total length of the data polygon. Thus, if we build the knot vector according to chord lengths, the parameters will be an approximation of the arc-length parameterization.

Figure 1: Manufactured model mounted ETW.

In Figure 1 the final manufactured model is shown. The picture shows it mounted in the European Transonic Wind-tunnel (ETW) in cologne. We now give a short overview of the given data and the different steps resulting in the

model shown in Figure 1.
The main wing for the cruise configuration was numerically described by the ordinates of 87 points (see [9] for more details). These were transformed to chord (wing depth) one in a first step. The cross-sections have to fulfill the following conditions afterwards: Start and end point is $(1,0)$. The leading edge is crossed vertically at $(0,0)$. Optionally the curve has a given curvature at the nose. The relative thickness *rt* is 11%. The exact definition at the fuselage will be described later. The tolerance due to chord length 1 is about $1.7\,10^{-4}$. From this information we compute smooth B-splines as reference for the cross-sections. All these computations are done in 2d space. The result is shown in Figure 2. The

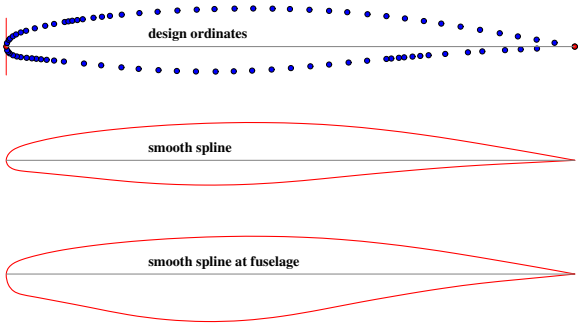

Figure 2: Design ordinates, spline ($rt = 11\%$) and spline at fuselage ($rt = 15\%$).

next step is to describe the top view of the multi-parted back-swept wing. This can be done with an arbitrary 2d CAD program. We use WinCAG ([2], [4]). The only information we need from this step is the front position of the cross-sections (A_i), their depth (l_i) and the relative position of R with respect to A_n ($= A_4$ here, compare Figure 3). R determines the shape of the wing tip.
The factors thickness/chord for the different cross-sections can be defined in the 3d module. For our wing these factors are all equal to 11%. At the fuselage the profile is treated in a different way. The enlargement of the relative thickness with respect to the previous section is only done in the lower part of the profile (see bottom plot of Figure 2). Between the wing and the blend to the

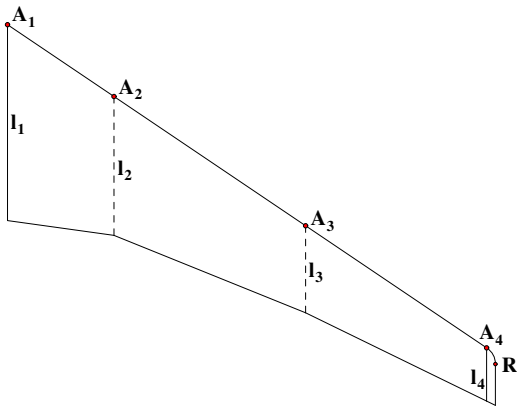

Figure 3: Top view of the multi-parted back-swept wing

mounting unit a cylindrical continuation can be added (see Figure 4 and Figure 5). The mount-

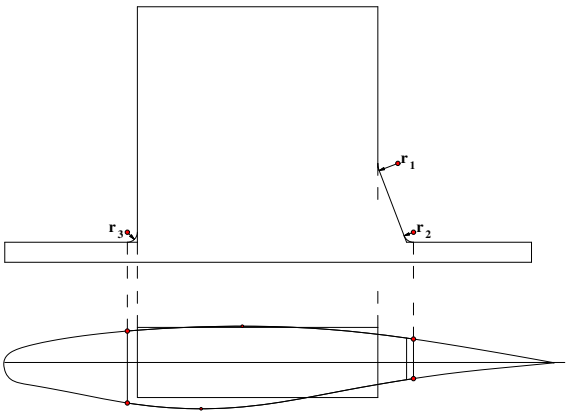

Figure 4: Top and front view of mounting unit with continuation

ing unit is given by top and front view and some rounding values. From the fillet only the top view is given. To avoid gaps, the fillet is not computed as a trimmed surface. For this reason the B-spline representing the cross-section at the fuselage has to be split up into five parts. This is done by knot insertion. The fillet and the mounting unit is computed as one block. Figure 5 shows the final result near the root. Hereby the main part of the wing is defined by a variable number of cross-sections which are connected by ruled surfaces. The plot shows about twice as many isolines as control points in each direction. Moreover the continuation to pass the fuselage and the blend to

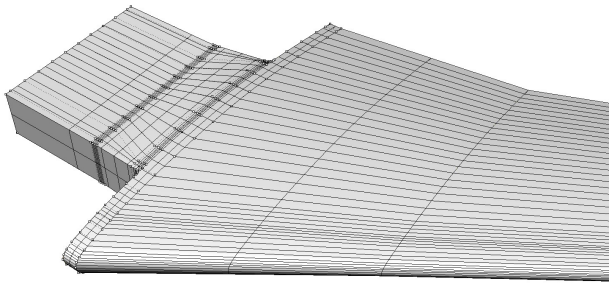

Figure 5: View of the model near to the root with mounting unit.

the mounting block can clearly be seen. In the next sections we will give more details on these computations. The sensors and cables have to be placed inside the wing. The necessary thickness of the aerofoil is roughly known from stress and eigenfrequencies computations (FE shell model considering webs) and is of variable size. Therefore a variable inner offset surface of the wing was computed. All detail constructions for the inner equipment have to remain inside this surface. They were done with the commercial software CATIA.

The flow results near the root of the airfoil are not very realistic if it is directly mounted on the wind tunnel wall. For this reason we have place the simplified half of a fuselage between the wing and the wind tunnel wall. The model is shown in Figure 6. Figure 1 shows the whole constellation

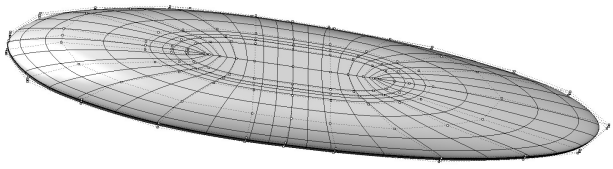

Figure 6: Simplified fuselage.

mounted in the ETW. Notice that there is no direct contact between the airfoil and the fuselage. A labyrinth-sealing was constructed to avoid a flow into the inner part. To make that construction as easy as possible the intersection area was constructed in such a way that it is flat. For the visualization of this feature we have extended (*closed*) the surface shown in Figure 6. Again we have about twice as many isolines as control points in each direction.

3 B-SPLINES

3.1 Main Notations

Throughout this paper we write B-spline curves in the form

$$\mathbf{x}(t) = \sum_{i=0}^{N} \mathbf{p}_i N_{i,T}^p(t) \tag{1}$$

where $N_{i,T}^p(t)$ is the i-th normalized B-spline function of order p (degree $p-1$) corresponding to the generally non-uniform knot vector $T = (t_0, t_1, \ldots, t_{N+p})$. The $\mathbf{p}_i$ are called **control** or **de Boor points**. They form (in ascending order) the **control polygon**. We usually assume that T is clamped, i.e., $t_0 = \ldots = t_{p-1}$ and $t_{N+1} = \ldots = t_{N+p}$. For the sake of simplicity we write N_i^p instead of $N_{i,T}^p$ since it becomes clear from the name of the function argument what the knot vector is. Mostly we have $p = 4$ and in this case we even write N_i instead of $N_{i,T}^p$. Surfaces are represented by B-spline tensor products of the form

$$\sum_{i=0}^{N} \sum_{j=0}^{M} \mathbf{p}_{ij} N_i^p(u) N_j^q(v). \tag{2}$$

The extension to volumes is straight foreword.

3.2 Basic properties

We will now summarize those properties of B-splines which we needed in this paper. We start with B-spline functions and will then report on curves, surfaces and volumes.

A B-spline function of order p is piecewise a polynomial of degree $p-1$. It can recursively be computed by

$$\begin{aligned} p > 1 \quad : \quad N_i^p(t) &= \frac{t-t_i}{t_{i+p-1}-t_i} N_i^{p-1}(t) + \\ &\quad \frac{t_{i+p}-t}{t_{i+p}-t_{i+1}} N_{i+1}^{p-1}(t) \\ p = 1 \quad : \quad N_i^1(t) &= \begin{cases} 1 & t_i \le t < t_{i+1} \\ 0 & \text{elsewhere} \end{cases} \end{aligned} \tag{3}$$

for $i = 0, 1, ..., N$. It is easy to verify that $N_i^p(t)$ has local support $[t_i, t_{i+p})$ ($N_i^p(t) = 0$ for $t \notin$

$[t_i, t_{i+p})$) and all B-Spline functions are positive. Another important property is

Theorem 1 *The B-spline functions of every order p are partition of unity on $[t_{p-1}, t_{N+1})$, that is*

$$\sum_i N_i^p(t) = 1. \tag{4}$$

For the derivatives of the normalized B-spline functions we have the following

Theorem 2

$$N_i^{p\prime}(t) = (p-1)\left\{\frac{N_i^{p-1}(t)}{t_{i+p-1}-t_i} - \frac{N_{i+1}^{p-1}(t)}{t_{i+p}-t_{i+1}}\right\} \tag{5}$$

The continuity of the B-spline functions is characterized by the following

Theorem 3 *B-spline functions of order p are C^{p-l-1}-continuous at a knot of multiplicity l.*

Remark: If we have only single knots in the interior, the B-spline functions are C^{p-2}-continuous on $[t_{p-1}, t_{N+1}]$.
Next we state some important properties of B-spline curves, surfaces and volumes. If we use the recursion formula of (3) for a B-spline curve and rearrange it, we get the following

Theorem 4

$$\begin{aligned}
\mathbf{x}(t) &= \sum_i \mathbf{p}_i N_i^p(t) \\
&= \sum_i \mathbf{p}_i \left(\frac{t-t_i}{t_{i+p-1}-t_i} N_i^{p-1}(t) \right. \\
&\quad \left. + \frac{t_{i+p}-t}{t_{i+p}-t_{i+1}} N_{i+1}^{p-1}(t)\right) \\
&= \sum_i \left(\frac{t-t_i}{t_{i+p-1}-t_i}\mathbf{p}_i + \frac{t_{i+p-1}-t}{t_{i+p-1}-t_i}\mathbf{p}_{i-1}\right) \\
&\quad N_i^{p-1}(t) \qquad (6) \\
&= \sum_i \mathbf{p}_i^1 N_i^{p-1}(t) \\
&= \sum_i \mathbf{p}_i^2 N_i^{p-2}(t)
\end{aligned}$$

Repeating the above process and noticing that the weights of $\mathbf{p}_i^l$ and $\mathbf{p}_{i-1}^l$ add to one (Theorem 1) we get the efficient and stable algorithm of de Boor for the point wise computation of $\mathbf{x}(t)$.

The extension to surfaces and volumes is straight forward if we notice that for instance

$$\begin{aligned}
\mathbf{x}(u,v) &= \sum_{i,j=0}^{N,M} \mathbf{p}_{ij} N_i^p(u) N_j^q(v) \\
&= \sum_{j=0}^{N} \left(\sum_{i=0}^{M} \mathbf{p}_{ij} N_i^p(u)\right) N_j^q(v) \qquad (7) \\
&= \sum_{j=0}^{N} \tilde{\mathbf{p}}_j(u) N_j^q(v).
\end{aligned}$$

The same way we can handle derivatives. Using Theorem 2 we get

Theorem 5

$$\begin{aligned}
(\mathbf{x}(t))' &= \left(\sum_i \mathbf{p}_i N_i^p(t)\right)' \\
&= \sum_i \mathbf{p}_i \left(N_i^p(t)\right)' \\
&= \sum_i \mathbf{p}_i (p-1) \\
&\quad \left\{\frac{N_i^{p-1}(t)}{t_{i+p-1}-t_i} - \frac{N_{i+1}^{p-1}(t)}{t_{i+p}-t_{i+1}}\right\} \qquad (8) \\
&= \sum_i \frac{p-1}{t_{i+p-1}-t_i} (\mathbf{p}_i - \mathbf{p}_{i-1}) N_i^{p-1}(t) \\
&= \sum_i \mathbf{v}_i N_i^{p-1}(t), \\
(\mathbf{x}(t))'' &= \sum_i \mathbf{a}_i N_i^{p-2}(t).
\end{aligned}$$

Thus $(\mathbf{x}(t))'$ is a B-spline curve of order $p-1$. For surfaces and volumes partial differentiation reduces the order for corresponding *direction*.
For more detailed information on B-splines see [8]. A comprehensive survey on curves and surfaces can be found in [1].

3.3 Interpolation with B-Splines

Now we assume that a set of points $\mathbf{x}_i$ for $i = 0, 1, ..., N-p+2$ is given and we search for a B-spline passing through this points. For this purpose we first additionally assume that a knot-vector

$$\begin{aligned}
T = (t_0 &= \ldots = t_{p-1} < t_p < t_{p+1} < \ldots \\
&< t_{N+1} = t_{N+2} = \ldots = t_{N+p})
\end{aligned} \tag{9}$$

is given (normally we use the chord length to compute this vector) to build the B-Spline curve

$$\mathbf{x}(t) = \sum_{i=0}^{N} \mathbf{p}_i^p N_i(t) \tag{10}$$

If we want to interpolate at the knots, we get the interpolating conditions

$$\mathbf{x}(t_{i+p-1}) = \mathbf{x}_i \ , \ i = 0,1,...,N-p+2. \quad (11)$$

These are $N-p+3$ conditions for the $N+1$ control points $\mathbf{p}_i$. Thus for $p=4$ we can formulate 2 further conditions. The following four condition sets are of practical use and therefore implemented in our system:

a) Vanishing curvature at the beginning and the end of the curve.

b) Given derivatives at the beginning and the end of the curve.

c) Not a knot condition. This means the curve is C^3 at the second and last but one interpolation point.

d) We construct a periodic C^2-continuous curve (only reasonable if $\mathbf{x}_0 = \mathbf{x}_{N-2}$).

Now we place all the control points $\mathbf{p}_i$ in a vector of $2D$- or $3D$-vectors $\mathbf{p}$ and the interpolation points $\mathbf{x}_i$ in $\mathbf{x}$. Due to the local support of the normalized B-spline functions the condition $\mathbf{x}(t_{i+3}) = \mathbf{x}_i$ results in a linear equation of the form

$$\alpha_i \mathbf{p}_i + \beta_i \mathbf{p}_{i+1} + \gamma_i \mathbf{p}_{i+2} = \mathbf{x}_i \quad (12)$$

with $\beta_0 = \gamma_0 = 0$ and $\alpha_{N-2} = \beta_{N-2} = 0$. Vanishing curvature yields two equations of the form:

$$\begin{aligned} a_0 \mathbf{p}_0 + b_0 \mathbf{p}_1 + c_0 \mathbf{p}_2 &= \mathbf{0} \\ a_N \mathbf{p}_{N-2} + b_N \mathbf{p}_{N-1} + c_N \mathbf{p}_N &= \mathbf{0} \end{aligned} \quad (13)$$

Given derivatives result in:

$$\begin{aligned} a_0 \mathbf{p}_0 + b_0 \mathbf{p}_1 &= \mathbf{t}_0 \\ b_N \mathbf{p}_{N-1} + c_N \mathbf{p}_N &= \mathbf{t}_N \end{aligned} \quad (14)$$

If we plug in the equations (13) (or (14) respectively) at the second and last but one position we end up with a sparse linear system

$$A\mathbf{p} = \tilde{\mathbf{x}}. \quad (15)$$

The resulting system matrix A is tridiagonal for the cases a) and b). The control points can easily be computed from this system in $O(N)$ time by Gauss-elimination and back substitution afterwards. The not a knot condition results in two equation with entries for the first five and the last five control points. Two pre-elimination steps for each of them yield a tridiagonal system again. Only for the periodic case d) we have a fill-in in the last column and row during the Gauss-elimination steps. But nevertheless the computation time is $O(N)$ again.

For surfaces we get a matrix A_u for the u-direction and A_v for the v-direction. Now the collections of control points $\mathbf{P}$ and interpolation points $\mathbf{X}$ are matrices of vectors ($2D$ or $3D$). Again we have add conditions at the boundary curves. Notice that at the boundary corners we have 4 undetermined conditions. This can be resolved by choosing a *twist vector* (the partial derivative $\mathbf{x}_{uv}$ at every corner). In standard literature this is solved the following way: Let $vec(S)$ be the vector obtained by catenating the columns of a matrix S; first column of S first, then second and so on. In MatLab notation this writes as $vec(S) = S(:)$. This concept can be generalized to matrices of vectors and the 3D case, but there is no direct MatLab notation. By $\otimes$ we denote the standard tensor product (Kronecker product, see [10] for more details on tensor products). In 2D the interpolation conditions result in the following equivalent equations:

$$(A_v \otimes A_u)\,\mathbf{P}(:) = \tilde{\mathbf{X}}(:). \quad (16)$$

To avoid tensor product matrices we write the resulting interpolation problem in the form

$$A_u \mathbf{P} A_v^T = \tilde{\mathbf{X}}, \quad (17)$$

which is equivalent to (16). Now a Gauss-elimination step regarding A_u not only works on one column vector $\tilde{\mathbf{x}}$, but on all columns of $\tilde{\mathbf{X}}$. Analogous the elimination regarding A_v works on all rows of $\tilde{\mathbf{X}}$. The same is true for back substitution. If we use standard indices i and j for the elements (3D vectors) of $\tilde{\mathbf{X}}$ we also say A_u acts on all

j and A_v on all i. With the L-U-decompositions $A_u = L_u U_u$ and $A_v = L_v U_v$ this can be written as

$$A_u \mathbf{P} A_v^T = L_u U_u \mathbf{P} U_v^T L_v^T = \tilde{\mathbf{X}}. \tag{18}$$

We need the 3D case for the volume grids around our models. It is obtained from the tensor product nature analogous to the 2D case above, but we cannot use the *standard* matrix notations as in (18). We can only write

$$(A_w \otimes (A_v \otimes A_u))\,\mathbf{P}(:) = \tilde{\mathbf{X}}(:). \tag{19}$$

Note that A_u, A_v and A_w are still tridiagonal matrices. Now there are the indices i, j and k for the elements of $\tilde{\mathbf{X}}$ and A_u acts on all j,k, A_v on all i,k and A_w on all i,j of $\tilde{\mathbf{X}}$. Thus our algorithms make use of the sparsity of the system matrices in the same way as above.

3.4 Approximation with B-Splines

If the data comes from measurements and thus is not *exact* interpolation leads to non smooth curves. For this reason we need methods for approximation and fairing. Figure 2 shows an example obtained with our methods. Since there are some constraints like fixed points, tangents and curvature usual CAD Systems can not be used for our approximation task. Thus we have to develop special algorithms for this stage of our modeling process. If we have more data points and constraints than control points we can no longer fulfill the equations like (15), (18) or (19). Instead we solve the corresponding (linear) least squares problem. If the problem is of tensor product structure we have similar equations as above. But know the systems are of bandwidth four and over-determined. Therefore we replace the L-U-decomposition by the Q-R-decomposition. For (15) this is straight foreword

$$\begin{aligned} \|A\mathbf{p} - \tilde{\mathbf{x}}\|_2 &= \|QR\mathbf{p} - \tilde{\mathbf{x}}\|_2 \\ &= \|R\mathbf{p} - Q^T\tilde{\mathbf{x}}\|_2. \end{aligned} \tag{20}$$

The upper part (N+1 equations) can be solved by back substitution. In the lower part we have only zeros in R and the corresponding right hand side gives us the residual of the overall problem.

For surfaces and volumes the situation is a little bit more complicated. We still want to use the structure of (18) instead of (16) for our Q-R-decomposition. At a first glance this is a slight modification of the standard least squares approximation that rapidly brings down computational time and space. In the surface case this can still be written in standard matrix notation as follows:

$$\begin{aligned} \left\|A_u \mathbf{P} A_v^T - \tilde{\mathbf{X}}\right\|_2 &= \\ \left\|Q_u R_u \mathbf{P} (Q_v R_v)^T - \tilde{\mathbf{X}}\right\|_2 &= \\ \left\|R_u \mathbf{P} R_v^T - Q_u^T \tilde{\mathbf{X}} Q_v\right\|_2 &\rightarrow min \end{aligned} \tag{21}$$

$\mathbf{P}$ is computed by backward substitution from left with R_u and right with R_v^T on the upper left part of $Q_u^T \tilde{\mathbf{X}} Q_v$. Thus we have minimized the 2-norm instead of the Frobenius-norm, which corresponds to the standard least squares approximation.
The following theorem is known

Theorem 6 *Let $A \in \mathbb{R}^{m\times n}$ with $m \geq n$. Then*

$$\|A\|_2 \leq \|A\|_F \leq \sqrt{n}\|A\|_2. \tag{22}$$

But we can proof that for matrices arising from tensor products like (21) in the above theorem equality holds between the Frobenius- and the 2-norm. Furthermore this result can be extended to the case of volumes. Thus it leads to algorithms for the standard 2-norm approximation. Their complexity is nearly proportional to the number of approximation points.
If there is no tensor product structure we switch to an iterative method. We have decided to use CGLS, a Conjugate Gradient method for linear Least Squares (also called CGNR in [11]). The complexity is governed by two (sparse) matrix-vector multiplications with A and A^T. Above we have totally avoided the normal equations for least squares problems, because they are ill conditioned (huge condition number of $A^T A$) very often. Using CGLS we still avoid to build $A^T A$ and having the huge condition number regarding precision, but we can not avoid its influence on the convergence rate. Notice that A and A^T are sparse matrices, but $A^T A$ is not. Using B-splines of order $p = 4$ surface points depend on

at most 16 control points and volumes of at most 64. These values limit the number of non zero entries in A. Therefore CGLS is an efficient method to solve the least squares problems arising from B-splines because it makes extensive use of the sparsity of the system matrix.

4 SOME SPECIAL STAGES OF THE THE CONSTRUCTION

In this chapter we will give some details on selected steps of the construction. The properties stated in the previous section will intensively be used.

4.1 The reference cross-section

In our case the cross-section was given by measurements in form of ordinates for points Thus we have to start our considerations with a (planar) cloud of $M+1$ sorted points $\mathbf{x}_j$. For the parameterization we compute the chord length (CAGD meaning) knot spacing of the corresponding curve $\mathbf{x}(t)$. It gives us an initial guess of the parameter values $\tilde{t}_j$ for $\mathbf{x}_j$. From the sequence $\tilde{t}_j$ we build the $N-p+3$ interior knots t_i in such a way that the density of the t_i accords to that of the $\tilde{t}_j$. The wanted tolerance ε (maximum distance between given points and final curve) is split into $\varepsilon = \varepsilon_1 + \varepsilon_2$, the tolerance for the approximation and that for the fairing process. The number of control points is adapted in such a way, that our curve fulfills

$$\max_j \min_t \|\mathbf{x}_j - \mathbf{x}(t)\|_2 \leq \varepsilon_1. \tag{23}$$

This is done for instance by repeatedly solving (20) until we get the smallest N that fulfills (23). Additionally we adjust the parameterization of the curve to arc length. It is much easier to fulfill the constraints on tangents and curvature in that form. For more details on such approximation problems see [7]. In a second step the fairing is done in such a way that the third derivative is close to a constant. For this process the movements of the control points is restricted by the distance ε_2 to guarantee that the overall error is limited by ε. The results can be found in Figure 2. The last step on the cross-sections is the split according to Figure 4. After that we can construct the blend to the mounting unit. The control points of the top and bottom surface can be seen in Figure 7

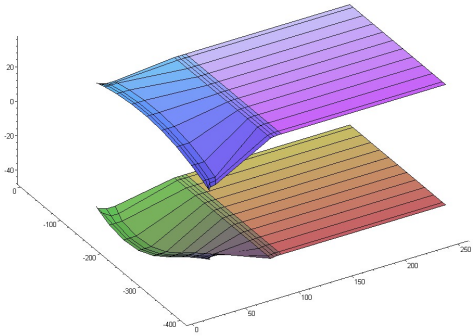

Figure 7: Control points of blend and mounting unit.

4.2 The simplified fuselage

In Figure 6 we have already seen the shape of a simplified fuselage and the control points of the B-spline surface representing it. We start the construction with a 2D sketch of the cross-sections in two orthogonal directions (see Figure 8). They are modeled as B-splines. The next step is to compute a periodical surface that passes these cross-sections. We use elliptical arcs for this purpose. Then a cylindrical part is place in the middle of the surface. This done by inserting knots, and stretching the parameterization and the control points on a straight line. Finally we take care that enough control points are planar to guarantee that the intersection with the wing is planar (compare section 2).

4.3 Winglet construction

In [5] algorithms for generating airplane wings for numerical simulation and manufacturing were presented. We have enhanced the methods given there by algorithms for winglet constructions. A final result is shown in Figure 9. The basic idea is to determine the necessary parameters in top and front view and then do all computations directly on the control points. The required modifications on the top view (compare Figure 3) are shown in Figure 10. Again WinCAG is used for these construction steps. We mark the bend-

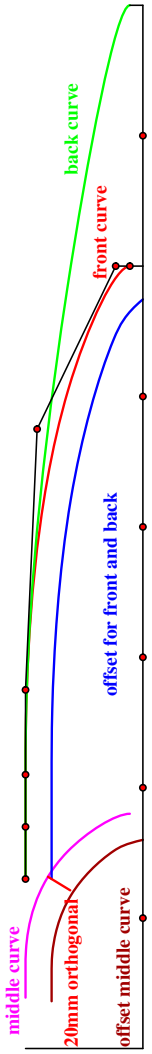

Figure 8: Sketches for the fuselage.

ing position $\mathbf{x}_0$ and add an additional dihedral angle a. From this the new positions A'_4 and A'_5 and the wing chords l'_4 and l'_5 are determined. The suggestion for R' is to use the same shortening as in the original case without the winglet. Next imagine a horizontal wing in x-direction and the (horizontal) plane of the wing chords ($z = 0$). In this plane the control points of our surfaces have (x,y)-coordinates and a certain height (positive or negative). Bending the plane at an axis in y-direction with radius r (see Figure 11) we transform the (x,y)-coordinates of the control points. Then we add their previous height perpendicular to the bended plane receiving the control points of the patches describing the winglet and its tip.

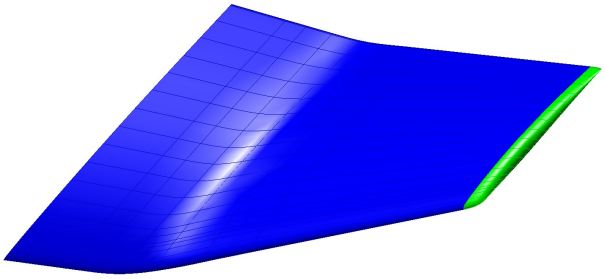

Figure 9: Winglet.

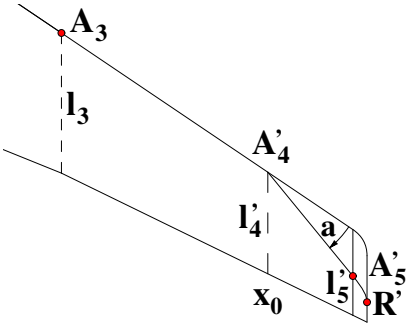

Figure 10: Winglet construction - top view.

To receive satisfactory results several knots have to be inserted for the x-direction (see Figure 11).

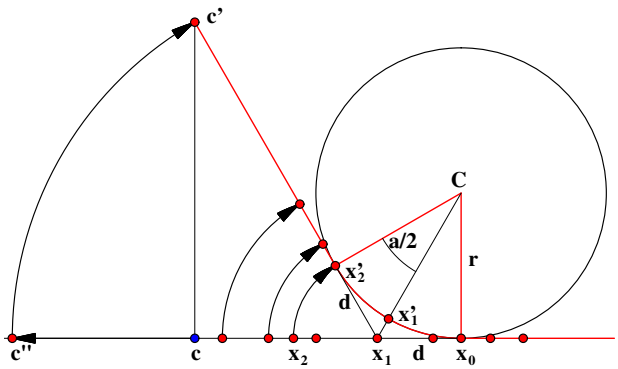

Figure 11: Winglet construction - front view.

5 CONCLUSIONS

In this paper we have described the use of B-spline techniques for the automated generation of sparse, watertight B-spline models for wind tunnel wing-fuselage configurations. Classical tools have been modified and adapted to the special requirements of this project. The resulting

geometry can be transferred without conversions or approximations between the various software which was used for the manufacturing, technical construction and grid generation by IGES files. Moreover, the parameters of the construction, like profile ordinates, bending radii, sweep angle, etc. can easily be modified since the modeling algorithms have been automated.

ACKNOWLEDGMENTS

This work has been performed with funding by the Deutsche Forschungsgemeinschaft in the Collaborative Research Center SFB401 "Flow Modulation and Fluid Structure Interaction at Airplane Wings" of the RWTH Aachen, University of Technology, Aachen, Germany.

REFERENCES

[1] Böhm, W., Farin, G. and Kahmann, J. A survey of curves and surface methods in CAGD. Computer Aided Geometric Design 1, 1–60, 1984.

[2] Brakhage, K.-H. Ein menugesteuertes, intelligentes System zur zwei- und dreidimensionalen Computergeometrie, VDI Reihe 20 CAD/CAM, Nr. 26 Edition. VDI Verlag,1990.

[3] Brakhage, K.-H., Lamby, Ph., Müller, S. et all. H-adaptive multiscale schemes for the compressible navier-stokes equations — polyhedral discretization, data compression and mesh generations. In J. Ballmann, editor, *Flow Modulation and Fluid-Structure-Interaction at Airplane Wings*, volume 84 of *Numerical Notes on Fluid Mechanics*, pages 125–204. Springer, 2003.

[4] Brakhage, K.-H. Wincag-education software for geometry. In: Proceedings of th 11^th International Conference on Engineering Computer Graphics and Descriptive Geometry. Guangzhou, China, August 1-5 2004.

[5] Brakhage, K.-H. and Lamby, Ph. Generating airplane wings for numerical simulation and manufacturing. In Soni B.K. et all, editor, 9^{th} *International Conference on Numerical Grid Generation in Computational Field Simulations*, San Jose, USA, June 11-18 2005.

[6] Brakhage, K.-H. and Lamby, Ph. Numerical Grid Generation for Solving the Navier-Stokes Equations using B-Spline Techniques. In Soni B.K. et all, editor, 9^{th} *International Conference on Numerical Grid Generation in Computational Field Simulations*, San Jose, USA, June 11-18 2005.

[7] Brakhage, K.-H. and Lamby, Ph. Application of B-Spline Techniques to the Modeling of Airplane Wings and Numerical Grid Generation. Submitted to Computer Aided Geometric Design, (IGPM preprint 2007).

[8] Farin, G., Curves and Surfaces for Computer-Aided Geometric Design. Academic Press, 4^{th} edition (September 1996).

[9] Moir, I. Measurements on a two-dimensional aerofoil with high lift devices. AGARD-AR-303 Vol. I+II, DRA, Farnborough, 1994.

[10] Pitsianis, N.P. The Kronecker Product in Approximation and Fast Transform Generation. Cornell University, 1997, (Dissertation)

[11] Saad, Y. Iterative Methods for Sparse Linear Systems, 2nd Edition. SIAM, 2003.

ABOUT THE AUTHOR

Karl-Heinz Brakhage is member of the Institute of Geometry and Numerical Mathematics at the Aachen University of Technology. His research interests are Computer Aided Geometric Design, CAx Technologies, Grid Generation, Scientific Computing, Computer Graphics, and Development of Education Software. He can be reached by e-mail: brakhage@igpm.rwth-aachen.de, by Fax: +49(241)8092317, by phone: +49(241)8096591, the postal address: Inst. of Geometry and Numerical Mathematics / RWTH Aachen / Templergraben 55 / D-52056 Aachen, Germany, or through the Web site: www.igpm.rwth-aachen.de/brakhage