Arch Computat Methods Eng (2018) 25:1059–1127
DOI 10.1007/s11831-017-9220-9

ORIGINAL PAPER

A Review of Trimming in Isogeometric Analysis: Challenges, Data Exchange and Simulation Aspects

Benjamin Marussig[1] · **Thomas J. R. Hughes**[1]

Received: 16 February 2017 / Accepted: 17 March 2017 / Published online: 2 June 2017

Abstract We review the treatment of trimmed geometries in the context of design, data exchange, and computational simulation. Such models are omnipresent in current engineering modeling and play a key role for the integration of design and analysis. The problems induced by trimming are often underestimated due to the conceptional simplicity of the procedure. In this work, several challenges and pitfalls are described.

1 Introduction

Trimming is much more complicated than most people think. It is one of the most fundamental procedures in Computer Aided Geometric Design (CAGD) that allows the construction of complex geometries. Unfortunately, it is also the source of one of the most serious impediments to interoperability between Computer Aided Design (CAD) systems and downstream applications like numerical simulation [268]. This work aims to increase awareness of this issue by providing a broad overview of trimmed geometries, addressing design, data exchange, and analysis aspects.

Once upon a time, the original vision of CAD was the holistic treatment of the engineering design process [247]. However, it has emerged as an autonomous discipline which seeks to optimize the modeling and visualization of geometric objects. On the other hand, computational analysis has focused on the problem-solving part of engineering. Thus, the main attention has been drawn to the development of mathematical models governing physical phenomena as well as the reliability and efficiency of their numerical treatment. Still, design models are usually the starting point of the analysis process in order to define the domain of interest. In current engineering design, however, they are subsequently approximated by finite element meshes for computation. Since this is a fundamental step in conventional simulations, there is a substantial body of literature on meshing, see e.g., [28, 94, 98, 281, 322] and the references cited therein. Finite element analysis (FEA) was a widely used commercially available procedure in engineering prior to the advent of commercial CAD. Nevertheless, FEA finds itself separated from design by its own representation of geometrical objects, which is different from CAD. The given situation has contributed to a loss of communication between these fields, both of which are essential in the process of addressing practical engineering problems.

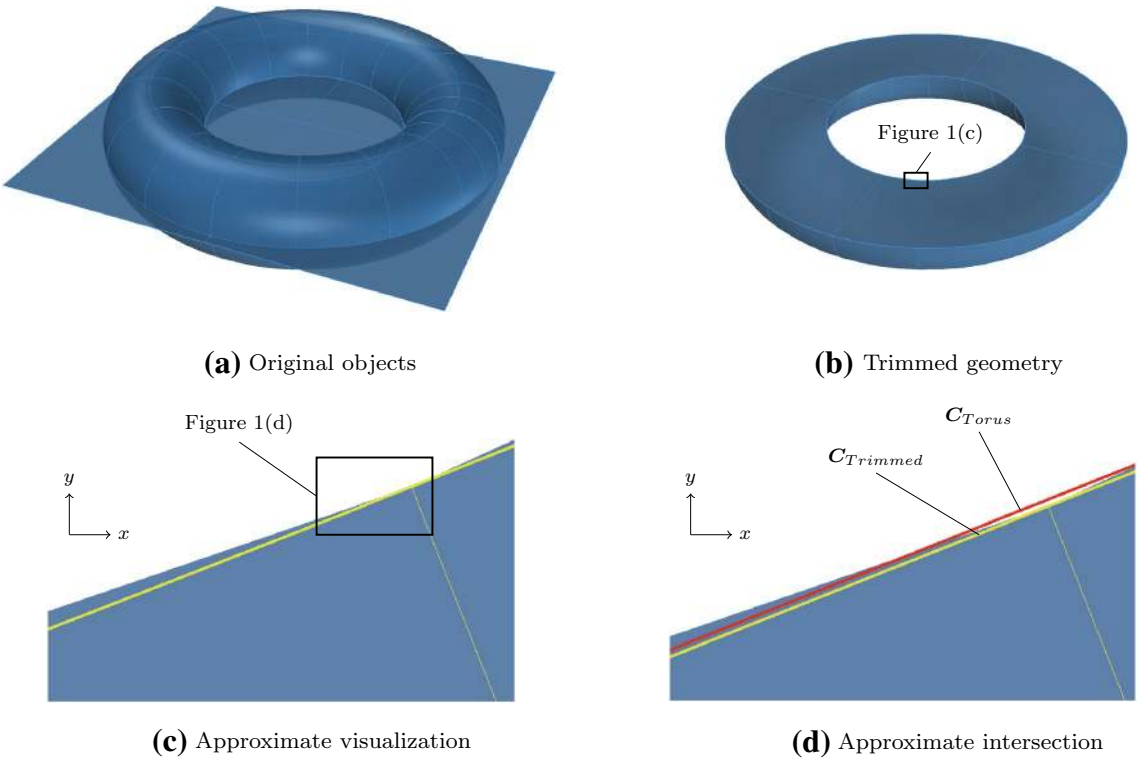
Isogeometric analysis [66, 140] provides an alternative to the conventional analysis methodology that converts CAD models for use in FEA. The key idea is to perform numerical simulations based on CAGD technologies. Besides the fact that this synthesis offers several computational benefits, such as high continuity [67, 68, 187], the long term goal of isogeometric analysis is to enhance the overall engineering product development process by closing the gap between design and analysis. An invaluable byproduct of this effort is the initiation of a dialog between these two communities which had drifted apart.

Gaining insight into each others' fields is an essential component to tackle the problem of interoperability between analysis and design representations. It is easily overlooked that each discipline has its open challenges

The original version of this article was revised due to a retrospective Open Access order.

✉ Benjamin Marussig
marussig@utexas.edu

[1] Institute for Computational Engineering and Sciences, The University of Texas at Austin, 201 East 24th St, Stop C0200, Austin, TX 78712, USA



(a) Original objects **(b)** Trimmed geometry

(c) Approximate visualization **(d)** Approximate intersection

Fig. 1 Model of a half of a torus: **a** initial non-trimmed torus and surface defined in the *xy*-plane, **b** resulting trimmed object, **c** closeup showing the deviation from the visualization mesh (*blue background*) and the computed intersection curve $\boldsymbol{C}_{Trimmed}$ (*yellow*) of the objects, and **d** closeup illustrating the difference of the original inner *circle* of the torus $\boldsymbol{C}_{Torus}$ (*red*) to the intersection curve computed. The images **c** and **d** are captured in top view. (Color figure online)

and limitations. Due to the capability of current CAD systems it may seem that design models are ideal creations, but this notion is simply not true. There are several open issues that need to be solved. Compact overviews are given in the independent compilations of Kasik et al. [151] and Piegl [228]. In these papers, robustness and interoperability issues are identified as crucial CAD problems. Although trimmed geometries are not explicitly mentioned in these papers, they play a central role in both cases.

The most common description of CAGD models is the boundary representation (B-Rep) where an object is represented by its boundary surfaces rather than a volume discretization. These surfaces are usually constructed independently from each other and often only certain regions of a surface are supposed to be part of the actual object. Trimming allows a modeler to cut away the superfluous surface areas. To be precise, the visualization of the surfaces is adapted while their parameterization and mathematical description remain unchanged. This procedure is very convenient and inevitable in many operations such as surface-to-surface intersection. However, the main problem is that trimming *cannot* practically be performed exactly within CAGD applications. Thus, the final object possesses small gaps and overlaps between its surfaces. Figure 1 illustrates some inaccuracies of a model defined by a torus intersected by a plane. Note that the discrepancy between the computed intersection $\boldsymbol{C}_{Trimmed}$ and the related exact solution $\boldsymbol{C}_{Torus}$ is scarcely visible. The imperfections of trimmed geometries are usually very well hidden from the user, but they surface as soon as a design model is applied to downstream applications. To use the words of Piegl [228]:

> While one can cheat the eye in computer graphics and animation, the milling machine is not as forgiving.

Numerical simulation of practical trimmed models is more than the analysis of a specific type of a CAGD representation. It rather addresses the core issue of the interoperability between design and analysis, namely the appropriate treatment of the deficiencies of design models. To be clear, this problem is not restricted to isogeometric analysis, but manifests itself as complications during the meshing process in the case of conventional analysis methodology. In fact, geometry repair and corrections of design models are mandatory tasks, before actual mesh generation can be applied [86, 114]. Isogeometric analysis of trimmed geometries tackles these issues directly at the source, i.e., the design model. Thus, many pitfalls that may occur in a meshing process can be circumvented [61].

It is important to note that the CAD community is also influenced by the ongoing dialog. An increasing number of researchers propose new modeling concepts that take the needs of downstream applications into account, see e.g. [61, 203, 247]. We believe that the aligned efforts of both communities are the keys to unite design and analysis, resulting in a holistic treatment of the engineering design process.

This paper intends to encourage the interaction of these fields by providing an overview of various aspects related to trimmed geometries. Section 2 begins by reviewing some basic concepts frequently used in CAGD. It is focused on non-uniform rational B-spline (NURBS) based B-Rep models since they are the most popular representation in engineering design. Based on this, Sect. 3 addresses the role of trimming in the context of design. A critical assessment of exchanging data between different software packages is provided in Sect. 4. Finally, various strategies to deal with trimmed geometries in an isogeometric analysis process are outlined in Sect. 5. Each of these three review sections closes with a brief summary of the main points and their discussion. Section 6 moves on to focus on a particular aspect, namely the stabilization of a trimmed basis. In the concluding section, the main findings are summarized and some open research questions are listed.

2 CAGD Fundamentals

B-splines and their rational counterpart NURBS provide the basis for the geometric modeling of most engineering models. This section gives a brief overview of this CAGD technology focusing on aspects which are crucial for the subsequent discussion. For further information related to spline theory the interested reader is referred to [34, 62, 83]. Detailed descriptions of efficient algorithms can be found in [230]. In the present paper, the terms B-spline and NURBS are used to refer to basis functions. The geometric objects described using these functions, i.e., curves and surfaces, may be generally denoted as patches.

2.1 Basis Functions

B-splines $B_{i,p}$ consist of piecewise polynomial segments which are connected by a certain smoothness. They are defined recursively for a fixed polynomial degree p by a strictly convex combination of B-splines of the previous degree, $p-1$, given by

$$B_{i,p}(u) = \frac{u-u_i}{u_{i+p}-u_i} B_{i,p-1}(u) + \frac{u_{i+p+1}-u}{u_{i+p+1}-u_{i+1}} B_{i+1,p-1}(u), \tag{1}$$

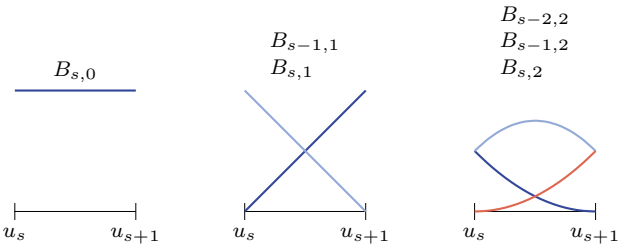
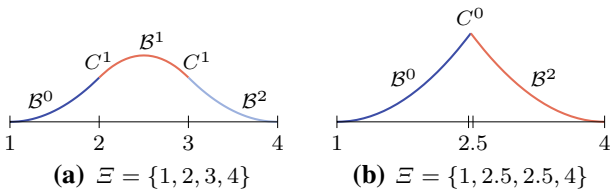


Fig. 2 Non-vanishing B-splines $B_{i,p}$ of knot span s for different degrees $p = \{0, 1, 2\}$ which are based on a knot vector with equally spaced knots



(a) $\Xi = \{1, 2, 3, 4\}$ **(b)** $\Xi = \{1, 2.5, 2.5, 4\}$

Fig. 3 Polynomial segments $\mathcal{B}^s$ of a quadratic B-spline due to different knot vectors Ξ. Note the different continuity C between the segments based on the knot multiplicity

with

$$B_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \le u < u_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The essential element for this construction is the *knot vector* Ξ characterized as a non-decreasing sequence of coordinates $u_i \leqslant u_{i+1}$. The parameters u_i are termed *knots* and the half-open interval $[u_i, u_{i+1})$ is called ith *knot span*. Each knot span has $p+1$ non-vanishing B-splines as illustrated in Fig. 2. Each basis function is entirely defined by $p+2$ knots and its support, $\text{supp}\{B_{i,p}\} = \{u_i, \ldots, u_{i+p+1}\}$, is local. Within each non-zero knot span s, $u_s < u_{s+1}$, of its support, $B_{i,p}$ is described by a polynomial segment $\mathcal{B}_i^s$. Each knot value indicates a location within the parameter space which is not C^∞-continuous, i.e., where two adjacent $\mathcal{B}_i^s$ join. Successive knots may share the same value, which is indicated by the knot multiplicity m, i.e., $u_i = u_{i+1} = \cdots = u_{i+m-1}$. In general, the continuity between adjacent segments is C^{p-m}. This control of continuity is demonstrated for a quadratic B-spline in Fig. 3. If the multiplicity of the first and last knot is equal to $p+1$, the knot vector is denoted an *open* knot vector. The knot sequence

$$\Xi = \left\{u_0 = \cdots = u_p,\ u_{p+1} = \cdots = u_{2p+1}\right\}, \tag{3}$$

is a special from of such a knot vector since it yields the classical pth-degree Bernstein polynomials. To be precise, Bernstein polynomials are usually defined over the interval

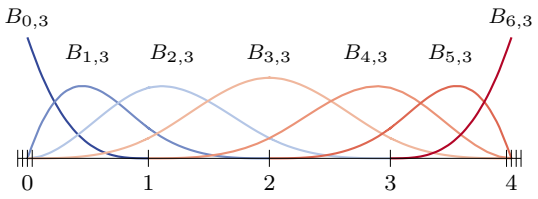


Fig. 4 B-spline basis specified by an open knot vector, i.e., $\Xi = \{0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4\}$

[0, 1]. If necessary, the restriction to this interval can be easily accomplished by a coordinate transformation.

As a whole, B-splines based on a common knot vector Ξ form a *partition of unity*, i.e.,

$$\sum_{i=0}^{I-1} B_{i,p}(u) = 1, \quad u \in \left[u_0, u_{I+p}\right], \tag{4}$$

and they are *linearly independent*, i.e.,

$$\sum_{i=0}^{I-1} B_{i,p}(u) c_i = 0, \tag{5}$$

is satisfied if and only if $c_i = 0,\ i = 0, \dots, I-1$. Due to the latter property, every piecewise polynomial $f_{p,\Xi}$ of degree p over a knot sequence Ξ can be uniquely described by a linear combination of the corresponding $B_{i,p}$. Hence, they form a *basis* of the space $\mathbb{S}_{p,\Xi}$ collecting all such functions

$$\mathbb{S}_{p,\Xi} = \sum_{i=0}^{I-1} B_{i,p} c_i, \quad c_i \in \mathbb{R}. \tag{6}$$

An example of a cubic B-spline basis defined by an open knot vector is shown in Fig. 4.

The first derivative of B-splines are computed by a linear combination of B-splines of the previous degree

$$B'_{i,p}(u) = \frac{p}{u_{i+p} - u_i} B_{i,p-1}(u) - \frac{p}{u_{i+p+1} - u_{i+1}} B_{i+1,p-1}(u). \tag{7}$$

For the computation of the kth derivative, this is generalized to

$$B^{(k)}_{i,p}(u) = \frac{p!}{(p-k)!} \sum_{\ell=0}^{k} a_{k,\ell} B_{i+\ell,p-k}(u), \tag{8}$$

with

$$\begin{aligned}
a_{0,0} &= 1, \\
a_{k,0} &= \frac{a_{k-1,0}}{u_{i+p-k+1} - u_i}, \\
a_{k,\ell} &= \frac{a_{k-1,\ell} - a_{k-1,\ell-1}}{u_{i+p+\ell-k+1} - u_{i+\ell}} \quad \ell = 1, \dots, k-1, \\
a_{k,k} &= \frac{-a_{k-1,k-1}}{u_{i+p+1} - u_{i+k}}.
\end{aligned}$$

Remark 1 The knot differences of the denominators involved in the recursive formulae (1), (7) and (8) can become zero. In such a case the quotient is defined to be zero.

2.2 Curves

B-spline curves of degree p are defined by basis functions $B_{i,p}$ due to a knot vector Ξ with corresponding coefficients in model space[1] $\boldsymbol{c}_i$ which denote *control points*. The geometrical mapping $\mathcal{X}$ from parameter space to model space is given by

$$\mathcal{X}(u) := \boldsymbol{C}(u) = \sum_{i=0}^{I-1} B_{i,p}(u) \boldsymbol{c}_i, \tag{9}$$

with I representing the total number of basis functions. The derivative is

$$\mathbf{J}_{\mathcal{X}}(u) := \sum_{i=0}^{I-1} B'_{i,p}(u) \boldsymbol{c}_i. \tag{10}$$

In general, control points $\boldsymbol{c}_i$ are not interpolatory, i.e., they do not lie on the curve. The connection of $\boldsymbol{c}_i$ by straight lines is called the *control polygon* and it provides an approximation of the actual curve. An important property of a B-spline curve is that it is contained within the *convex hull* of its control polygon. In particular, a polynomial segment related to a non-zero knot span s, i.e., $u \in [u_s, u_{s+1})$, is in the convex hull of the control points $\boldsymbol{c}_{s-p}, \dots, \boldsymbol{c}_s$. The continuity of the whole piecewise polynomial curve $\boldsymbol{C}(u)$ is inherited from its underlying basis functions, i.e., the continuity at knots is determined by the knot multiplicity, and the position of its control points. These relationships are illustrated in Fig. 5. Note that the interpolatory B-spline $B_{4,2}$ of Fig. 5a corresponds to the kink at $\boldsymbol{c}_4$ in Fig. 5b and that the second polynomial segment lies within the convex hull of $\boldsymbol{c}_1$ to $\boldsymbol{c}_3$. If the curve consist of a single polynomial segment, i.e., the associated Ξ is of form (3), the curve is referred to as *Bézier* curve. A polynomial segment of a B-spline curve is termed a Bézier segment, if it can be

[1] The model space is also referred to as physical space.

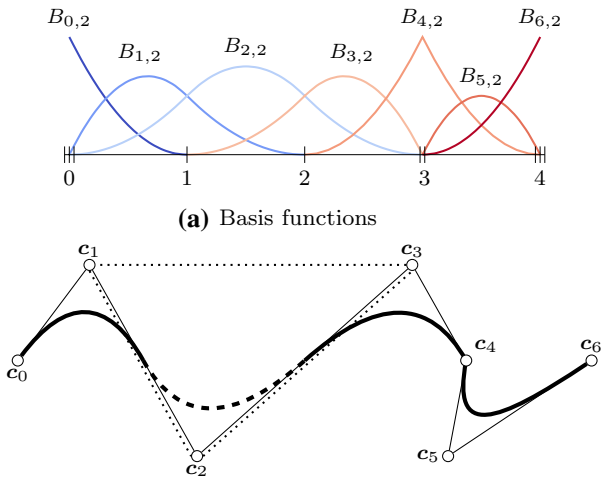


Fig. 5 Example of a B-spline curve: **a** B-splines based on $\Xi = \{0, 0, 0, 1, 2, 3, 3, 4, 4, 4\}$ and **b** a corresponding piecewise polynomial curve. In **b**, *circles* denote control points and the *dotted lines* indicate the convex hull of the *dashed curve* segment $u \in [1, 2)$

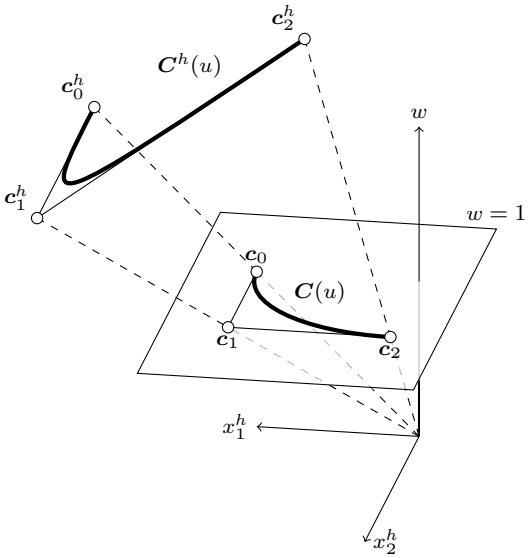


Fig. 6 Perspective mapping $\mathcal{P}$ of a quadratic B-spline curve $\boldsymbol{C}^h(u)$ in homogeneous form $\mathbb{R}^3$ to a circular arc $\boldsymbol{C}(u)$ in model space $\mathbb{R}^2$. The mapping is indicated by *dashed lines*

represented by a Bézier curve. In Fig. 5b, this is the case for the segment $u \in [3, 4]$ defined by the control points $\boldsymbol{c}_4$ to $\boldsymbol{c}_6$.

B-spline curves can be generalized to represent rational functions such as conic sections. For this purpose, *weights* w_i are associated with the control points such that

$$\boldsymbol{c}_i^h = \left(w_i \boldsymbol{c}_i,\, w_i\right)^\intercal = \left(\boldsymbol{c}_i^w,\, w_i\right)^\intercal \in \mathbb{R}^{d+1}, \tag{11}$$

where d denotes the spatial dimension of the model space. The *homogeneous coordinates* $\boldsymbol{c}_i^h$ specify a B-spline curve $\boldsymbol{C}^h(u)$ in a projective space $\mathbb{R}^{d+1}$. In order to obtain a curve in $\mathbb{R}^d$, the geometrical mapping (9) is extended by a perspective mapping $\mathcal{P}$ with the center at the origin of $\mathbb{R}^{d+1}$. This projection is given by

$$\boldsymbol{C}(u) = \mathcal{P}(\boldsymbol{C}^h(u)) = \frac{\boldsymbol{C}^w(u)}{w(u)}, \tag{12}$$

where $\boldsymbol{C}^w = (\boldsymbol{C}_1^h, \ldots, \boldsymbol{C}_d^h)^\intercal$ are the homogeneous vector components of the curve and the weighting function is determined by

$$w(u) = \sum_{i=0}^{I-1} B_{i,p}(u) w_i. \tag{13}$$

The application of Eq. (12) is illustrated in Fig. 6. The projection $\boldsymbol{C}(u)$ is denoted as a non-uniform rational B-spline (NURBS) curve. The term rational indicates that the resulting curves are piecewise rational polynomials, whereas the term non-uniform emphasizes that the knot values can be distributed arbitrarily.

The derivative of the NURBS geometrical mapping is defined by

$$\mathbf{J}_{\mathcal{X}}(u) := \frac{w(u)\frac{\partial \boldsymbol{C}^w(u)}{\partial u} - \frac{\partial w(u)}{\partial u}\boldsymbol{C}^w(u)}{(w(u))^2}, \tag{14}$$

with

$$\frac{\partial w(u)}{\partial u} = \sum_{i=0}^{I-1} B'_{i,p}(u) w_i, \tag{15}$$

$$\frac{\partial \boldsymbol{C}^w(u)}{\partial u} = \sum_{i=0}^{I-1} B'_{i,p}(u) \boldsymbol{c}_i^w. \tag{16}$$

Another way to represent NURBS curves is

$$\boldsymbol{C}(u) = \sum_{i=0}^{I-1} R_{i,p}(u) \boldsymbol{c}_i, \tag{17}$$

with

$$R_{i,p}(u) = \frac{w_i B_{i,p}(u)}{w(u)}. \tag{18}$$

The weighting function $w(u)$ is the same as in Eq. (13) and $R_{i,p}$ denotes a NURBS basis function. Since the weights w_i are now associated with B-splines $B_{i,p}$ the mapping (17) employs control points $\boldsymbol{c}_i$ of the model space. In general, NURBS curves degenerate to B-spline curves, if all weights are equal. Hence, they are a generalization of them.

The properties of B-spline curves apply to their rational counterpart as well, if the weights are non-negative, which is usually the case.

2.3 Spline Interpolation

In case of a spline interpolation problem, a given function f shall be approximated by a B-spline patch $I_h f := \sum_{i=0}^{I-1} B_{i,p} c_i$. They agree at I data sites $\bar{u}$ if and only if

$$f(\bar{u}_j) = \sum_{i=0}^{I-1} B_{i,p}(\bar{u}_j) c_i, \quad j = 0, \dots, I-1. \tag{19}$$

The corresponding system of equations consists of the unknown coefficients c_i and the *spline collocation matrix* $\mathbf{A}_u$ which is defined by

$$\mathbf{A}_u[j, i] = B_{i,p}(\bar{u}_j), \quad i, j = 0, \dots, I-1. \tag{20}$$

The *Schoenberg–Whitney theorem* [34, 83] states that the matrix $\mathbf{A}_u$ is invertible if and only if

$$B_{i,p}(\bar{u}_i) \neq 0, \quad i = 0, \dots, I-1. \tag{21}$$

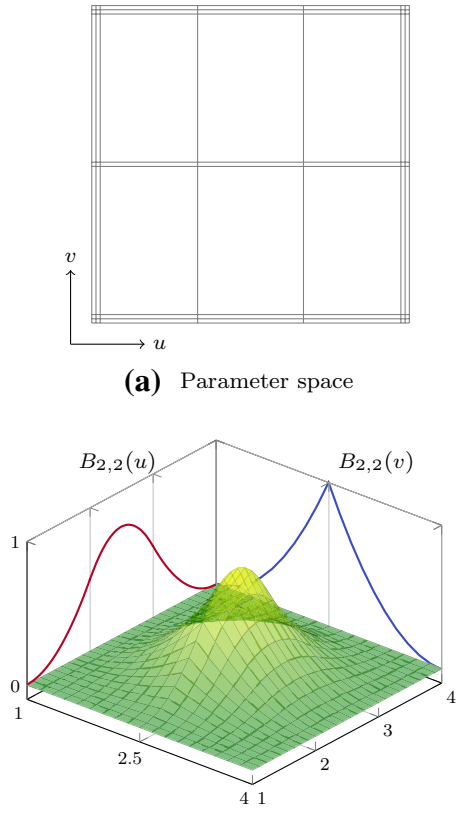
Since condition (21) guarantees that $\mathbf{A}_u$ does not become singular, it is expected that the corresponding condition number gets large if $\bar{u}$ approaches the limits of its allowed range. Non-uniformity of $\bar{u}$ is another reason for an increasing condition number. In fact, it gets arbitrary large if two interpolation sites approach each other, while the others are fixed. Several authors [11, 34, 184] recommend to interpolate at the *Greville abscissae* u^g which are obtained by the following knot average

$$u_i^g = \frac{u_{i+1} + u_{i+2} + \cdots + u_{i+p}}{p}. \tag{22}$$

These abscissae are well known in CAGD and used for different purposes, e.g., to generate a linear geometrical mapping [83]. The most important feature of this approach is that it induces a stable interpolation scheme for moderate degrees p.

2.4 Tensor Product Surfaces

Tensor product surfaces allow an extremely efficient evaluation of patches. They play an important role in CAGD. In particular, B-spline and NURBS patches are very common. Bivariate basis functions for B-spline patches are obtained by the tensor product of univariate B-splines which are defined by separate knot vectors Ξ_I and Ξ_J. These knot vectors determine the parameterization in the directions u and v, respectively. Moreover, they span the bivariate basis of a



(a) Parameter space

(b) Basis function

Fig. 7 A bivariate basis determined by $\Xi_I = \{1, 1, 1, 2, 3, 4, 4, 4\}$ and $\Xi_J = \{1, 1, 1, 2.5, 2.5, 4, 4, 4\}$ for u and v, respectively: **a** shows the bivariate basis spanned by Ξ_I and Ξ_J, whereas **b** illustrates the construction of a corresponding bivariate B-spline

patch. Combined with a bidirectional grid of control points $\boldsymbol{c}_{i,j}$ the geometrical mapping $\mathcal{X}$ is determined by

$$\mathcal{X}(u, v) := \boldsymbol{S}(u, v) = \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} B_{i,p}(u) B_{j,q}(v) \boldsymbol{c}_{i,j}. \tag{23}$$

The polynomial degrees are denoted by p and q, respectively for each parametric direction. The Jacobian of the mapping (23) is computed by substituting the occurring univariate B-splines by their first derivatives, alternately for each direction. In general, derivatives of B-spline patches are specified by

$$\frac{\partial^{k+l}}{\partial^k u \partial^l v} \boldsymbol{S}(u, v) = \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} B_{i,p}^{(k)}(u) B_{j,q}^{(l)}(v) \boldsymbol{c}_{i,j}. \tag{24}$$

The efficiency of tensor product surfaces stems from the fact that their evaluation can be performed by a successive evaluation of curves [62]. Suppose the parametric value v^{iso} is fixed, the surface equation yields

$$\begin{aligned} \boldsymbol{S}(u,\, v^{iso}) &= \sum_{i=0}^{I-1}\sum_{j=0}^{J-1} B_{i,p}(u) B_{j,q}(v^{iso}) \boldsymbol{c}_{i,j} \\ &= \sum_{i=0}^{I-1} B_{i,p}(u) \left(\sum_{j=0}^{J-1} B_{j,q}(v^{iso}) \boldsymbol{c}_{i,j} \right) \\ &= \sum_{i=0}^{I-1} B_{i,p}(u) \tilde{\boldsymbol{c}}_i = \boldsymbol{C}^{iso}(u), \end{aligned} \tag{25}$$

with $\boldsymbol{C}^{iso}(u)$ denoting an *isocurve* of the surface defined by new control points $\tilde{\boldsymbol{c}}_i$. Hence, a surface can be evaluated by $I + 1$ or $J + 1$ curve evaluations, depending which parametric direction is evaluated first.

The tensor product nature of the patches is illustrated in Fig. 7 by means of a bivariate basis. Note that the univariate knot values propagate through the whole parameter space. If both knot vectors of the resulting patch are of form (3), it is referred to as Bézier surface. NURBS surfaces are derived analogous to curves by the introduction of weights.

2.5 Constructing Patches by Boundary Curves

The most basic surface construction scheme is to connect two curves $\boldsymbol{C}_i$ with $i = 1, 2$ by a linear interpolation. The resulting surfaces are termed *ruled surfaces* and they are defined as

$$\begin{aligned} \boldsymbol{S}^r(u,\, v) &= (1 - v)\boldsymbol{C}_1(u) + v\boldsymbol{C}_2(u) \\ &= (1 - v)\boldsymbol{S}^r(u,\, 0) + v\boldsymbol{S}^r(u,\, 1), \end{aligned} \tag{26}$$

where $u, v \in [0, 1]$. If $\boldsymbol{C}_i$ have the same degree and knot vector, it is straightforward to represent $\boldsymbol{S}^r$ as a single tensor product surface. In this case the connection lines on $\boldsymbol{S}^r$ associate points of equal parameter value. Alternatively, the rulling (26) could also be performed according to relative arc length. This yields a different geometry which cannot be converted to a NURBS patch [230].

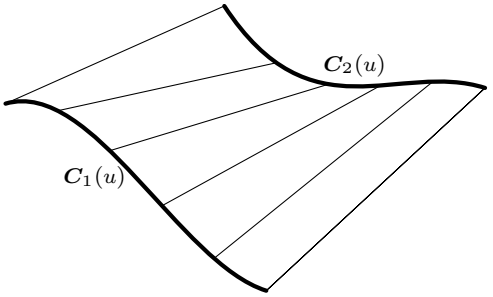
The construction of *Coons patches* is another very common procedure. Thereby, a surface $\boldsymbol{S}^c$ is sought to fit four boundary curves $\boldsymbol{C}_i(u)$ and $\boldsymbol{C}_j(v)$ with $i = 1, 2$ and $j = 3, 4$. The parameter range is again $u, v \in [0, 1]$. The curves have to satisfy the following compatibility conditions at the corners of the surface

$$\boldsymbol{S}^c(0,\, 0) = \boldsymbol{C}_1(u = 0) = \boldsymbol{C}_3(v = 0), \tag{27}$$

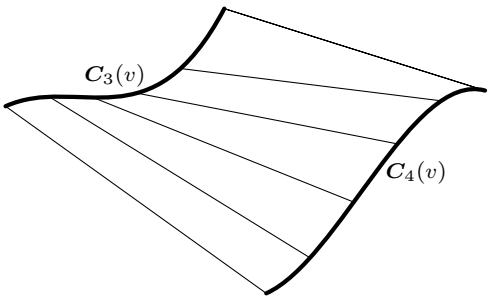
$$\boldsymbol{S}^c(1,\, 0) = \boldsymbol{C}_1(u = 1) = \boldsymbol{C}_4(v = 0), \tag{28}$$

$$\boldsymbol{S}^c(0,\, 1) = \boldsymbol{C}_2(u = 0) = \boldsymbol{C}_3(v = 1), \tag{29}$$

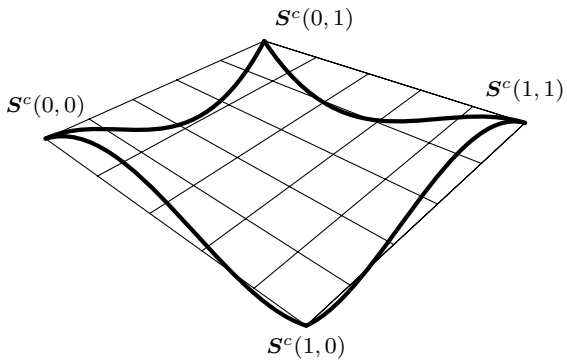
$$\boldsymbol{S}^c(1,\, 1) = \boldsymbol{C}_2(u = 1) = \boldsymbol{C}_4(v = 1). \tag{30}$$



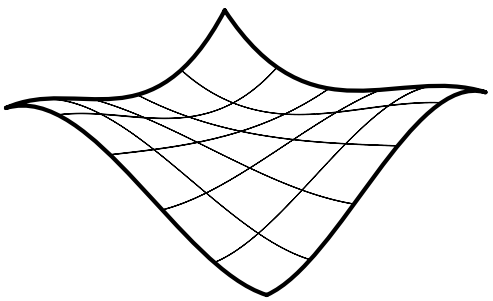
(a) Ruled surface $\boldsymbol{S}^r_u(u, v)$



(b) Ruled surface $\boldsymbol{S}^r_v(u, v)$

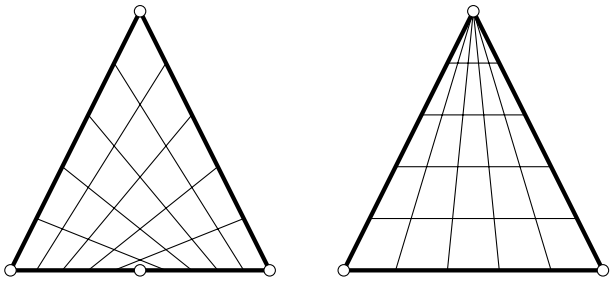


(c) Bilinear interpolation $\boldsymbol{S}_c(u, v)$



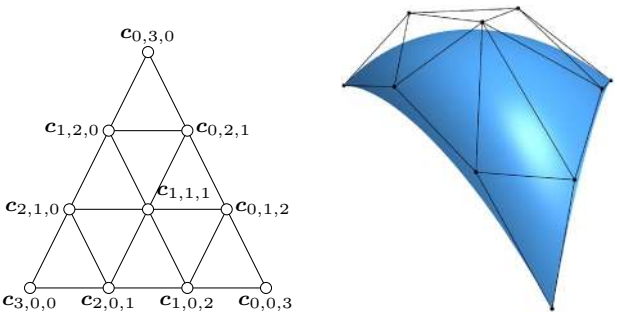
(d) Coons patch $\boldsymbol{S}(u, v)$

Fig. 8 Components of a bilinear Coons patch due to the boundary curves $\boldsymbol{C}_i(u)$ and $\boldsymbol{C}_j(v)$ highlighted by *thick lines*



(a) Degenerated side **(b)** Degenerated point

Fig. 9 Tensor product representation of triangular patches: **a** an angle between adjacent edges has 180° and **b** a side shrinks to a point. *Circles* mark the corner points of the resulting patch



(a) Control grid structure **(b)** Triangular patch

Fig. 10 Triangular Bézier patch of degree $p = 3$: **a** the general structure of the control grid and **b** a corresponding surface

Using a bilinear interpolation a Coons patch is given by

$$\boldsymbol{S}^c(u,\, v) = \boldsymbol{S}^r_u(u,\, v) + \boldsymbol{S}^r_v(u,\, v) - \boldsymbol{S}^r_c(u,\, v), \tag{31}$$

where $\boldsymbol{S}^r_u$ and $\boldsymbol{S}^r_v$ are ruled surfaces based on $\boldsymbol{C}_i(u)$ and $\boldsymbol{C}_j(v)$, respectively, and $\boldsymbol{S}^r_c$ is the bilinear interpolant to the four corner points

$$\boldsymbol{S}^r_c(u,\, v) = \begin{bmatrix} 1 \\ u \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \boldsymbol{S}^c(0,\, 0) & \boldsymbol{S}^c(0,\, 1) \\ \boldsymbol{S}^c(1,\, 0) & \boldsymbol{S}^c(1,\, 1) \end{bmatrix} \begin{bmatrix} 1 \\ v \end{bmatrix}. \tag{32}$$

These various parts of a Coons patch are visualized in Fig. 8. Equation (31) can be generalized by using two arbitrary smooth interpolation functions $f_0(s)$ and $f_1(s)$ fulfilling

$$f_k(\ell) = \delta_{k\ell}, \quad k,\, \ell = 0,\, 1, \tag{33}$$

and

$$f_0(s) + f_1(s) = 1, \quad s \in [0,\, 1], \quad s = u,\, v. \tag{34}$$

The corresponding Coons patch can be expressed in matrix form as

$$\begin{aligned} &\boldsymbol{S}^c(u,\, v) = \\ -&\begin{bmatrix} -1 \\ f_0(u) \\ f_1(u) \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} \mathbf{0} & \boldsymbol{S}^c(u,\, 0) & \boldsymbol{S}^c(u,\, 1) \\ \boldsymbol{S}^c(0,\, v) & \boldsymbol{S}^c(0,\, 0) & \boldsymbol{S}^c(0,\, 1) \\ \boldsymbol{S}^c(1,\, v) & \boldsymbol{S}^c(1,\, 0) & \boldsymbol{S}^c(1,\, 1) \end{bmatrix} \begin{bmatrix} -1 \\ f_0(v) \\ f_1(v) \end{bmatrix}, \end{aligned} \tag{35}$$

with $\mathbf{0} \in \mathbb{R}^d$ denoting the zero vector. Various functions may be used to specify f_k such as Hermite polynomials or trigonometric functions. In case of Bernstein polynomials, the surfaces $\boldsymbol{S}^r_u$, $\boldsymbol{S}^r_v$, and $\boldsymbol{S}^r_c$ are in Bézier or B-spline form and the resulting Coons patch can be represented as a single NURBS surface.

Finally, *Gordon surfaces* are a further generalization of Coons patches, where the surface $\boldsymbol{S}^r_u$ and $\boldsymbol{S}^r_v$ interpolate sets of isocurves rather than boundary curves. Gordon surfaces are also referred to as *transfinite interpolation* [103]. The term indicates that these surfaces interpolate an infinite number of points, i.e., the boundary curves and isocurves. Based on this definition, ruled surfaces, Coons patches, and Gordon surfaces may be generally referred to as transfinite interpolations.

2.6 Representation of Triangles

Triangular patches may be represented by tensor product surfaces despite their four-sided nature. Therefore, either a side or a point is degenerated as shown in Fig. 9. Such degenerated patches are often used since it is convenient to use only one surface representation. However, it is apparent that this can lead to a distorted parameterization. In addition, the enforcement of continuity between adjacent surfaces is difficult in this case.

An alternative is to use *triangular patches*. A point on such surfaces is defined by barycentric coordinates, i.e., $(r,\, s,\, t)$ with $r + s + t = 1$. We will focus on triangular Bézier patches $\boldsymbol{S}^{\triangle}$ which are specified as

$$\boldsymbol{S}^{\triangle}(r,\, s,\, t) = \sum_{\substack{i+j+k=p \\ i,\, j,\, k \geqslant 0}} B_{i,j,k,p}(r,\, s,\, t) \boldsymbol{c}_{i,j,k}, \tag{36}$$

with

$$B_{i,j,k,p}(r,\, s,\, t) = \frac{p!}{i!j!k!} r^i s^j t^k, \tag{37}$$

representing linearly independent bivariate Bernstein polynomials of degree p. The related control points $\boldsymbol{c}_{i,j,k}$ form a triangular array as shown in Fig. 10 for the cubic case. The resulting patch fulfills the convex hull property and its boundaries are Bézier curves. Rational triangular Bézier patches may be defined again by the introduction of weights. Despite the potential of triangular patches, there are currently no commercial CAD applications that admit the use of splines on triangulations.

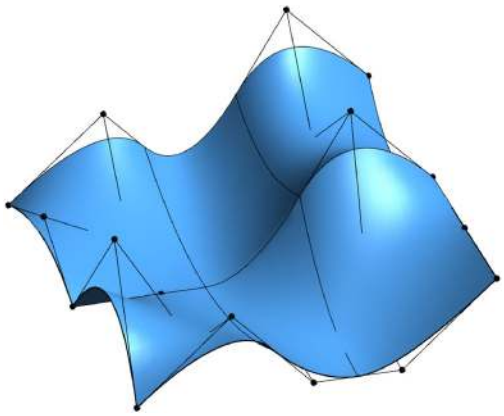
2.7 Trimmed Surfaces

In order to represent arbitrary surface boundaries when using tensor product surfaces, patches can be modified by trimming procedures. For this purpose, curves are defined within the parameter space of a surface $\boldsymbol{S}(u, v)$. These *trimming curves* $\boldsymbol{C}^t(\tilde{u})$ are usually B-spline or NURBS curves. They are given by

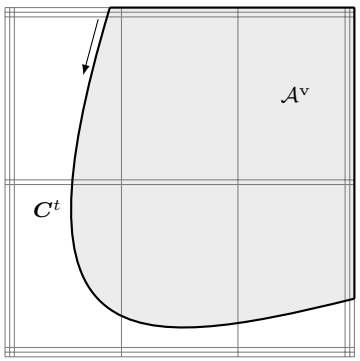
$$\boldsymbol{C}^t(\tilde{u}) = \begin{bmatrix} u(\tilde{u}) \\ v(\tilde{u}) \end{bmatrix} = \sum_{i=0}^{I-1} R_{i,p}(\tilde{u}) \boldsymbol{c}_i^t, \tag{38}$$

where $\boldsymbol{c}_i^t \in \mathbb{R}^2$ are the control points of the trimming curve given in the parameter space of the trimmed surface. Connected trimming curves are ordered such that they form a closed directed *loop*. Loops also include the boundary of the original patch if it is intersected by trimming curves. These loops divide the resulting *trimmed patch* into distinct parts where the curves' directions determine which parts are visible. In other words, trimming procedures are used to define visible areas $\mathcal{A}^{\mathrm{v}}$ over surfaces independent of the underlying parameter space.

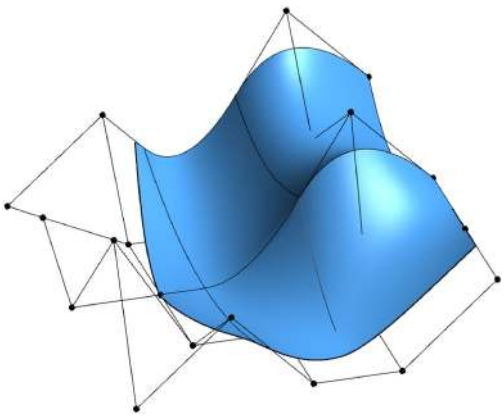
As a result, surfaces with non-rectangular topologies can be represented in a very simple way. An example of a trimmed patch is shown in Fig. 11. It is emphasized that the mathematical description, i.e., the tensor product basis and the related control grid, of the original patch does *not* change and is *never* updated to reflect the trimmed boundary represented by the independent trimming curves. Trimmed surfaces should be considered as an "engineering" extension of tensor product patches [83]. On the one hand, they permit a convenient way to define arbitrary surface topologies and provide a means for visually displaying them in graphics systems. On the other hand, they do not offer a canonical solution to related problems such as a smooth connection of two adjacent patches along a trimming curve, although the graphics system leads the user to incorrectly believe so. In fact, enormous effort has been and is still devoted to resolve the shortcomings of trimming procedures as discussed later on in Sect. 3.



(a) Regular B-spline patch



(b) Trimmed parameter space



(c) Trimmed patch

Fig. 11 Trimmed tensor product surface: **a** regular surface defined by a tensor product basis, **b** trimmed parameter space where a loop of trimming curves (*thick line*) specifies the visible part $\mathcal{A}^{\mathrm{v}}$ of **c** the resulting trimmed surface as displayed by a CAD system's graphics display. The *arrow* in **b** denotes the direction of the trimming curves

2.8 Solid Models

Most CAGD objects are geometrically represented by their boundary only. In other words, these models consist of several boundary patches γ

$$\Gamma = \bigcup_{i=1}^{I} \gamma_i, \tag{39}$$

where Γ denotes the entire boundary of the object. If Γ is a curve, several patches may be needed to represent distinct sections with different polynomial degrees. This is not a critical issue since curves can be joined rather easily, even with a certain continuity. However, a problem arises as soon as surfaces are considered, because tensor

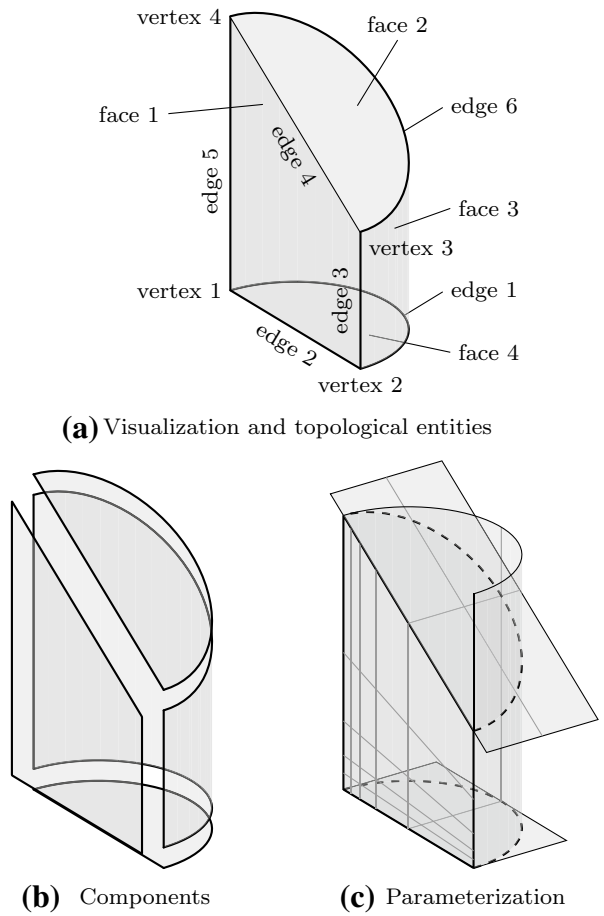


Fig. 12 Different perspectives of a CAGD solid model: **a** visible part of the object and its topological entities (to be precise, the related geometric objects, i.e., points, curves, and surfaces, are displayed), **b** the geometric segments of the B-Rep and **c** the underlying mathematical parameterization of each surface. In **c**, *dashed lines* mark the boundary of the visible area and *gray lines* indicate the underlying tensor product basis. Note that the parameterization along common edges does not match

product surfaces are four-sided by definition. A single regular NURBS surface may be closed equivalently to a cylinder or a torus. Spherical objects may be represented as well, if degenerated edges are introduced. Yet, more complicated objects such as a double torus require a partition into *multiple* NURBS patches. The connection of two adjacent surfaces is complicated, especially if a certain continuity is desired. In general, *non-conforming* parameterizations along surface boundaries need to be expected.

In addition to the geometric representation of the boundary patches, the *topology* of the object has to be described. It addresses the connectivity of the various components, and the corresponding entities are termed

– *vertices* relating to points,
– *edges* relating to curves,
– *faces* relating to surface.

It should be noted that the descriptions of an object's shape, i.e., geometry, and its structure, i.e., topology, are separated [290]. By definition, a B-Rep model always consists of a data structure of both topological and geometric objects. Regarding isogeometric analysis, the parameterization of the geometry is a further important issue that should be taken into account. Figure 12 summarizes these various perspectives of a CAGD model representing a simple solid. The corresponding object consists of three trimmed surfaces and an untrimmed, or regular one. It is apparent that even simple CAGD models rely on multiple non-conforming surfaces.

Finally, it should be mentioned that B-Reps are also used to describe dimensionally reduced objects, i.e., shell structures. In this case, the patches γ specify the object itself rather than its boundary. It is important to note that the terms *surface* model and *solid* model do not refer to the dimension of an object. In CAGD, they rather indicate if a model contains topology information (solid model) or not (surface model). Based on the brief outline given here, the discussion on the representation of CAGD models will be continued in Sect. 3.2.

3 Trimming in Computer Aided Geometric Design

There is a large body of literature on trimmed B-spline and NURBS geometries in CAGD. Trimming is addressed in the context of surface intersection, the development for appropriate data structures for solid modeling, the visualization of objects which is referred to as rendering, and remodeling approaches. The following outline of these topics is meant to be comprehensive, but it is by no means complete. Further, some auxiliary techniques are presented.

The motivation for this section is twofold: first of all, it provides an overview of the historical development of trimming approaches in the field of CAGD. Apart from being interesting in its own right, this insight exposes a number of general challenges, techniques, and ideas regarding trimmed geometries. Hence, it is hoped that the subsequent sections also give insight to further strategies dealing with trimming in the context of isogeometric analysis.

3.1 Surface Intersection

Trimming is closely related to the problem of surface-to-surface intersection. In general, the intersection of two parametric surfaces

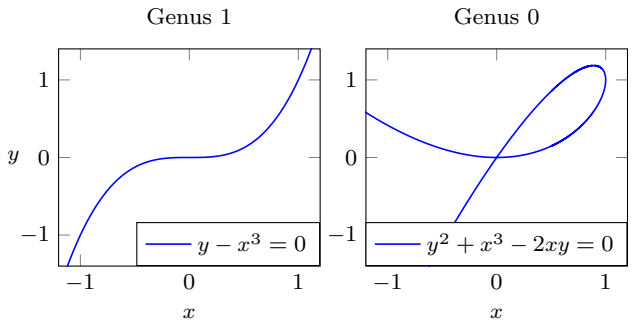


Fig. 13 Cubic curves with different genus. Note the double point in case of the genus 0 curve

$$\boldsymbol{S}_1(u, v) = \big(x_1(u, v), y_1(u, v), z_1(u, v)\big), \tag{40}$$

$$\boldsymbol{S}_2(s, t) = \big(x_2(s, t), y_2(s, t), z_2(s, t)\big), \tag{41}$$

leads to a system of three nonlinear equations, i.e., the three coordinate differences of $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$, with four unknowns u, v, s, t [62]. If surfaces intersect, the solution usually yields curves, but also subsurfaces or points may occur. The computation of intersections is one of many "geometric interrogation" techniques, or processes, employed in all types of modeling. The development of a good surface intersection scheme is far from trivial since the method has to balance three contradictory goals: *accuracy*, *efficiency*, and *robustness*. The surveys [219, 220] and the textbooks [4, 130, 221] provide detailed information on various approaches. Surface intersection algorithms can be broadly classified into four main categories: (i) analytic methods, (ii) lattice evaluation, (iii) subdivision methods, and (iv) marching methods.

3.1.1 Analytic Methods

The intersection of two surfaces may be solved analytically, i.e., an explicit representation of the intersection curve is obtained. Early solid modeling systems used analytic methods to obtain exact parametric representations of the intersection of quadratic surfaces [40]. The intersection problem always has a simple solution when both surfaces are given as functions in implicit form [130]. The good news is that parametric surfaces can always be represented implicitly [265], but the main problem is that the algebraic complexity of the intersection increases rapidly with the degree of the surfaces. This is often illustrated by a popular example of the intersection of two bicubic patches which has an algebraic degree of 324 as shown by Sederberg [265, 266]. In addition, the intersection of two bicubic patches has a genus[2] of 433 and only curves of genus 0, i.e., all degree two curves, cubic curves with one double point, quartic curves with three double points or one triple point, etc., can be expressed parametrically using rational polynomials [152]. Figure 13 illustrates two examples of implicit cubic curves with different genus. The complexity of surface intersection curves has also been discussed in the study of Farouki and Hinds [88]. It is argued that the derivation of an implicit representation is not practical and an approximation scheme may be preferred. In general, analytic methods have been restricted to low degree intersections, which yield exact results very fast.

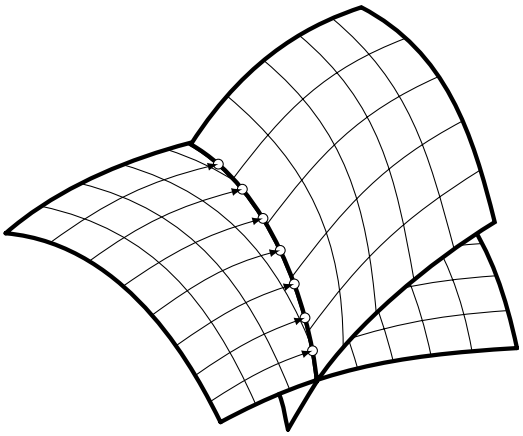


Fig. 14 Intersection points based on line-to-surface computations

3.1.2 Lattice Evaluation

The basic idea of this technique is to reduce the dimensionality of surface intersections by computing intersections of a number of isoparametric curves [186, 250] (see Fig. 14). Once the discrete intersection points are obtained, they are sorted and connected by an interpolation scheme. In order to define an intersection curve, lattice evaluation involves an initial choice of a proper grid resolution. This is crucial for both the robustness and efficiency of the method. Unfortunately, determination of an appropriate discrete step size is not straightforward and, if too coarse, may lead to a failure in identifying critical features [170].

Curve intersection schemes are also useful in the context of ray tracing for visualization and point classification in solid modeling [220]. In these applications a patch is intersected by a straight line, as discussed later on in Sects. 3.3.2 and 3.5.2.

[2] The genus g of a plane algebraic curve is specified by the degree p of the curve and the number I and multiplicity m of its singular points: $g = \frac{1}{2}\Big(p^2 - 3p + 2 - \sum_{i=1}^{I} m_i(m_i - 1)\Big)$.

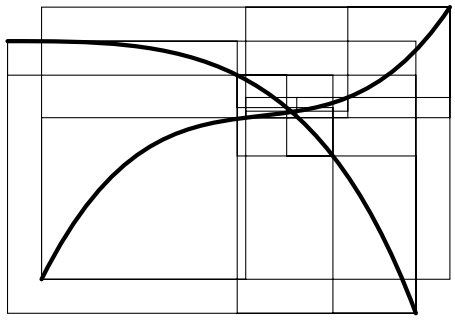


Fig. 15 Determination of an intersection region of two curves by means of a divide-and-conquer scheme that uses axis-aligned bounding boxes

3.1.3 Subdivision Methods

The key idea of subdivision approaches is to compute an intersection using approximations of the patches involved, rather than the actual objects themselves. These approximations are often defined by piecewise linear elements. Consequently, the intersection problem is subdivided into many, but significantly simpler, problems. The final intersection curve is obtained by merging the individual intersection results.

A good approximation of the original objects is of course essential for the accuracy of such techniques. However, subdivision algorithms become inefficient for high-precision evaluation, especially if the decomposition is performed uniformly. A considerable improvement can be achieved if the region of the intersection, i.e., the affected elements, is estimated in a preprocessing step. This can be carried out by *bounding boxes* that completely enclose the corresponding element. Various construction schemes of such bounding boxes are outlined in Sect. 3.5.1. Their common aim is to allow an efficient determination if two objects are clearly separated or not. In particular, elements are recursively refined if their bounding boxes overlap, which leads to a non-uniform, adaptive subdivision algorithm as shown in Fig. 15. This process of successive refinement and removing of separable boxes is referred to as a *divide-and-conquer* principle [130].

An important advantage of subdivision methods is that they do not require starting points. On the other hand, the drawbacks can be summarized according to Patrikalakis and Maekawa [221] as follows: (i) they are only able to isolate zero-dimensional solutions, (ii) there is no certainty that each root has been extracted, (iii) the number of roots in the remaining subdomains is typically not provided, and (iv) there is no explicit information about root multiplicities without additional computations. Last but not least, (v) the method is not efficient in case of high-precision or higher order evaluations [181, 219].

3.1.4 Marching Methods

Marching methods[3] derive an intersection curve by stepping piecewise along the curve, e.g., [14, 20, 84]. Such methods usually consist of a *search*, a *marching*, and a *sorting* phase. The first phase detects an appropriate starting point on the intersecting curve. Often, this is performed by a subdivision or lattice approach. In the marching phase, a point sequence along the intersection curve is developed starting from the points determined in the previous phase. The direction and the length of the next step are defined by the local differential geometry. Finally, the individual points and segments of the intersection curve are sorted and merged to disjoint pieces and curve loops.

According to Hoschek and Lasser [130], all marching methods share some common problems: (i) determination of good starting points, (ii) detection of all branches of the intersecting curve, (iii) avoiding of multiple detections of a intersection segment, (iv) correct behavior at self-intersections and singularities, (v) proper choice of the direction and length of the subsequent step, and (vi) a robust automatic stopping criterion.

Despite all these issues, marching methods are by far the most widely used approaches due to their generality and ease of implementation [170]. In addition, accuracy improvement can be easily achieved by decreasing the step size, and they are also very efficient, especially in combination with subdivision methods.

At this point, it should be emphasized that the problems related to topology detection of the intersection curve, i.e., finding all its branches and singular points, apply to all intersection methods and several authors have addressed them, e.g., [6, 105, 168, 222, 267, 283].

3.1.5 Hybrid Methods

Every intersection method type has its benefits and drawbacks, hence a number of authors have established hybrid methods that combine features of the different categories.

One of the elementary surface intersection schemes has been proposed by Houghton et al. [133]. The algorithm combines a divide-and-conquer approach with a Newton–Raphson procedure: firstly, the surface is subdivided into flat sub-pieces. Then, each sub-piece is approximated by two triangles and the intersection of these triangles is computed. In the next step, the resulting linear segments are sorted and connected using the information provided by the subdivision tree. Finally, the intersection points are refined by the Newton-Raphson scheme. The main advantage of this method is that it is very general and can be

[3] Marching methods are also referred to as tracing methods.

applied to any surface representation, in contrast to earlier techniques that utilize properties of certain surface types, e.g., [46, 111, 165, 180, 224, 256].

Barnhill et al. [19] presented another general procedure to compute the intersection of two rectangular C^1 patches. It relies on a combination of subdivision and a marching scheme. It does not assume a special structure of the intersecting surfaces and special cases are considered, e.g., infinite plane intersections, creases, and self-intersection. The algorithm has been enhanced in [20], including the utilization of the divide-and-conquer concept presented by Houghton et al. [133].

Another combination of a divide-and-conquer subdivision with an iterative marching approach has been developed by Kriezis et al. [169]. The method enables intersecting algebraic surfaces of any degree with rational biquadratic and bicubic patches.

Krishnan and Manocha [170] developed an approach for NURBS surfaces that combines marching methods with the algebraic formulation. The starting points on the intersection curve are computed by Bézier curve–surface intersections that are obtained by eigenvalue computations. Moreover, they introduced a technique that allows detection of singularities during the tracing process.

3.1.6 Representation of the Intersection Curve

Various techniques for the computation of approximate solutions to the surface-to-surface intersection problem have been outlined so far. It remains to discuss the actual representation of the result.

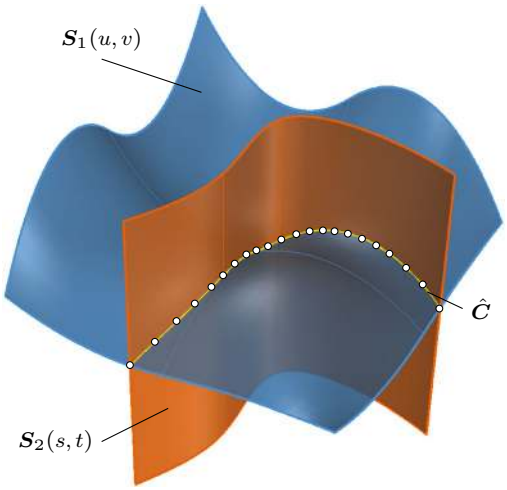
In general, three *distinct* representations of an intersection are obtained. On the one hand, the intersection curve in model space is computed. This may seem to be the main objective of the whole procedure at first glance, yet it is just a part of the overall solution process. The intersection curve has to be represented in each parameter space of the trimmed patches. These curves are referred to as trimming curves in the following and are needed to determine which surface points are visible.

Intersection and trimming curves can be defined by any kind of representation, but usually low-degree B-splines are used. They are constructed based on a set of sampling points that result from the surface-to-surface intersection algorithm applied [207]. Subsequently, an interpolation scheme or another curve-fitting technique is used to generate a continuous approximation of the intersection in model space $\hat{\boldsymbol{C}}$. In general, this curve does not lie on either of the intersecting surfaces. A trimming curve $\boldsymbol{C}^t$ is obtained based on the sampling points given in the corresponding parameter space [240]. The related curve $\tilde{\boldsymbol{C}}^t$ in the model space is obtained by evaluating the equation of the surface $\boldsymbol{S}$ along its $\boldsymbol{C}^t$. Alternatively, $\tilde{\boldsymbol{C}}^t$ may be represented explicitly. DeRose et al. [71] presented an efficient and stable algorithm based on blossoming[4] that can be use to exactly compute the control points of $\tilde{\boldsymbol{C}}^t$. Such an expansion of $\boldsymbol{S}\circ\boldsymbol{C}^t$ into an explicit representation $\tilde{\boldsymbol{C}}^t$ may be used to join another patch to a trimmed surface, but there is no computational benefit [189]. Curves on surfaces have a degree of $p(m+n)$ with p denoting the degree of the trimming curve and m and n correspond to the degrees of the trimmed surface [83]. Renner and Weiß [240] compared exact and approximate representations of $\tilde{\boldsymbol{C}}^t$ and concluded that high degree is the main reason for preferring an approximation scheme. Furthermore, they formulated the following requirements for such a scheme: (i) low degree, (ii) fast and stable generation, (iii) full control over deviation between exact curve and approximation, and (iv) consideration of the specific surface geometry. According to them, these requirements are often not satisfied in current CAD systems. Besides the schemes presented in [240], several other approaches have been proposed to compute good approximate curves on surfaces, see e.g., [90, 119, 317].

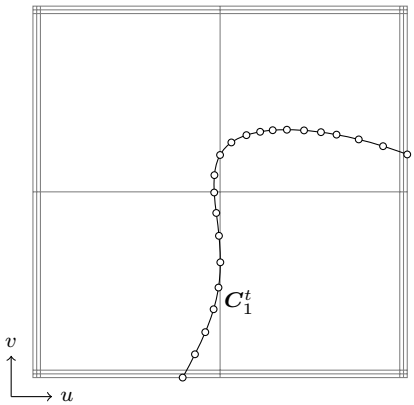
It is emphasized that $\tilde{\boldsymbol{C}}^t$ does *not* coincide with the intersection curve in model space $\hat{\boldsymbol{C}}$, regardless of its representation. In addition, all procedures related to trimming curves are performed for each patch separately. Hence, the images of these curves $\tilde{\boldsymbol{C}}_i^t$ do not coincide, neither with each other, nor with $\hat{\boldsymbol{C}}$. As a consequence, gaps and overlaps may occur between intersecting patches. There is *no connection* between these three representations of the intersection; although the sample points provide some information during the construction, this data is only stored temporarily during the approximation procedure and never retained in memory for further use. These various approximations of a surface-to-surface intersection are summarized in Fig. 16.

Currently, the most common geometric modeling kernels are `ACIS`, `C3D`, and `Parasolid`. They provide software components for the representation and manipulation of objects, and form the geometric core of many CAD applications. All of them use splines for the description of trimming curves [43, 65, 282]. Yet, the representation of the intersection curve in model space varies: `ACIS` defines it by a three-dimensional B-spline curve, `Parasolid` uses a set of sorted intersection points that can be interpreted as a linear approximation, and in `C3D` the intersection curve is not stored at all. In `C3D`, trimming curves are computed such that they have the same radius and derivatives at the same parametric values. However, this is only satisfied at the intersection point used for the construction, for more details see [102].

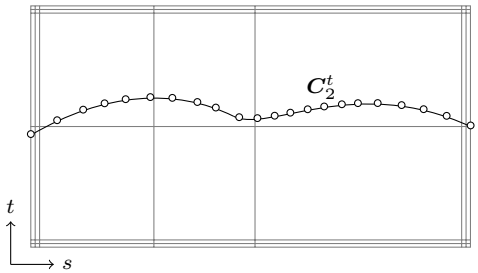
[4] A multi-affine and totally symmetric mapping is called a blossom (or polar form). Blossoms can be used to define spline algorithms in an elegant way. For details see [236].



(a) Model space



(b) Parameter space of $S_1(u,v)$



(c) Parameter space of $S_2(s,t)$

Fig. 16 Independent curve interpolation of an ordered point set to obtain approximations of the intersection of two patches $S_1(u, v)$ and $S_2(s, t)$. The set of sampling points depends on the surface-to-surface intersection algorithm applied. The subsequent interpolation of these points is performed in **a** the model space and the parameter space of **b** $S_1(u, v)$ and **c** $S_2(s, t)$ leading to the curves $\hat{C}$, C_1^t, and C_2^t, respectively. The point data is usually discarded once the curves are constructed

3.2 Solid Modeling

Solid modeling is concerned with the use of unambiguous representations of three-dimensional objects. It is based on a consistent set of principles for mathematical and computer modeling, and focuses on informational completeness, physical fidelity, and universality [276]. An essential aspect is the *topology* of complex models. The consideration of topology is indeed the fundamental difference between solid models and surface models, because the latter describes only the geometry of an object [5]. Topological properties are not metrical, but address connectivity and dimensional continuity of a model [207]. There are several textbooks on solid modeling [120, 194, 207, 290] and for an elaboration of the historical development of this field of research the interested reader is referred to the landmark paper of Requicha [242] and the subsequent surveys by him and his co-authors [241, 243, 244, 251]. In the following, the progress towards a trimmed solid model as well as the related challenges are outlined.

3.2.1 Formulation of Trimmed Solid Models

Pierre Bézier outlined the idea of trimming already in the 1970s. In his paper [29], it is proposed to perform the segmenting of a model by curves defined on a square patch as illustrated in Fig. 17. These curves are termed "transposant", the present participle of the French word for transpose. The concept reduces the amount of data and enables an easier blend with other patches. However, this idea was presented with little theoretical support and solid modeling requires an adequate mathematical theory as emphasized in a survey by Requicha and Voelcker [243].

It took some time to develop a rigorous way to represent trimmed free-form models. There are three broad categories for representing geometric objects: (i) decomposition, (ii) boundary, and (iii) constructive representations. Popular examples of decomposition representations are voxel models where a solid is approximated by identical cubic cells. Advantages and limitations of this approach are discussed in [153]. A B-Rep[5] (B-Rep) defines an object by its bounded geometry, along with an associated topological structure of corresponding entities, such as faces, edges, and vertices. The benefits of storing an object's shape by means of its boundary were already elaborated in the seminal work of Braid [36]. Most B-Reps consist of several surface patches and additional information is stored to efficiently identify the various components and their relation to each other [241]. Various data structures for B-Reps have been used, e.g., [21, 73, 106], to find a compromise between storage requirements and response to topological questions. The best known constructive representation is so-called constructive solid geometry (CSG) [241]. Simple

[5] B-Rep models with free-form surfaces are also referred to as sculptured models.

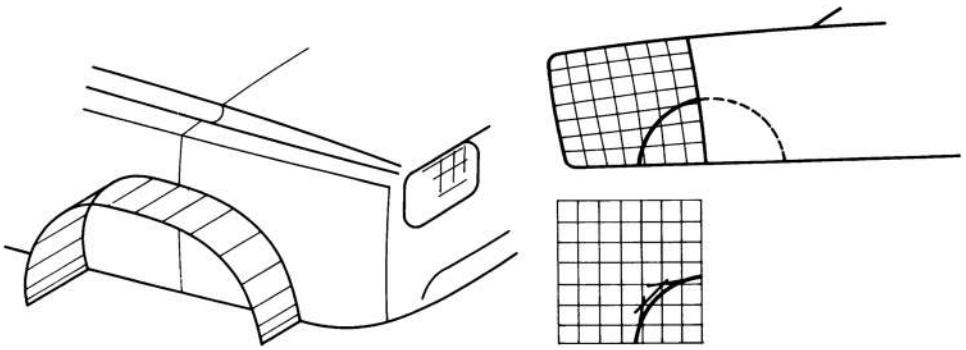


Fig. 17 An early sketch of a trimmed surface (reprinted from [29], with permission from Elsevier)

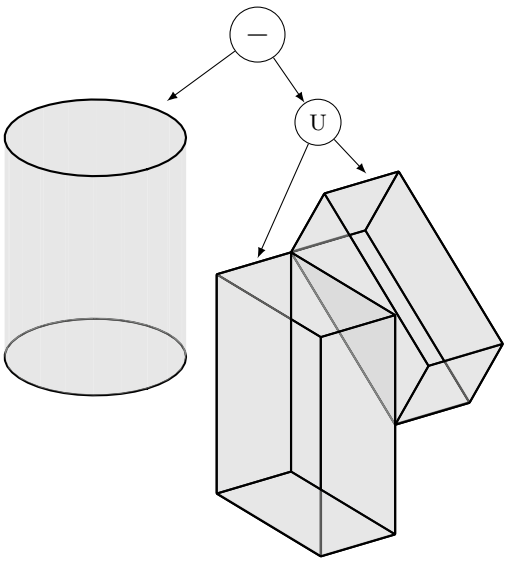


Fig. 18 Representation of the object shown in Fig. 12a by means of a CSG tree: the object is specified by a composition of simple solids using Boolean operations, i.e., union (U) and difference (−)

primitives are combined by means of rigid motions and regularized Boolean operations—union, intersection, and difference. The resulting object is represented by a binary tree where the internal nodes correspond to the Boolean operations and the primitive solids (or half spaces) are given in the leaves. An example of such a tree is given in Fig. 18.

Remark 2 The developments of isogeometric analysis and additive manufacturing using heterogeneous materials yield to a growing interest in another representation of three-dimensional geometric models, namely *volumetric representations* (V-Reps). As a matter of fact, several researchers in the CAGD community have started addressing this issue, see e.g., [31, 197, 321], including the definition of trimmed V-Reps [203].

Trimmed solid models combine concepts of B-Rep and constructive representation, i.e., they consist of free-form B-Reps merged by Boolean operations [245]. A unification of CSG and free-form surfaces was presented in the early 1980s [56], perhaps for the first time. It was proposed to use models with straight edges but free-form surface interpolation in between. In the context of Boolean operations, however, the merging of B-Reps and CSG is more involved. Gossard et al. [104] developed a polyhedral modeler which combines the two representations by means of a graph structure. In particular, two relative position operators have been implemented. For manifold polyhedral objects, the implementation of Boolean operations is well understood [194]. However, the definition of a convenient representation for trimmed NURBS is a major challenge since the topology of patches becomes quite complicated if Boolean operations are performed [120]. B-Rep solid modeling utilizes surface-to-surface intersection schemes to create arbitrarily defined free-form geometric entities, but the corresponding algorithms require more than just computing the intersection curve. Weiler's thesis [310] provides a study on topological data structures. He summarized the essential attributes of geometric modeling operators as follows:

(i) Determination of the topological descriptions,
(ii) Determination of the geometric surface descriptions,
(iii) Guarantee that the geometry corresponds unambiguously to the topology.

Setting up a topology requires the classification of the neighborhood of various entities (faces, edges, and vertices) involved in the intersections [120]. The correlation of topology and geometry becomes particularly complicated if intersection curves have singularities or self-intersections. In addition, various forms of set membership classification, i.e., the determination if parts are inside, outside, or on the boundary of a domain, are used to compute B-Reps through Boolean operations. In order to determine if a surface point is inside or outside of the surface, the trimming curve must be defined in the parameter space as noted before. If the trimming curve would only be defined in the model space, the problem would be in fact ill-defined [205].

The first formulation of a trimmed patch representation that supports Boolean operations and free-form geometry was presented in the late 1980s by Casale and Bobrow [47, 49]. The domain of trimmed patches is specified by the two-dimensional equivalent of a CSG tree. Hence, a B-Rep is obtained that contains topology information of its trimmed components. Patches are intersected similar to the divide-and-conquer procedure of [133], but the trimming curve is then also transformed into the parameter space in order to perform Boolean operations and set membership classifications. At the same time, a rigorous trimmed surface definition has been formulated by Farouki [85]. The formulation is based on Boolean operation definitions. In particular, a trimmed patch is given by its parametric and implicit surface equations together with a *trimming boundary* that is defined as a tree structure of non-intersecting and nested piecewise-algebraic loops. These loops consist of monotonic branches. The integral over the trimmed surface is determined by a proper tessellation of the patch. It should, however, be pointed out that all these approaches fail to guaranty exact topological consistency since the images of the trimming curves do not match in general, as noted by Farouki et al. [87]. This leads to gaps and overlaps of the solid model, which can introduce failure of downstream applications such as numerical simulations.

3.2.2 Robustness Issues

Several robustness issues arise in case of imprecise geometric operations. As a matter of fact, the numerical output from simple geometric operations can already be quite inaccurate. Complications may occur even for linear elements as discussed by Hoffmann [120–123] or the computation of the convex hull of a set of points [16], for instance. These problems are induced by propagation of numerical conversion, roundoff, and digit-cancellation errors of floating point representation. The issue of rounding errors of numerical computations is known for a long time, at least since an early study by Forsythe [92]. Investigating the effects of floating point arithmetic on intersection algorithms is an important area of research [220]. Since intersection problems can be expressed as a nonlinear polynomial system of equations, the robustness issue maybe addressed from a computational point of view. Troubles arise if the problem is ill-conditioned which is for example the case for tangential intersections and surface overlaps [135, 195].

The key issue is that numerical errors may cause misjudgment as pointed out in [291]. Since the geometrical decisions are based on *approximate* data and arithmetic operations of *limited* precision, there is an interval of uncertainty in which the numerical data cannot yield further information [122]. Of course, the situation gets even more delicate if trimmed free-form surfaces are involved where approximation errors are quite apparent due to the gaps and overlaps between intersecting patches. In case of topological decisions, the accumulation of approximation errors is especially crucial since inaccuracy leads to inconsistency of the output as indicated in Fig. 19.

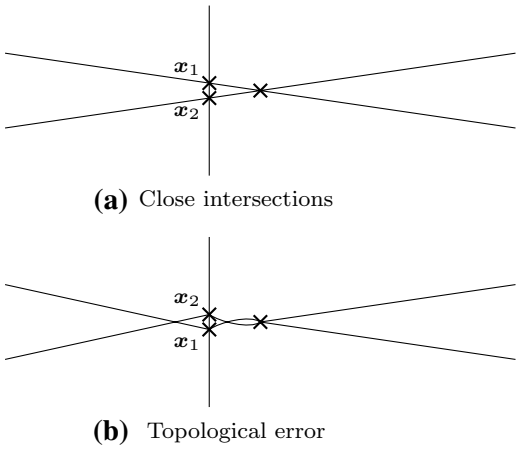


Fig. 19 Example of an incorrect topology: **a** two intersection points x_i are close together which may lead to **b** an incorrect topological placement along the *vertical line* due to numerical approximation errors (re-execution of the original example [204])

There is a large amount of research that addresses the issue of accurate and robust solid modeling. The various concepts are outlined in the following subsections. The approaches are based on tolerances, interval arithmetic, and exact arithmetic.

3.2.2.1 Tolerances Often, tolerances are used to assess the quality of operations like the computation of an intersection [19, 130]. Several authors have suggested to use adaptive tolerances where each element of the model is associated with its own tolerance, e.g., [143, 271]. In addition, tolerances may be dynamically updated [82]. Robustness of topology decisions may be improved by choosing the related precision higher than the one for the input data [291]. Another strategy is to adjust the data in order to obtain topologically consistent functions [204]. There are various other approaches that improve the application of tolerance and the interested reader is referred to the review of Hong and Chang [128] for a comprehensive discussion. In fact, all common CAD software tools are based on a user-defined tolerance that determines the accuracy of the geometrical operations performed. For example, the default tolerance values of `ACIS` are 10^{-6} for the comparison of points and 10^{-3} for the difference of an approximate curve or surface to its exact counterpart [65]. Unfortunately, tolerances cannot guarantee robust algorithms since they do not deal with the inherent problem of limited-precision arithmetic.

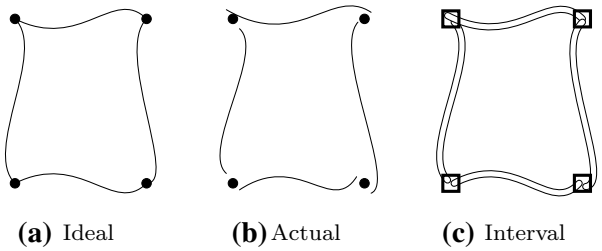


Fig. 20 Four splines representing a quadrilateral: **a** ideal mathematical object, **b** floating point model with approximation errors, and **c** interval arithmetic based representation

3.2.2.2 Interval Arithmetic In case of interval arithmetic, e.g., [76, 208], numerical errors are taken into account by associating an interval of possible values to a variable. This approach is correct in the sense that result intervals are guaranteed to contain the real number that is the value of the expression. The interval size indicates the reliability of floating point computations. In particular, a narrow interval is obtained in case of a successful operation whereas a wide interval reveals a risk [117]. This concept may be modified to rounded intervals to assure that the computed endpoints always contain the exact interval, see e.g., [3, 57, 193]. Interval arithmetic may also be combined with backward error analysis [255].

In the context of solid modeling, Hu et al. [134–136] suggested to use *interval NURBS*, i.e., NURBS patches with interval arithmetic. The control points are described by interval numbers rather than real numbers. Consequently, they are replaced by control boxes and thus, curves and surfaces are represented by slender tubes and thin shells, respectively. A conceptional sketch is illustrated in Fig. 20. The object is defined by a graph with nodes representing the topological entities. Each node has two lists: one for higher dimensional nodes and another for lower dimensional nodes that are arranged in counterclockwise order. This data structure has been applied to Boolean operations [136] and various intersection problems including ill-conditioned cases [134]. Gaps between actual intersecting objects are avoided and no intersection point is missed. However, objects that do not intersect each other originally, may do after several geometric processing steps using rounded interval arithmetic. Furthermore, interval arithmetic approaches cannot achieve very high precision in reasonable computation time [220].

3.2.2.3 Exact Arithmetic In order to achieve robustness, algorithms have been developed that are based on exact arithmetic, which is the standard in symbolic computation [318]. Most of these approaches focus on polyhedral objects, see e.g., [26, 93, 291]. For linear geometries like planes and their intersections, exact rational arithmetic is enough to handle all necessary numbers. However, more involved objects rely on real algebraic numbers and therefore, they require more complicated data structures and algorithms. Keyser et al. [157–159] presented such a scheme for curve-to-curve intersection in a plane and the theoretical framework for exact computation based on algebraic numbers has been discussed by Yap [318]. A combination of exact approach and floating point calculation may also be used as suggested by Hoffmann et al. [123]. In their paper, symbolic reasoning is used when floating point calculation yields ambiguous results. Krishnan et al. [171] demonstrated that exact arithmetic can be applied to large industrial models. They presented a B-Rep modeling system dealing with models using over 50,000 trimmed Bézier patches.

Still, the main drawback of such approaches is their efficiency. Exact computation can be several orders of magnitude slower than a corresponding floating point implementation [156]. According to Patrikalakis and Maekawa [220], much research remains to be done in bringing such methods to practice. In particular, more efficient algorithms should be explored that are generally applicable in low and high degree problems.

3.2.2.4 Concluding Remarks Overall, the formulation of *robust* solid models with trimmed patches is still an open issue. Tolerance based approaches are usually preferred since they are faster than the more precise ones. Hence, there is again a tradeoff between efficiency, accuracy, and robustness as discussed in the context of intersection schemes. Of course, the importance of these properties to the object representation strategy depends on the application context [45].

In fact, the problem of topologically correct merging of trimmed surfaces is such a challenge that more recent research in CAGD tries to circumvent this issue by employing other surface descriptions like T-splines and subdivision surfaces. These representations inherently possess a consistent topology and corresponding models are *watertight*, i.e., they do not have unwanted gaps or holes. However, they also have some drawbacks and the transformation of the original object usually leads to approximations, at least in the vicinity of the intersection curve as discussed later on in Sect. 3.4.

3.3 Rendering

In computer graphics rendering refers to the process of generating images of a CAGD model. There are two different ways to approach this goal. On the one hand, indirect schemes first tessellate the surfaces of the object and the actual visualization is based on this render-mesh. On the other hand, rendering may be performed directly on free-form surfaces by ray tracing. For a general introduction

to the creation of realistic images, the interested reader is referred to the textbook of Glassner [99].

3.3.1 Tessellation

All commercial rendering systems tessellate free-form surfaces before rendering, because it is more efficient to optimize the code for a single type of primitive [27].

Early on, trimmed surfaces had been rendered using the de Boor [33], Oslo [60], or Boehm's knot insertion [32] algorithm. In addition to the subdivision, the regions must be sorted to find out which ones are hidden and have to be removed for rendering, see e.g., [52, 179, 292]. In general, subdivision approaches are expensive if they are performed to pixel level.

The rendering of trimmed NURBS surfaces can also be carried out using a combination of subdivision and adaptive forward differencing [185, 275]. This method allows fast sampling of a large number of points, but suffers from error propagation. The main drawback in rendering transparent objects is the redundant pixel painting in adaptive forward differencing. Furthermore, the overall performance of the algorithm obtained is rather slow [191].

Rockwood et al. [249] presented a scheme enabling rendering of trimmed surfaces in real-time. Firstly, the surface is tessellated, i.e., approximated by linear triangles or other polygons. Therefore, all surfaces are subdivided into individual Bézier patches. A trimmed Bézier patch may be subdivided further to obtain monotone regions that have convex boundaries in the parameter space [165]. Each patch is tessellated into a grid of rectangles which are connected to the region boundaries by triangles. The actual rendering is performed on the approximate mesh. This idea has been adapted and enhanced by several other authors, e.g., [2, 176, 191].

The triangulation of trimmed surfaces by a restricted Delaunay triangulation has been proposed by Sheng and Hirsch [279]. The basic idea of this technique is to compute the approximation mesh in the parameter space. Although it has been developed for stereolithography[6] applications, the suitability for rendering is emphasized. Stereolithography was also the motivation in [74] where trimmed surfaces are triangulated by an adaptive subdivision scheme. In contrast to the approach by Rockwood et al. [249], both algorithms contain strategies to avoid cracks between patches. A general discussion on how to avoid edge gaps in case of an adaptive subdivision is given by Dehaemer and Zyda [69].

According to Vigo and Brunet [300], the main drawback of the approaches previously mentioned [249, 279] is that the resulting elements may be odd-shaped, especially near the boundary. They suggested to overcome this issues by a piecewise linear approximation of trimmed surfaces using a triangular mesh that is based on a max-min angle criterion. The algorithm is designed so that the resulting mesh can be used for stereolithography, FEA, and rendering. The mesh obtained consists of shape-regular elements and has no cracks between patches.

The determination of a proper step size of a tessellation is of course an important issue. The elements should not be too small in order to avoid oversampling of the surface, nor too big, since this would decrease the quality of the rendering [1]. Lane and Carpenter [178] presented a formula for calculating the upper bound of the distance between a right triangle interpolating a surface. Later, the bound was improved by Filip et al. [89]. Based on this work, Sheng and Hirsch [279] derived the following formula for arbitrary triangles: the approximation error can be estimated by the difference of a parametric surface $\boldsymbol{S}(u, v)$ to a linear triangle $\boldsymbol{T}(u, v)$

$$\sup_{(u,v)\in T} \|\boldsymbol{S}(u, v) - \boldsymbol{T}(u, v)\| \leq \frac{2}{9}\lambda^2\big(M_1 + 2M_2 + M_3\big), \tag{42}$$

where T is the correspond region in the parameter space, λ denotes the maximal edge length of $\boldsymbol{T}(u, v)$, and M_i are specified by

$$M_1 = \sup_{(u,v)\in T} \left\| \frac{\partial^2 \boldsymbol{S}(u, v)}{\partial^2 u} \right\|, \tag{43}$$

$$M_2 = \sup_{(u,v)\in T} \left\| \frac{\partial^2 \boldsymbol{S}(u, v)}{\partial u \partial v} \right\|, \tag{44}$$

$$M_3 = \sup_{(u,v)\in T} \left\| \frac{\partial^2 \boldsymbol{S}(u, v)}{\partial^2 v} \right\|. \tag{45}$$

Hence, the upper bounds of second derivatives of the surface are required. Once these bounds are determined, λ can be computed for a given tolerance ε by

$$\lambda = 3\left(\frac{\varepsilon}{2(M_1 + 2M_2 + M_3)} \right)^{1/2}. \tag{46}$$

The bounds on the second derivatives for a B-spline surface (43)–(45) can be computed by constrained optimization [89] or conversion to a Chebyshev basis [279]. Piegl and Richard [229] use the fact that the derivative of a B-spline is again a B-spline to define the upper approximation bounds by computing the maxima of the control points of the differentiated surfaces. They address the treatment

[6] Stereolithography is an early and widely used 3D printing technique.

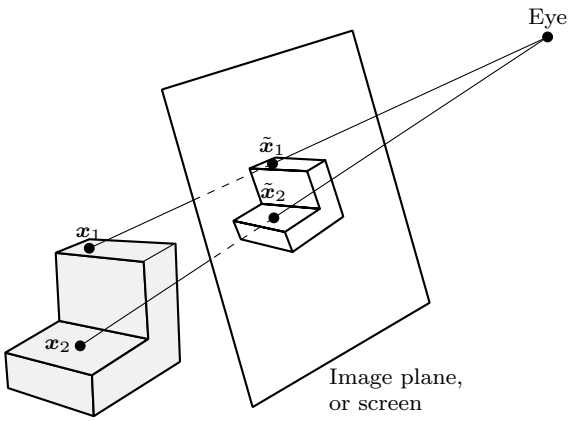


Fig. 21 Rays spawned from an eye-point in order to get a pixel-wise image of an object

of rational surfaces by means of homogeneous coordinates and adjustment of the tolerance due to the perspective mapping (12).

A few years later, Piegl and Tiller [231] proposed a triangulation scheme which is geometry-based, i.e., the procedure is based on the geometry rather than the parameterization. The trimming curves are polygonized in the model space by cubic Bézier curves and the surface itself is subdivided by its control net. The main advantage of this approach is that the trimmed NURBS surface is not required to have more than C^0-continuity, in contrast to the previous methods that assumed that the surfaces are C^2-continuous in order to estimate a step length in the parameter space [89]. Elber [79] proposed two alternative approaches that are also independent of the parameterization: one based on an intermediate linear surface fit and another based on global normal curvature. In general, tessellations do not require an element connectivity or shape regular elements. However, several authors have presented the construction of conforming meshes for trimmed patches that yield triangles with good aspect ratios [55, 57, 58].

Irregular meshes are also an issue regarding hardware implementation on the graphical processing unit (GPU). Moreton [206] presented tessellation of polynomial surfaces for hardware rendering using forward differences and dividing the work of tessellation between CPU and GPU. To avoid gaps along shared boundaries of patches due to the different floating point engines, all boundary curves of the patches are calculated on the GPU. In order to enable GPU based tessellation, Guthe et al. [108] presented a *trim texture* scheme which can be parallelized. In this approach, the visible domain is specified based on a texture-map of black and white pixels, hence the trimming task is performed on pixel-level.

3.3.2 Ray Tracing

In contrast to tessellation, ray tracing tries to compute an image one pixel at a time [76]. Every object in a scene is tested if it intersects with rays spawned from an eye-point as indicated in Fig. 21. The result must return at least the closest intersection point and the corresponding normal of the surface for each ray. Hence, the heart of any ray tracing package is the set of ray intersection routines [99].

Ray tracing is a powerful, yet simple approach to image generation [144]. Already early attempts of this technique have been successfully applied for automatic shading of objects [8], modeling of global lightning effects [311], and the visualization of fuzzy reflections and blurred phenomena [64]. In the context of parametric surfaces, the pioneering works focus on different ray-surface intersection methods [144, 146, 297]. Various intersection algorithms have been developed. One of the first algorithms used a lattice approach where the problem is reduced to finding the root of univariate polynomials [146]. Several numerical methods based on Newton schemes have been employed, e.g., [198, 293, 297]. Nishita et al. [214] introduced a ray tracing technique for trimmed patches based on Bézier clipping. This concept has been adapted and enhance by a large number of researchers, e.g., [44, 78, 97, 217, 303]. Bézier clipping is discussed in more detail in Sect. 3.5.2.

Ray tracing of trimmed patches has also been addressed. Usually, the untrimmed surface is intersected first and the determination if the intersection point lies inside or outside of the trimmed domain is performed in a subsequent step. This point classification task can be employed by ray-tests, e.g., [198, 214, 261]. The regions that require trimming may be identified in a preprocessing step in order to improve the performance [97]. Section 3.5.2 provides more information on the ray-test concept. An alternative way of point classification is to generate a trim texture that returns whether a point is inside or not [108]. This approach is very efficient since it requires only a single texture look-up to classify a domain point. However, the trim texture has to be updated every time the view changes [313].

One of the greatest challenges of ray tracing is efficient execution [99]. Hence, many researchers have focused on this issue. Early attempts improved the performance by means of bounding box trees, e.g., [198, 316]. Havran [116] compared a number of such schemes and concluded that the kd-tree is the best general-purpose acceleration structures for CPU. Kd-trees define a binary space partition that always employs axis-aligned splitting planes. Pharr et al. [226] have shown that coherence can be exploited to improve ray tracing. Their rendering algorithms improve locality of data storage and data reference. Further improvement can be obtained by simplifying and streamlining the basic algorithms in order to exploit performance features of

processors like single instruction, multiple data extensions [27, 97, 302]. Purcell et al. [235] demonstrated that the entire ray tracing process can be performed on the GPU. Since then, several other GPU based approaches emerged, e.g., [91, 172, 217, 261].

Despite the efficiency deficit compared to tessellation schemes, direct rendering of surfaces has several advantages. The memory requirements and preprocessing costs are reduced since fewer primitives are used and geometric precision and image quality are improved by eliminating artifacts [27].

3.4 Remodeling of Trimmed Models

Solid models with trimmed surfaces suffer from robustness issues that may lead to inconsistencies as previously discussed in Sect. 3.2.2. In order to obtain an unambiguous and watertight description of a solid model, several authors considered replacing trimmed objects by other surface representations. In particular, it has been suggested to remodel trimmed surfaces by means of a set of regular patches, subdivision surfaces, or T-splines.

3.4.1 Regular Patches

The treatment of trimmed surfaces in the early automotive industry was discussed by Sarraga and Waters [257], in which a *repatching* method is proposed. To be precise, the intersection curves are used as edges of new regular patches approximating the original surface. As pointed out by Sarraga and Waters, repatching has several distinct disadvantages for modeling, but it is applied as a compromise between the complexity of free-form surfaces and the requirements of solid modeling. The common aim of the subsequent approaches is to improve this compromise. Besides the desire for an unambiguous and robust solid model, exchange of geometric data between dissimilar CAD software has been a motivation for this remodeling concept. Various constructions for the repatching procedure have been proposed. Hoschek and Schneider [131] convert trimmed rational Bézier patches into a set of bicubic and biquintic Bézier patches. The segmentation is based on arguments related to the curvature of the surface and conditions on the geometrical continuity. The procedure combines some of Hoschek's previous works, i.e., [129, 132], and consists of four steps: (i) determination of new geometrically oriented boundary curves, (ii) approximation of these curves, (iii) fitting of the interior of each patch using geometric continuity conditions for the boundary and corner points, and (iv) approximation of the intersection curves of trimmed surfaces. The use of ruled surfaces [110], Coons patches [41, 301], and Clough–Tocher splines[7] [167] have also been suggested to remodel trimmed surfaces. Another concept is based on clipping isoparametric curves of a B-spline surface [9]. Later, this approach has been adapted for the design of aircraft fuselages and wings [304, 320].

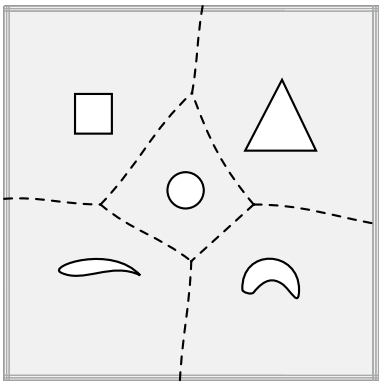


Fig. 22 Generalized Voronoi diagram for five trimming curves (re-execution of the original figure of [110])

Generalized Voronoi diagrams may be used to obtain a proper decomposition of the trimmed domain with multiple trimming curves [110, 142]. Thereby, the parameter space is partitioned into convex polygons such that each polygon contains exactly one trimming curve as illustrated in Fig. 22. Details on Voronoi diagrams can be found in the survey of Aurenhammer [10].

Another strategy to remodel trimmed models is local *perturbation*. In contrast to repatching, the control points of the original surfaces are modified in order to obtain an unambiguous configurations along the intersection curves. Hu and Sun [137] proposed to close gaps between trimmed B-spline surface by an algorithm that moves one of the patches towards the trimming curve defined by the other one. This approach modifies the control point of the patch near the trimming curve using singular value decomposition. It can be used to improve the accuracy of *small* gaps, but yields bad-shaped surfaces if the gaps are too large. Moreover, this approach does not produce an exact topological consistency. Song et al. [285] defines the differences of corresponding trimming curves by means of a so-called error curve in model space. It is specified so that its coefficients depend linearly upon the control points of the intersecting surfaces. The perturbation is carried out by setting all coefficients of this curve to zero. This is found by solving a linear system of equations and results in an adaptation of the control points. A complement to this work was presented by Farouki et al. [87]. They propose

[7] Clough–Tocher is a splitting scheme to construct C^1-continuous splines over triangulations.

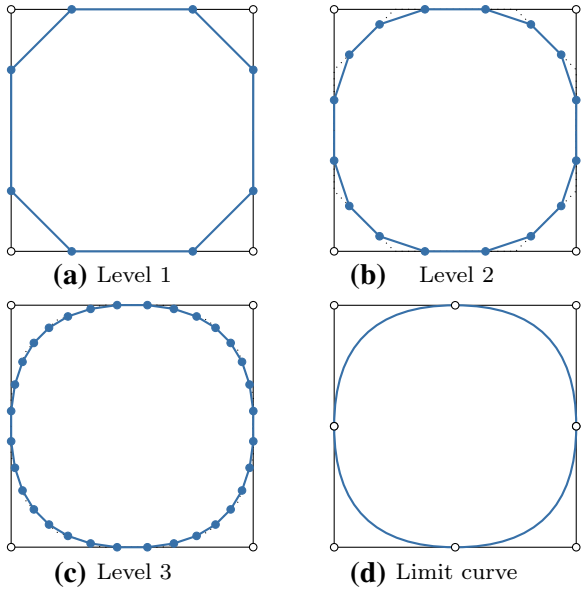


Fig. 23 Chaikin's corner-cutting algorithm: construction of a quadratic B-spline curve by a subdivision of the control polygon

to remodel trimmed surface by a hybrid collection of tensor product patches and triangular patches. In particular, they demonstrated the approximation of trimmed bicubic patches by quintic triangular patches such that the intersection curves are explicitly defined by one side of a triangular Bézier patch. The approach considers pairs of rectangular patches that intersect along a single diagonal arc. This can be achieved by a preprocessing step as described in the follow-up paper [115].

3.4.2 Subdivision Surfaces

The basic concept of subdivision approaches goes back to the 1970s. Chaikin developed an elegant algorithm to draw a curve by cutting the corners of a linear polygon [54]. The basic steps of the procedure are shown in Fig. 23. Later, it was shown that this cutting algorithm converges to a quadratic B-spline curve and the initial polygon is equivalent to its control polygon [246]. This idea of sequential subdivision of a control polygon was generalized by Doo and Sabin [75] as well as Catmull and Clark [53] to compute bi-quadratic and bi-cubic B-spline surfaces, respectively. Since then, a vast number of different subdivision schemes emerged for various surface types, such as triangular splines [190] and NURBS patches [51], for instance. The final objects of subdivision schemes are referred to as *limit* curves or surfaces. The distinguishing feature of these approaches is that they can be applied to arbitrary control polygons which are not restricted to a regular grid structure. The smoothness between the resulting surfaces is controlled by the subdivision scheme.

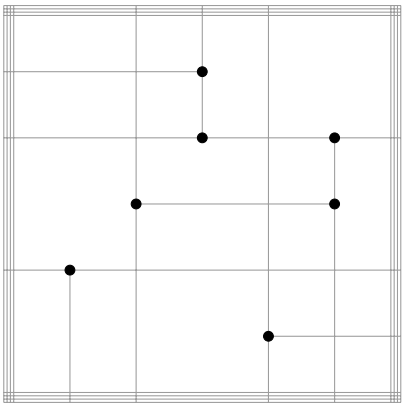


Fig. 24 An example of a parameter space with T-junctions which are highlighted by *circles*

In 2001, the issue of Boolean operation for subdivision surfaces was addressed. Litke et al. [188] presented a trim operator, but do not address surface-to-surface intersections. An algorithm for approximate intersections was developed by Biermann et al. [30]. High accurate results can be achieved at additional computational expense. Both approaches employ subdivision based on triangular splines. Shen et al. [278] convert trimmed NURBS surfaces to untrimmed subdivision surfaces using Bézier edge conditions. The limit surface fits the original object to a specified tolerance. The resulting Catmull–Clark models are watertight and smooth along the intersection. Recently, Shen et al. [277] presented a generalization of the approach that converts B-Rep models of regular and trimmed bicubic NURBS patches to a single NURBS-compatible subdivision surface. During this process, a quadrilateral mesh topology is constructed in the parameter space of each patch and the corresponding control points are computed by solving a fitting problem. Finally, the individual parts are merged into a single subdivision mesh. In order to obtain gap-free joints, the preserved boundary curves in model space are used as target curves of the subdivision surface.

Subdivision models possess a greater flexibility due to their inherent topological consistence while conventional NURBS models have greater control of an objects shape. This attribute of subdivision attracted considerable attention, especially in the field of computer animation [70]. For a detailed discussion on subdivision schemes the interested reader is referred to the textbook [308].

3.4.3 T-splines

T-splines were introduced by Sederberg et al. [270] in 2003. They are generalizations of B-splines that allow T-junctions in the parameter space and the control net of

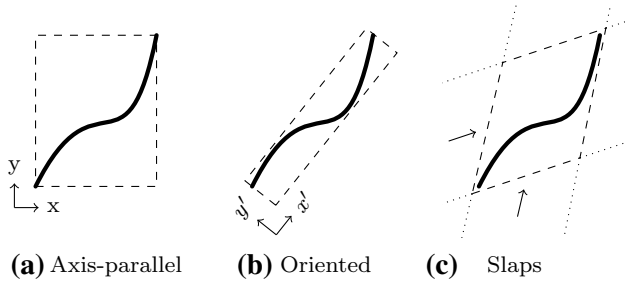


Fig. 25 Various types of bounding boxes for the same curve. The orientation of the enclosing region is indicated by *arrows*

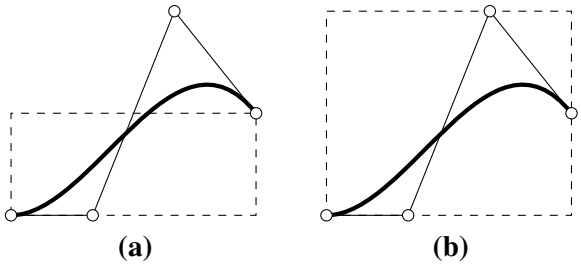


Fig. 26 Construction of axis-parallel bounding boxes by **a** the endpoints and **b** the control points of a spline

a surface as illustrated in Fig. 24. In a subsequent paper [268], the related ability of local refinement is used to close gaps between trimmed surfaces by converting them to a single watertight T-spline model. The resulting T-spline representation can be converted to a collection of NURBS surfaces again, without introducing an approximation error. On the other hand, conversion to the T-spline representation includes some perturbation in the vicinity of the intersection. It is argued that the approximation error can be made arbitrarily small, and the perturbation can be confined to an arbitrarily narrow neighborhood of the trimming curve. The conversion is performed such that C^2-continuity is obtained between the intersecting surfaces. These papers are focused on cubic splines since they are the most important ones in CAGD. However, the T-spline concept is not restricted to the cubic case.

3.5 Auxiliary Techniques

Techniques and strategies frequently used in the context of trimming are outlined in this section. They may be useful for researchers dealing with trimmed models in isogeometric analysis.

3.5.1 Bounding Boxes

Bounding boxes are often applied to significantly accelerate geometrical computations. The basic idea is to use rough approximations of objects in order to get a fast indicator if two regions are well separated or not. Hence, involved operations have to be carried out only if necessary. These approximations may be refined adaptively as in divide-and-conquer based surface intersection approaches introduced in Sect. 3.1.3.

The simplest and perhaps most common approach is to embed objects into *min-max* boxes where the corner points of the object define an axis-parallel box. The axis aligned setting is not mandatory but allows the most efficient evaluation of the distance between two boxes [116]. Some authors suggest to use oriented bounding boxes to improve the geometry approximation, e.g., [15, 20, 133]. In this case, the bounding box is rotated such that it is aligned with the connection of the corner points of the surface it encloses. An object can also be bounded by a combination of *slaps*, also known as *fat lines*, with different orientations [154]. Slaps denote regions between two parallel planes which are specified by their normal vector. This concept includes conventional bounding boxes, simply by using two orthogonal slaps. Figure 25 summarizes these various bounding box types.

Bounding boxes constructed by corner points do not guarantee the enclosing of the whole spline, especially if a spline is highly curved. The *convex hull* property of the control points can be used in order to get a proper approximation. Consequently, the area of the bounding box increases since it is computed based on the control polygon rather than the actual geometry, as illustrated in Fig. 26. Sederberg and Nishita [269] proposed an optimized bound for planar quadratic and cubic Bézier curves. They suggested defining the bounding region by lines parallel to the connection ℓ of the first and last control point. They are determined by the minimal and maximal distance d_i of the other control points $\boldsymbol{c}_i$ perpendicular to ℓ. The tighter bound is determined in the quadratic case by

$$d_{min} = \min\left\{0, \frac{d_1}{2}\right\} \quad \text{and} \quad d_{max} = \max\left\{0, \frac{d_1}{2}\right\}, \tag{47}$$

while for the cubic splines it is

$$d_{min} = \alpha \cdot \min\left\{0, d_1, d_2\right\}, \tag{48}$$

$$d_{max} = \alpha \cdot \max\left\{0, d_1, d_2\right\}, \tag{49}$$

with the scaling factor α given by

$$\alpha = \begin{cases} \frac{3}{4} & \text{if } d_1 d_2 > 0, \\ \frac{4}{9} & \text{otherwise.} \end{cases} \tag{50}$$

Figure 27 illustrates the improvement of the bounding boxes due to these bounds. Note that the bounding boxes

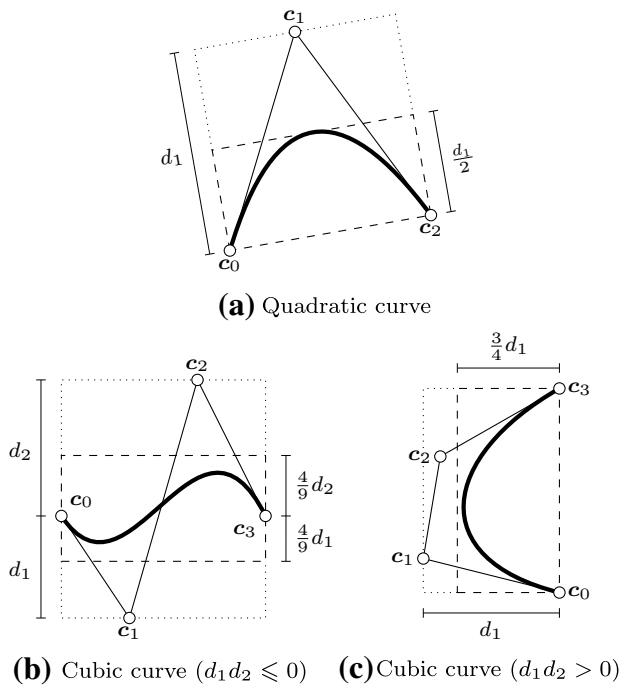


Fig. 27 Tighter bounds for bounding boxes for quadratic and cubic B-spline curves. The original bounding boxes are shown by *dotted lines* whereas *dashed lines* mark the improved ones

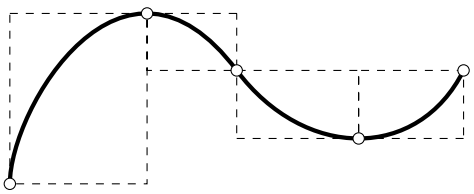


Fig. 28 Definition of axis-parallel bounding boxes based on monotonic regions. The *white points* mark the characteristic points considered

are oriented according to the locations of the first and last control point.

Another way to assure that a curve lies within its bounding box is to subdivide it into *monotonic regions*. The essential idea is that if a domain of any continuously differentiable function f is subdivided at its characteristic values, the range of f on each of the subintervals can be simply found by evaluating f at the endpoints of that subinterval [165, 208]. The set of characteristic points may include zeros of the first or second derivatives of f, start and end points of open curves, and singular points such as cusps or self-intersections. Figure 28 shows an example of a B-spline curve that has been divided into monotonic regions and the corresponding bounding boxes. In order to detect these points, a preprocessing step is required. Despite this additional effort, monotonic regions have been used in several application like intersecting planer curves [156, 157] and surfaces [84], tessellation of trimmed NURBS [249], and ray tracing [261].

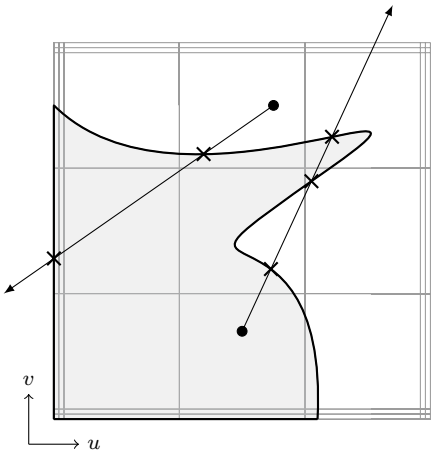


Fig. 29 Classification of interior and exterior points by counting intersections of the trimming curve with a ray

3.5.2 *Point Classification*

One of the most fundamental operations in the context of trimmed surfaces is the determination if a point $\boldsymbol{x}$ of a patch is inside *or* outside the visible domain. This can be done by counting the number of intersections of a ray emanating from $\boldsymbol{x}$ with the trimming curves and the boundary of the patch. If the number is odd $\boldsymbol{x}$ is inside and otherwise it is outside of the visible area. The direction of the ray can be chosen arbitrary. This rule is based on the Jordan curve theorem, that is, every simple *closed planar curve* separates the plane into a bounded interior and an unbounded exterior region [109]. Hence, the intersection is determined in the parameter space of the patch, in contrast to the ray tracing approach for rendering outlined in Sect. 3.3.2. Furthermore, if a trimming curve is not closed, it is associate to the visible part of the patch boundary to obtain a closed loop as illustrated in Fig. 29. Another possibility is to connect open trimming curves with the non-visible boundary of the patch and intersect only with the trimming curves. It should be noted that in the latter case, the even-odd rule turns upside down, i.e., $\boldsymbol{x}$ is inside the visible domain if the number of intersections is even.

Despite its conceptional simplicity, the implementation of the corresponding algorithm is not trivial [77]. For example, ambiguous cases may occur like tangency between the ray and the curve. Nishita et al. [214] proposed the following procedure: the ray is chosen such that it intersects perpendicularly with the closest boundary of the patch. As a consequence, the parameter space is divided into four quadrants which meet at the origin of the ray as

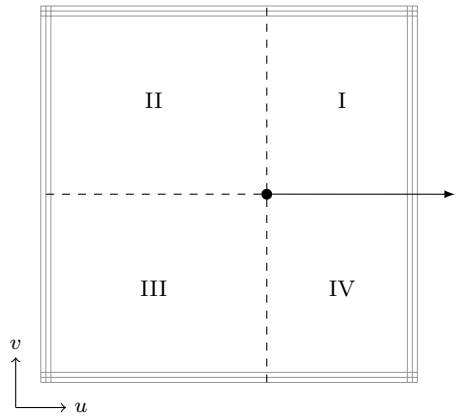


Fig. 30 Specification of quadrants for the point classification procedure of Nishita et al. [214]

shown in Fig. 30. They are labeled counter-clockwise such that the quadrants I and IV are adjacent to the ray. If the trimming curve is specified as a set of Bézier curves the following cases may be considered:

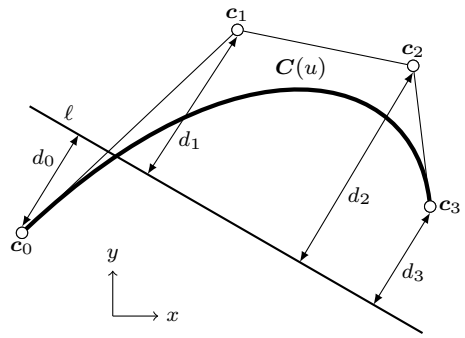
(i) There are no intersections with the ray if *all* control points of the trimming curve are within the quadrants I and II, or II and III, or III and IV.
(ii) All control points of a trimming curve are within the quadrants I and IV. The number of intersections is even if the endpoints of curve are in the *same* quadrant; otherwise it is odd.

Tangency between the ray and the trimming curve do not pose any problem for these exclusion criteria. However, it should be pointed out that an intersection may be counted twice if the ray goes through an endpoint which is shared by two trimming curves. For the other cases where the intersection has to be computed, Nishita et al. [214] suggested to employ *Bézier clipping*. This concept has been introduced by Sederberg and Nishita [269] in the context of curve-to-curve intersection and locating points of tangency between two planar Bézier curves. The basic idea is to use the convex hull property of Bézier curves to identify regions of the curves which do not include the solution. The bounding regions are defined by fat lines parallel and perpendicular to the line through the endpoints of the Bézier curve. By iteratively clipping away such regions, the algorithm converges to the solution at a quadratic rate and with a guarantee of robustness.

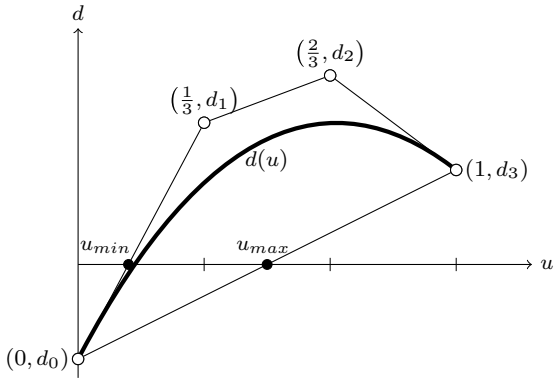
In particular, the ray is defined implicitly by

$$ax + by + c = 0 \quad \text{with} \quad a^2 + b^2 = 1. \tag{51}$$

The coordinates are denoted by x and y in order to emphasize that this approach is applicable for any plane



(a) Intersection



(b) Non-parametric curve

Fig. 31 Bézier clipping: **a** intersection of a ray ℓ with a Bézier curve $\boldsymbol{C}(u)$ and **b** the corresponding non-parametric Bézier curve which is used to determine the parameter range $[u_{max}, u_{min}]$ that contains the intersection of $\boldsymbol{C}(u)$ and ℓ

coordinate system. The distance $d(u)$ of a point on the Bézier curve $\boldsymbol{C}(u)$ to the ray ℓ is given by

$$d(u) = \sum_{i=0}^{p} d_i B_{i,p} \quad \text{with} \quad d_i = ax_i + by_i + c. \tag{52}$$

The coefficients d_i are the distances of the control points $\boldsymbol{c}_i$ of the Bézier curve to the ray and $B_{i,p}$ are Bernstein polynomials of degree p. Equation (52) can be represented as a non-parametric Bézier curve $\tilde{\boldsymbol{C}}(u, d(u))$ where the values d_i are related to their corresponding Greville abscissae, i.e., $u_i = \frac{i}{p}$. The relationship of the original and non-parametric Bézier curve are depicted in Fig. 31.

The roots of $\tilde{\boldsymbol{C}}(u, d(u))$ are equivalent to the parametric values u at which ℓ intersects $\boldsymbol{C}(u)$. Hence, the convex hull of $\tilde{\boldsymbol{C}}(u, d(u))$ can be used to identify regions where the objects do not intersect. To be precise, the minimal and maximal parametric values, i.e., u_{min} and u_{max}, of the intersections of the convex hull with the u-axis splits the parameter space into three regions of which only one,

i.e., $u_{min} \leqslant u \leqslant u_{max}$, has to be considered for the intersection with the ray. This region is extracted as a Bézier curve by means of knot insertion and the procedure is repeated until a certain tolerance is reached.

This technique can also be utilized to determine the intersection of two Bézier curves by iteratively clipping both objects [269]. Sherbrooke and Patrikalakis [280] developed a generalization of Bézier clipping that allows computing the roots of an n-dimensional system. The so-called *Projected-Polyhedron* method subdivides an object into Bézier segments and generates each side of its bounding boxes by projecting the control points onto different planes. Thus, only the convex hull of two-dimensional point sets has to be computed.

3.6 Summary and Discussion

The previous parts of this section shed light on various aspects of trimmed NURBS in the context of CAD. On the basis of the discussion of surface intersection in Sect. 3.1, it can be concluded that there is no canonical way to derive trimming curves, but a wide range of different techniques to address this problem. Further, an exact representation of an intersection of two patches is not feasible in most cases. In fact, several distinct curves are usually used to specify a single intersection: a curve in model space and trimming curves in the parameter spaces of all patches involved. These curves are *independent* approximations of the actual intersection and there is no connection between them. This missing link makes it very difficult to transfer information from a trimmed patch to an adjacent one.

The necessity of approximation yields gaps and overlaps between intersecting patches. As a result, robustness problems arise for solid modeling as outlined in Sect. 3.2. Considerable effort has been devoted to derive consistent trimmed models. Still, the problem is unresolved and the absence of truly robust representations poses a demanding challenge, especially for downstream applications. Even within the field of CAGD, the replacement of trimmed surfaces by other representations may be needed. Tessellations are used in the context of rendering, for instance. In this particular case, the main reason is efficiency as discussed in Sect. 3.3. However, all other remodeling approaches presented in Sect. 3.4 are motivated by the flaws of trimmed models and the limitation of tensor product patches due to their four-sided nature. It should be emphasized that these schemes may yield watertight models, but there are certain tradeoffs. First of all, approximations are introduced at least in the vicinity of trimming curves. The number of control variables increases particularly if regular tensor product patches are used for the remodeling. Subdivision surfaces and T-splines are promising techniques, but may induce new problems like extraordinary vertices[8] and linear dependence of the basis functions. In addition, they are designed for a specific surface type which may be an issue if a model consist of parts with different polynomial degree. Overall, it is apparent that there is no simple solution to the trimming problem.

Despite their difficulties, trimmed NURBS are *the* standard in engineering design and for the exchange of geometrical information in general. On the one hand, trimmed tensor product surfaces persist for historical reasons since they are a well-established technology, integrated in current CAD software. On the other hand, this representation distinguishes itself by its efficiency, precision, and simplicity. Trimming problems are hidden from the user who usually designs a model with the help of black box algorithms. Isogeometric analysis and adaptive manufacturing may lead to new developments in CAGD, but trimmed models are the state of the art and changes will certainly take time. It is not clear to the authors whether trimmed NURBS or other techniques like T-splines and subdivision surfaces will triumph in the future, but it is good to see the competition. At this juncture, however, trimmed NURBS seem to be the dominant technology of engineering design.

4 Exchange Standards

At the beginning of this section, general considerations for exchanging data between different computer software systems is discussed. Next, the most popular neutral exchange standards, i.e., the Initial Graphics Exchange Specification (IGES) and the Standard for the Exchange of Product Model Data (STEP), are briefly introduced and compared. Finally, this section closes with some concluding remarks.

4.1 General Considerations

In modern CAD systems, parameters and constraints govern the design of a model, rather than the definition of specific control points. Further essential components are local features and the construction history. All these various factors are referred to as *design intent* [162]. Each software has its own native data structure to keep track of the geometry, the topology, and the design intent of its models. Thus, a translation process is required when information is exchanged between systems with different native structures. The conversion of formats may seem like an easy task, but it is in fact very complicated. Usually, there is no

[8] Regular internal vertices have four incident edges, also referred to as valency $k = 4$. All other settings, i.e., $k \neq 4$, are denoted as extraordinary vertices.

direct mapping from one format to another. This holds true in particular for information related to the design intent since there are no canonical guidelines for its representation. Consequently, the exchange of the complete data of a CAD model between different systems is scarcely possible, especially when the systems are designed for different purposes. In most cases, only the geometric information of the final object is transferred.

This interoperability issue has been investigated in a study focusing on the US automotive supply chain [294]. The following possible solutions have been discussed: (i) standardization on a single system, (ii) point-to-point translation, and (iii) neutral format translation.

In case of a single system standardization the same native format is used for all processes, e.g., design and analysis. The main advantage is that the compatibility of the model data is assured since no translation is required. However, this approach implies the restriction to a single system. Consequently, every part has to be adjusted to the developments of the dominant application of the software. Most importantly, translation problems can arise even within one system due to different software versions—just imagine you would like to open a PowerPoint presentation created 10 years ago.

The basic idea of point-to-point translation is to convert a native format of a system directly to a native format of another one. This concept works reasonably well for unambiguous data exchange tasks. Unfortunately, it is not always clear how a given information should be translated so that it is properly interpreted in another native format. In addition, a high degree of vendor cooperation is necessary in order to develop a direct translator. Similar to the previous strategy, direct translators have to be rewritten for each new system or perhaps even for new versions of the same software.

Neutral format translation is based on a common neutral format for the exchange of (geometric) data. This approach enables an independent development of various tools working on the same model. The minimization of dependencies simplifies the maintenance of each software and eventually leads to robust implementations since a clean code is designed to do one thing well, as noted by Stroustrup[9] [196]. Further, vendors are more willing to develop translators for neutral formats since it does not require the disclosure of proprietary code. This is beneficial since interpretation errors of the native format are most likely minimized when the conversion is provided by vendors themselves. An additional advantage of neutral formats is that they are ideally suited for long term storage of data. However, there are also a number of weaknesses. First of all, it is not possible to capture the design intent and thus, translation to a neutral format provides only a snapshot of the current geometric model. In general, every translation leads to loss of information and the quality of an exchanged model depends on the capability of the neutral format used.

In the context of isogeometric analysis, the minimal requirement is the accurate exchange of geometrical information of the final model. Topology is also essential to assess the connectivity between patches. The reconstruction of topological data based on edge comparison or related strategies is very cumbersome and extremely error-prone, especially in cases of trimmed geometries where edges only coincide within a certain tolerance as elaborated on in Sect. 3. The following two approaches are suggested: (i) direct extraction of topological data from CAD software by a point-to-point translation or (ii) using a neutral format that is able to cope with topological data. The former may be preferred if there is a cooperation with a CAD vendor and the developments focus on the specific product. However, neutral exchange formats will be discussed in the following because they are the most general and independent approaches. Despite their deficiencies, native formats seem to be the most sustainable solution.

4.2 Neutral Format Translators

Concepts for a common data exchange format emerged in the 1970s. These attempts were borne by a variety of partners from industry, academia, and government [101]. Based on the initiative of the CAD user community, in particular General Electric and Boeing, vendors agreed to create an American national standard for CAD data exchange. The final result was the first version of IGES [209]. IGES provided the technical groundwork to a more involved exchange format, namely STEP.

4.2.1 IGES

The name of this neutral exchange format already reveals its original purpose [101]:

- *Initial*[10] to suggest that it would not replace the work of the American National Standards Institute.
- *Graphics* not geometry, to acknowledge that academics may come up with superior mathematical descriptions.
- *Exchange* to suggest that it would not dictate how vendors must implement their native database.
- *Specification* to indicate that it is *not* imposed to be a standard.

[9] Bjarne Stroustrup is the inventor of the programming language C++.

[10] The word “interim” was used in the first draft.

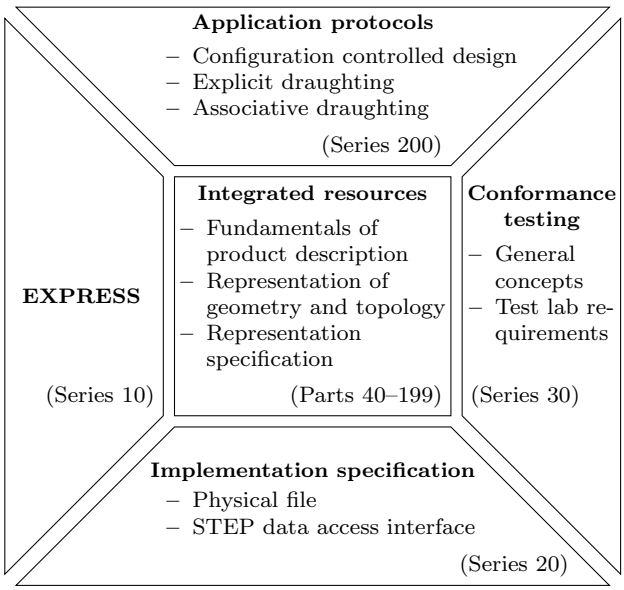


Fig. 32 Structure of STEP (re-execution of the original diagram [177])

IGES provided an important and very practical first solution to the exchange problem, resulting in a file format that is implemented in almost every CAD system. Regarding current literature on isogeometric analysis, it seems that IGES is still the preferred choice when it comes to the extraction of geometric information. Here, we will try to disprove this notion.

According to Goldstein et al. [101] and the studies cited within, the shortcomings of IGES can by summarized as follows: (i) it contains several ways to capture the same information leading to ambiguous interpretation, (ii) loss of information during exchange, (iii) development without rigorous technical discipline, (iv) restriction of exchange capabilities due to the compliance with earlier IGES versions, (v) it was developed as a method to exchange engineering drawings, but not designed for more sophisticated product data, (vi) vendors implemented only portions of IGES, and (vii) there is no mechanism for testing the translators. In addition, IGES is a national standard which may lead to translation problems if other than US software is used. Most importantly, IGES is a *stagnant* exchange format. The last official version of IGES, i.e., version 5.3 [155], was published more than 20 years ago in 1996.

Although IGES continues to be deployed in industry, its main legacy is the disclosure of several weaknesses of the neutral exchange concept, thereby enhancing new emerging standards. The most notable one is STEP, which provides a broader, more robust standard for the exchange of data [101].

Table 1 STEP stages

Stages	Names
PWI	Preliminary work item
NWI	New work item
AWI	Approved work item
WD	Working draft
CD	Committee draft
FCD	Final committee draft
DIS	Draft international standard
FDIS	Final draft international standard
PRF	Proof of new international standard
IS	International standard

4.2.2 STEP

Since 1984 the International Organization for Standardization (ISO) has been working on a standard for the exchange of *product data* and its first parts were published in 1994 [232]. The objective of this development effort—one of the largest ever undertaken by ISO—is the complete and unambiguous definition of a product throughout its entire life cycle, which is independent of any computer system [264]. Hence, the corresponding standard includes the exchange of CAD data, yet its scope is much broader.

STEP is the informal term for the standard officially denoted as ISO 10303. It is organized by an accumulation of various *parts* unified by a set of fundamental principals. These parts are referred to as ISO 10303-xxx, where xxx is determined by the part number. Each of them is separately published and has to pass several development phases summarized in Table 1. Each part is associated with one of the following *series*: (i) description methods, (ii) implementation specifications, (iii) conformance testing, (iv) generic integrated resources, (v) application integrated resources, (vi) application protocols. Figure 32 gives an overview of these various components of STEP. The description is given by the common formal specification language EXPRESS (Series 10) defining data types, entities, rules, functions, and so on [274]. It is not a programming language, but has an object-oriented flavor. The transfer of data is defined by the implementation specifications (Series 20). The exchange by a neutral ACSII file is addressed in Part 21, "clear text encoding the exchange structure." This STEP-file transfer is the most widely used data exchange form of STEP [264]. However, other approaches, like shared memory access, are covered by the series as well, see e.g., Part 22, "standard data access interface." Conformance tests provide the verification requirements (Series 30).

The most fundamental components of STEP are the integrated resources. They contain generic information such

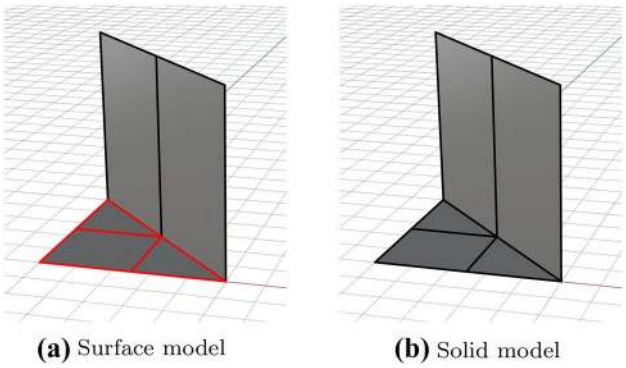


Fig. 33 Design model with the same geometry but different topology: **a** two independent trimmed surfaces and **b** connected surfaces by a Boolean operation. In **a**, the isocurves of the separated surfaces are displayed in *black* and *red*, respectively. (Color figure online)

Table 2 Entity types of the IGES example

Numbers	Names	Information
126	Rational B-spline curve	Geometry
128	Rational B-spline surface	Geometry
141	Boundary entity	Topology
143	Bounded surface entity	Topology

as geometric data and display attributes (Series 40) as well as further elementary units that are specialized for certain application areas (Series 100). For example, Part 42, "geometric and topological representations," focuses on the definition of geometric models in general, while Part 104, "finite element analysis," is devoted to applications in the context of FEA. These parts provide the entities needed to build application protocols denoted as APs (Series 200). They are the link to the needs of industry and other users. Their purpose is to interpret the STEP data in the context of a specific application which may be part of one or more stages of a life cycle of a particular product. Part 209, "multidisciplinary analysis and design," addresses engineering analysis. Each STEP application protocol is further subdivided into a set of conformation classes (CCs). These subsets must be completely implemented if a translator claims to be conform with the standard [233]. Hence, it is important to know what conformance classes are supported by a software system. This modular structure, with several APs and their CCs, may seem complex and daunting, but it gives users the necessary transparency of what can be expected of the data exchanged. Moreover, the complexity of the overall concept of STEP does not imply that it is difficult to use.

The downside of the broad scope of STEP is the large amount of detailed information which may seem overwhelming at first glance. In addition, ISO documents are not available for free, but can be purchased through the ISO homepage.[11] There are, however, helpful resources to start with: the US Consortium called Product Data Exchange using STEP (*PDES, Inc.*) provides several resources[12] like a handbook for a general introduction to STEP [264]. Further background articles are also released by the developers of *STEP tools, Inc.*[13] ISO permits EXPRESS listings to be distributed without copyright restrictions and several examples are given in the software's archive.[14]

One of the advantages of STEP is that it is more than just a specification for exchanging geometric information. It provides a complete product data format allowing the integration of business and technical data of an object, from design to analysis, manufacturing, sales, and service [294]. STEP is perfectly aligned with the spirit of isogeometric analysis, i.e., unifying fields. The most important feature of STEP is its extensibility. Efforts have been made to include the design intent into STEP [162, 233, 234]. Particularly interesting for isogeometric analysis is the specification of volumetric NURBS and local refinement in the next versions of Part 42 and other parts [284].

4.2.3 Comparative Example

In order to demonstrate the representation of trimmed geometries in IGES and STEP, an example of two intersecting planes is considered. A square $[0, 5]^2$ within the *xy*-plane is perpendicularly intersected along its diagonal by another plane surface as illustrated in Fig. 33. Thereby, the perpendicular patch is also trimmed into two halves by the square.

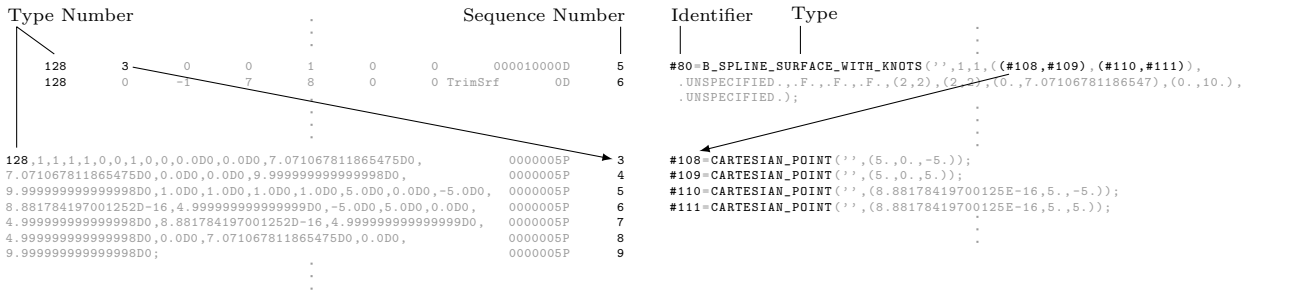
The model investigated has been constructed using the software `Rhinoceros` and the intersection has been computed in two different ways: using (i) the `trim`-command and (ii) the `Boolean`-command, respectively. Both schemes lead to the same geometry, yet the topology varies as indicated by the different highlighting of Fig. 33a and b. To be precise, the `trim`-command produces a *surface model* that consists of two independent trimmed surfaces, while the `Boolean`-command results in a *solid model* where the patches are connected. Both models have been exported to neutral exchange formats. The corresponding IGES and STEP files are provided in the Appendix. In the following, certain aspects of the exported files are discussed.

[11] http://www.iso.org, September 2016.

[12] https://www.pdesinc.org/ResourceIndex.html, September 2016.

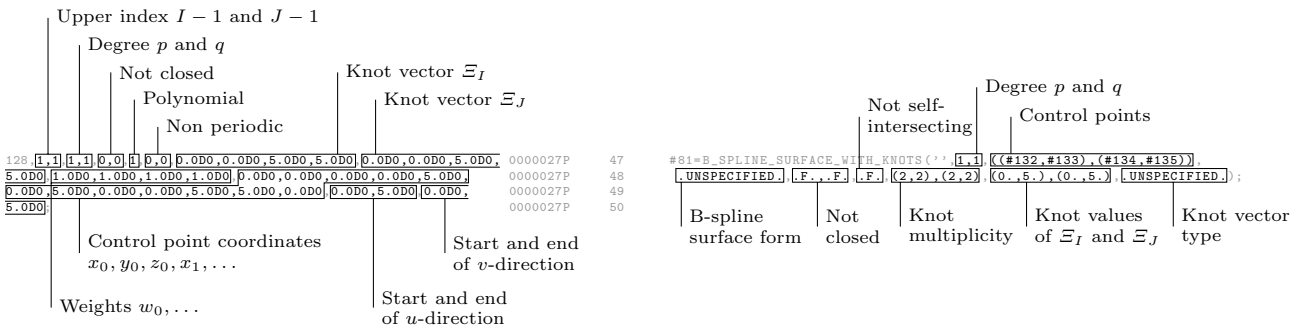
[13] http://www.steptools.com/library/standard/, September 2016

[14] http://www.steptools.com/sc4/archive/, September 2016.



(a) IGES **(b)** STEP

Fig. 34 Entity connection of the exchange formats. Pointers are indicated by *arrows*. The examples have been extracted from Files 1 and 3 of the Appendix, respectively



(a) IGES **(b)** STEP

Fig. 35 Descriptions of the surface model's regular square patch. The B-spline surface data has been extracted from Files 1 and 3 of the Appendix, respectively

Remark 3 The default setting of the `Rhinoceros` export has been chosen, i.e., AP213AutomotiveDesign, for the STEP examples. However, the elements discussed in this section are not affected by this choice.

4.2.3.1 File Structure The fixed ASCII file format[15] of IGES is structured by the subsequent sections: Start (S), Global (G), Directory Entry (D), Parameter Data (P), and Termination (T). The letters in the brackets label these distinct parts and they are shown in column 73 of every file. The Directory Entry and the Parameter Data is specified by *entities* which are associated with a unique type number. Table 2 lists the entities used in this example. The Directory Entries provide attribute information for each entity in an IGES file. Each entry is fixed in size and is specified by 20 fields. The first field contains the entity type and the second one points to the first line of the related Parameter Data record. This connection is shown for a rational B-spline surface in Fig. 34a. The Parameter Data, on the other hand, is free-formatted and it consists of a sequence of integer and real numbers starting with the entity type number.

STEP files are easy to read since the language used is based on an English-like syntax [274]. In general, an accumulation of entities pointing to each other shapes the structure of the exchange data. Lines specifying entities begin with the symbol #, followed by the unique identifier of the corresponding object. This identifier is used to connect various entities with each other as shown in Fig. 34b. Besides pointers, an entity may consists of integers, real numbers, Booleans (.F./.T.), and enumeration flags (e.g., .UNSPECIFIED.).

4.2.3.2 Surfaces Representation Both exchange formats provide the fundamental informations of B-spline patches, i.e., degree, knot vectors, and control points, together with auxiliary information. In case of IGES, a sequence of numbers separated by commas is used, while STEP additionally groups associated components using brackets. In Fig. 35, the representations of the regular square patch are compared. Note that knot vectors are specified by knot values with their multiplicity and that coordinates of control points are stored

[15] There exist also a compressed format for details see [155].

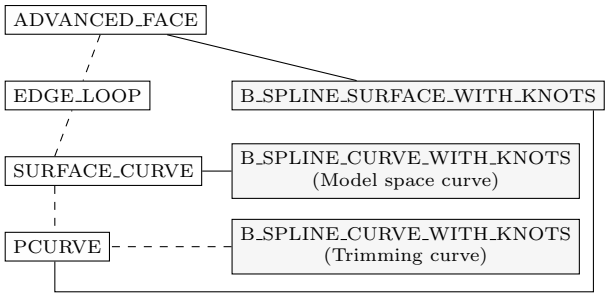


Fig. 36 Graph related to a trimmed surface in STEP. Entities that provide geometrical information are highlighted in *gray*. Intermediate nodes may be skipped which is indicated by *dashed lines*

within its own entity in case of STEP. Hence, error-prone comparisons of floating point numbers are avoided.

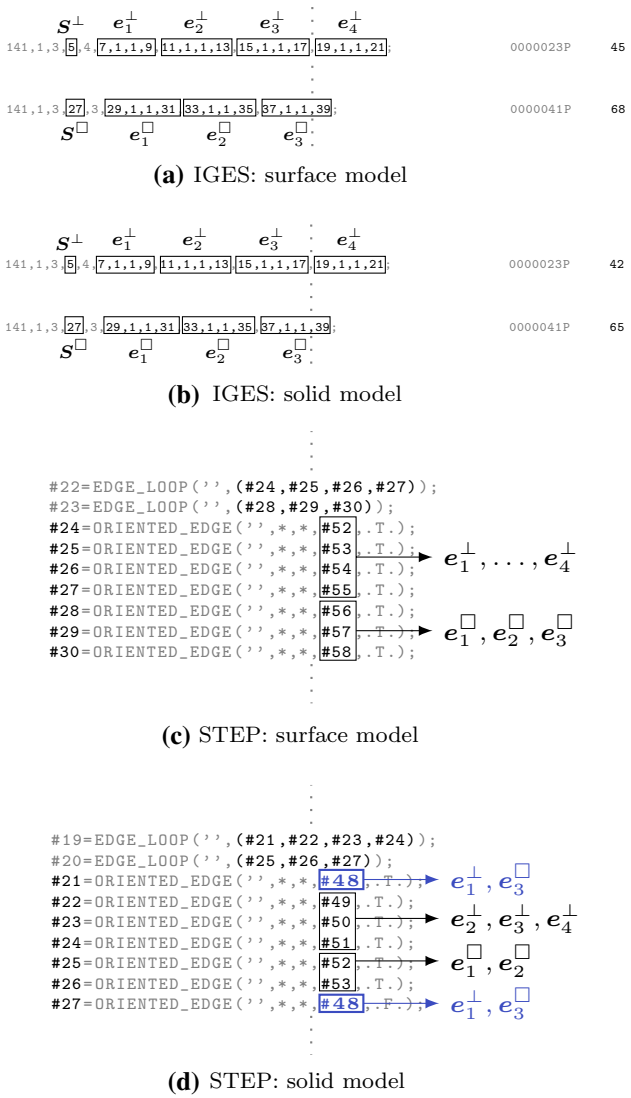
Regarding trimmed surfaces, the following information is provided in both exchange formats: (i) the regular surface, (ii) the related loops of trimming curves, and (iii) their counterparts in model space. All this information can be found in a single IGES entity, i.e., 141. In particular, the fourth and fifth number within the sequence of this entity define the reference to the regular surface and the number of related curves, respectively. These numbers are followed by arrays of the size 4. The first value of an array refers to model space curves while the last value points to trimming curves. The total number of arrays is determined by the number of related curves.

In case of STEP, the trimmed surface data is not coalesced in a single object, but it is embedded in a graph structure. Hence, the information is represented by various different entities which are linked together. The ADVANCED_FACE entity may be viewed as the starting point of the graph structure that specifies the trimmed patch. Figure 36 illustrates such a collection of entities. For the sake of clarity, some intermediate entities have been neglected as indicated by the dashed lines.

4.2.3.3 Topology So far, the specification of certain parts of a model has been addressed. Here, the differences between the exchange formats regarding an object's topological information is examined by comparing the output for the surface model and solid model shown in Fig. 33. The former is defined by two independent surfaces, while the latter is a single coherent manifold.

In the following the square patch in the xy-plane is denoted by $\boldsymbol{S}^{\square}$ and the perpendicular patch is referred to as $\boldsymbol{S}^{\perp}$. The corresponding edges of the model are labeled $\boldsymbol{e}_i^{\square}$ with $i = \{1, \dots, 3\}$ and $\boldsymbol{e}_j^{\perp}$ with $j = \{1, \dots, 4\}$, respectively.

The topology due to the STEP and IGES formats is compared in Fig. 37. To be precise, the provided edge loop data



(a) IGES: surface model

(b) IGES: solid model

(c) STEP: surface model

(d) STEP: solid model

Fig. 37 The STEP and IGES loop data of the model illustrated in Fig. 33. Entries referring to edges are labeled by $\boldsymbol{e}_i^{\square}$ and $\boldsymbol{e}_j^{\perp}$. The index numbers are not consistent and thus, may differ from sub-figure to sub-figure. In **d**, the highlighted pointer, i.e., #48, refers to the edge where the two surfaces $\boldsymbol{S}^{\square}$ and $\boldsymbol{S}^{\perp}$ intersect

is shown. Further details are neglected for the sake of brevity, but the entire files can be found in the Appendix.

Comparing Fig. 37a and b shows that IGES yields the *same* output for both models. In other words, the different topologies of them are not recognized. Note that the only values that differ are the sequence numbers of the entities which are completely independent from the actual object. In fact, the topological connection of $\boldsymbol{S}^{\square}$ and $\boldsymbol{S}^{\perp}$ is lost in case of the solid model, despite the simplicity of the example. That the solid model has been properly constructed is proven by the STEP output shown in Fig. 37d where the edges $\boldsymbol{e}_1^{\perp}$ and $\boldsymbol{e}_3^{\square}$ are joined in a single reference, i.e., #48.

The STEP data related to the surface model is illustrated in Fig. 37c.

4.3 Summary and Discussion

In Sect. 4.1, the problem of exchanging data between different systems is outlined from a general point of view. It is argued that the use of neutral exchange standards is the most comprehensive way for this task. Nevertheless, every mapping from one system to another may cause problems, especially with respect to the design intent of a model where no canonical representation exists. There is often no one-to-one translation from one format to another, which leaves room for (mis)interpretation. As a result, exchange is usually restricted to snapshots of an object's geometry. The transfer of topology data is also possible if (i) the design model is properly constructed as a coherent solid model and (ii) the neutral format is able to capture this information. It is apparent that the capability of the neutral standard applied is essential for the quality and success of the exchange. This has been demonstrated by a simple example given in Sect. 4.2.3 where IGES does not export the topology correctly.

STEP should generally be preferred as a neutral exchange format. According to Tassey et al. [294], STEP is superior to other translators because it

- addresses many types of data,
- incorporates a superset of elements common to all systems,
- supports special application needs, and
- provides for international exchanges.

In their paper, several studies are discussed in which STEP excels with respect to the quality of exchanging data of industrial examples. In addition, this standard is constantly developed and improved, e.g., by its enhancements for isogeometric analysis [284].

Theoretically, the broad scope and modular structure of STEP provides coverage of various application domains which are indicated by the application protocols and their conformance classes. However, this functionality has to be supported by the CAD vendors. Most vendors have chosen to implement only certain parts of STEP, i.e., some conformance classes of AP 203 and AP 214 [264]. It is not surprising that vendors seem to show little interest in neutral exchange formats, since their implementation slows down the development of the actual software and users become more independent from their products. Hence, it is likely that neutral file formats will always provide less information than the original model. Translation errors may be avoided if the needed data is extracted directly from the native format, but this requires vendor interaction and the restriction to a single software. This alternative is not very sustainable since a native format may become obsolete after a new software version is released.

5 Isogeometric Analysis of Trimmed Geometries

Isogeometric analysis of trimmed NURBS is an important research area, simply due to the omnipresence of such geometry representations. Integration of design and analysis can only be achieved if the simulation is able to cope with CAGD models that are actually used in the design process. Moreover, sound treatment of trimmed solid models is also an essential step for the derivation of volumetric representations.

Current attempts to integrate trimmed geometries into isogeometric analysis may be classified as *global* and *local* approaches. The latter uses the parameter space of the trimmed patch as background parameterization and the trimming curves determine the domain of interest, i.e., $\mathcal{A}^{\mathrm{v}}$, for the analysis. Knot spans that are cut by trimming curves require special attention during the simulation. In that sense, local approaches are closely related to fictitious domain methods,[16] see e.g., [124, 239, 252, 259]. Consequently, similar tasks have to be undertaken: (i) detection of trimmed elements, (ii) application of special integration schemes in these elements, and (iii) stabilization of the trimmed basis. CAGD models are not modified but the analysis has to deal with all the related robustness issues pointed out in Sect. 3.2.2. Global reconstruction, on the other hand, substitutes a trimmed surface by one or several regular patches which can be analyzed with regular integration rules. In other words, it is endeavored to fix the design model, before it is used in the downstream application, e.g., the simulation. These approaches are similar to remodeling schemes in CAGD presented in Sect. 3.4.1. Isogeometric analysis of subdivision surfaces, e.g., [59, 248, 309], and T-splines, e.g., [22, 262, 263, 321], may be included into the class of global reconstruction techniques. However, the discussion of the analysis of these representations is beyond the scope of this review.

Coupling of multiple patches is another issue that has to be addressed. Adjacent patches usually have nonmatching parameterizations and a robust treatment of tolerances is required to link the degrees of freedom along an intersection due to the gaps between trimmed surfaces and the missing link between their trimming curves. Local

[16] There are various names for fictitious domain methods such as embedded domain methods, finite cell methods, WEB-spline methods, and immersed boundary methods. The principle idea, however, is the same.

approach have to deal with the issue directly during the analysis, while global approach apply this crucial step beforehand during the remodeling phase. The coupling itself is usually performed by a weak coupling technique. Alternatively, some global schemes try to establish matching parameterizations during the reconstruction procedure. This allows an explicit coupling of patches and a better control of the continuity between adjacent patches [149]. It is also noteworthy that the coupling procedure may be neglected in certain simulation methods. For example, the boundary element method and the Nyström method do not require certain continuity between elements or patches, see e.g., [218, 223, 225, 319].

The following approaches for analyzing trimmed geometries have been applied to finite element and boundary element methods. The former focuses on shell analysis while the latter is used for volumetric B-Rep models. However, the basic concepts are not restricted to a specific simulation type since in both cases the treatment of trimmed surfaces is in the focus. It will be highlighted if a certain part explicitly applies for a specific simulation method.

The overview begins with a short historical note, which, to the best of the authors' knowledge is the first direct simulation with trimmed patches. Afterwards, the current state of research is reviewed in the Sects. 5.2 and 5.3 addressing global and local approaches, respectively.

5.1 The First Analysis of Trimmed Models

It is fascinating that the analysis of trimmed patches goes back to the genesis of trimmed patch formulations. In fact, Casale et al. [48, 50] presented an analysis of such geometries a few years after they had suggested one of the first trimmed solid model formulations [47, 49]. In particular, *trimmed patch boundary elements* had been proposed.

The basic idea of their approach is to employ the trimmed patch for the geometrical representation and to define an independent Lagrange interpolation over the tensor product surface for the description of the physical variables. This additional basis does not take the trimming curves into account. Thus, the nodes of the Lagrange interpolation may lie inside or outside the trimmed domain. This is emphasized by using the term *virtual nodes*. The analysis is performed by means of a collocated boundary element formulation, see e.g., [96], where all Lagrange nodes contribute to the system matrix. If a node is outside of the trimmed domain, the jump term coefficient[17] of the boundary integral equation is set to 1 since the node is not part of the object's boundary. Numerical integration is performed over a triangulation of the trimmed domain. These triangles are used to define integration regions only and do not contribute any degrees of freedom.

This concept has various deficiencies, but it consists of features that can be found in current approaches as well. For example, defining the geometrical mapping by the trimmed parameter space, but the physical fields by a different (spline) basis is employed in some global techniques presented later. There are also similarities to local schemes since the trimmed domain is treated like a background parameterization leading to special considerations regarding numerical integration and points that are not within the domain. Furthermore, the motivation for the application of the boundary element method was the same as today in isogeometric analysis, i.e., the potential of a direct analysis of B-Rep models without the need of generating a volumetric discretization.

5.2 Global Approaches

Global reconstruction schemes decompose trimmed surfaces into regular patches. The general concept is the same as presented in Sect. 3.4.1 in the context of CAGD. The distinguishing aspect is that the following strategies are aimed to provide analysis-suitable models.

5.2.1 Reconstruction by Ruled Surfaces

Trimming curves $\boldsymbol{C}^t$ may be used to define a mapping $\mathcal{X}_t$ such that a regular tensor product basis specifies the valid area $\mathcal{A}^{\mathrm{v}}$ of the corresponding trimmed patch, as proposed by Beer et al. [25]. To be precise, $\mathcal{X}_t$ is given by a linear interpolation between two opposing $\boldsymbol{C}^t_i$, $i = \{1, 2\}$. The geometrical mapping to the model space is performed by the original trimmed patch, hence the approach is also referred to as *double mapping method*.

The following assumptions are made for the sake of notational simplicity. Firstly, the regular basis functions are defined over a unit square, i.e., $s, t \in [0, 1]$. In addition, it is assumed that both trimming curves are specified within the same parameter range $\tilde{u} \in [a, b]$. Based on that, the intrinsic coordinate $\tilde{u}$ can be linked to the boundaries of the regular basis at $t = 0$ and $t = 1$ by the coordinate transformations $f(s)$ and $g(s)$. They are given by

$$\tilde{u} = f(s) = a + s(b - a), \tag{53}$$

and

$$\tilde{u} = g(s) = b + s(a - b). \tag{54}$$

These equations traverse the interval of $\tilde{u}$ in opposite directions, e.g., $f(0) = g(1) = a$, since one of the trimming curves has to be evaluated in reverse order. Finally, $\mathcal{X}_t$ is determined by

[17] Usually, the jump term coefficient depends on the geometric angle of the boundary at the point considered, see e.g., [96].

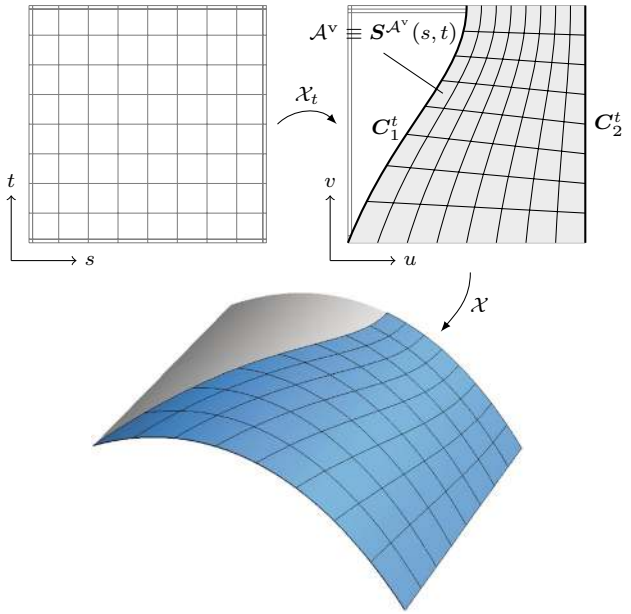


Fig. 38 Double mapping scheme to fit a regular tensor product surface to a trimmed patch. The first mapping $\mathcal{X}_t$ specifies the transformation to the valid area $\mathcal{A}^{\mathrm{v}}$ of the trimmed parameter space, while the geometric mapping is denoted by $\mathcal{X}$. The trimming curves $\boldsymbol{C}_i^t(\tilde{u})$, $i = \{1, 2\}$, are illustrated by *thick lines*

$$\boldsymbol{S}^{\mathcal{A}^{\mathrm{v}}}(s,\, t) = (1-t)\boldsymbol{C}_1^t(f(s)) + t\boldsymbol{C}_2^t(g(s)). \tag{55}$$

From a CAGD point of view, the mapping (55) is equivalent to the one of a ruled surface (26). The main difference is that the ruled surface is defined in the parameter space in this case. The geometric mapping $\mathcal{X}$, however, is still performed by the trimmed patch. Figure 38 summarizes the concept of the double mapping approach.

The main advantage of this approach is its simplicity and ease of implementation. However, there are various restrictions: first of all, the assumption that $\mathcal{A}^{\mathrm{v}}$ is governed by two opposing trimming curves limits the application to very specific trimming situations. Furthermore, the four-sided nature of $\mathcal{A}^{\mathrm{v}}$ is implied. Consequently, trimmed patches with more complex topology have to be decomposed by an additional preprocessing step. There is no control over the quality of the parameterization due to the mapping $\mathcal{X}_t$. Elements may become very distorted depending on the position of the trimming curves $\boldsymbol{C}_i^t$. Such a situation occurs for a triangular-shaped $\mathcal{A}^{\mathrm{v}}$, see Fig. 9. Since the parameterization is completely independent of the basis functions of the trimmed parameter space, the double mapping method works well for Bézier patches. An integration issue arises as soon as B-spline patches are considered. The problem is depict in Fig. 39. Note that the parameter lines of the geometry representation propagate through the elements defined by the mapped regular parameterization. Thus, integration

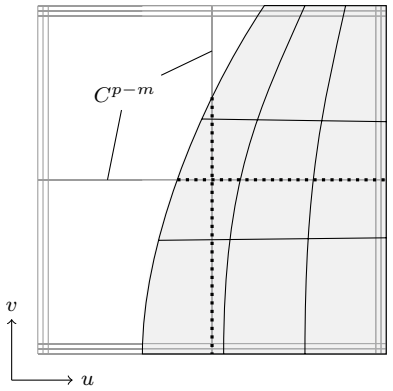


Fig. 39 Double mapping method for a B-spline patch. The *dotted lines* indicate parameter curves that are not C^∞-continuity within $\mathcal{A}^{\mathrm{v}}$

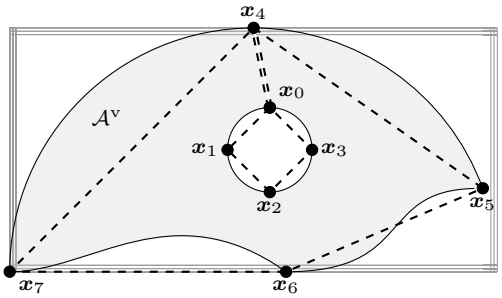
of the regular elements is not performed over a C^∞-continuous region. In order to get a proper distribution of quadrature points, the elements must be subdivided along the non-smooth edges. The specification of such regions is not straightforward. To conclude, the double mapping method is a simple solution for (Bézier) patches which had been trimmed during the design process, at least for ones that can be represented by a regular patch.

5.2.2 Reconstruction by Coons Patches

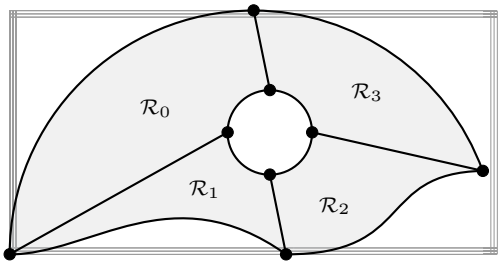
A natural extension of the previous method is to define the mapping $\mathcal{X}_t$ to a trimmed parameter space by means of Coons patches. In contrast to the ruled surface interpolation (55), $\mathcal{X}_t$ takes four boundary curves into account. Randrianarivony [237, 238] developed such an approach, which has been applied to wavelet Galerkin BEM in collaboration with Harbrecht [113]. Although they do not focus on isogeometric analysis per se, most of their techniques can be directly utilized: (i) decomposition of $\mathcal{A}^{\mathrm{v}}$ into several four-sided patches, (ii) identification if $\mathcal{X}_t$ is a diffeomorphism,[18] and (iii) construction of matching parameterizations of adjacent patches.

The first step of the decomposition procedure is to substitute the trimming curves $\boldsymbol{C}^t$ of each patch by a linear approximation $\boldsymbol{C}^l$. The vertices $\boldsymbol{x}$ of $\boldsymbol{C}^l$ are located along $\boldsymbol{C}^t$ as illustrated in Fig. 40a. $\boldsymbol{C}^l$ should be as coarse as possible since the number of vertices determines the number of patches that decompose $\mathcal{A}^{\mathrm{v}}$. As initial approximation, the endpoints of the trimming curves may be used. However, $\boldsymbol{C}^l$ has to be fine enough to resolve the topology of the trimmed patch, e.g., lines of exterior loops may not intersect ones of interior loops. In order to get a single polygon representing $\mathcal{A}^{\mathrm{v}}$, interior loops are connected to exterior

[18] A diffeomorphism is a C^∞ mapping with a C^∞ inverse.



(a) Approximation of the trimmed domain



(b) Decomposition

Fig. 40 Decomposing of a trimmed domain $\mathcal{A}^{\mathrm{v}}$ into **b** regular four-sided patches $\mathcal{R}_i$. In **a**, the trimming curves are continuous whereas the linear approximation is illustrated by *dashed lines*. Further, vertex $\boldsymbol{x}_0$ and $\boldsymbol{x}_4$ are connected by a double edge

loops by so-called double edges. The vertices of the resulting polygon are basis for the decomposition of $\mathcal{A}^{\mathrm{v}}$ into a set of quadrilaterals $\mathcal{R}$. Therefore, it is important that the total number of vertices $\boldsymbol{x}$ is even. In the next step, the straight boundary curves of $\boldsymbol{C}^l$ are replaced by the complementary portions of $\boldsymbol{C}^t$. An example of a decomposition is shown in Fig. 40b. Due to this procedure, the following problems may arise. The most obvious one is that the curved boundary may intersect an internal edge. In addition, sharp corners become degenerated points if the corresponding $\boldsymbol{x}$ is smoother than C^0. As a result, no diffeomorphism for this region can be found [237]. Finally, it is not assured that a Coons patch interpolation is regular. Such problems arise particularly in case of non-convex domains. An example of a non-regular Coons patch where the parametric lines of the surface overspill is shown in Fig. 41. A remedy to the mentioned issues is local refinement of $\boldsymbol{C}^l$ or the affected $\mathcal{R}_i$. The detection of the first two problems is straightforward, but determination of a Coons patch's regularity requires a more detailed discussion.

The following identification procedure assumes that the Coons patch interpolation (35) is planar and described by boundary curves given in Bézier form, i.e.,

$$\boldsymbol{C}_i(u) = \sum_{k=0}^{p} B_{k,p}(u)\boldsymbol{c}_k^i, \quad i = 1,\, 2, \quad u \in [0,\, 1], \tag{56}$$

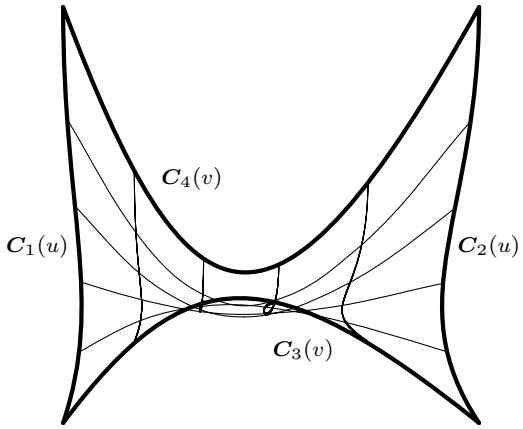


Fig. 41 Example of a planar non-regular Coons patch where isocurves overlap

$$\boldsymbol{C}_j(v) = \sum_{k=0}^{p} B_{k,p}(v)\boldsymbol{c}_k^j, \quad j = 3,\, 4, \quad v \in [0,\, 1]. \tag{57}$$

The interpolation function f_1 is also described as a Bézier polynomial

$$f_1(s) = \sum_{k=0}^{p} B_{k,p}(s)\psi_k = 1 - f_0(s), \quad s \in [0,\, 1]. \tag{58}$$

For the determination of the regularity, the factors μ, τ, and α are specified by

$$\mu := \max\left\{|f_1'(s)|\colon s \in [0,\, 1]\right\}, \tag{59}$$

$$\tau := \min\left\{\tau_{k\ell}^{ij}\right\}, \quad k,\, \ell = 0, \ldots, p, \tag{60}$$

$$\alpha := \max\left\{\alpha_1,\, \alpha_2\right\}. \tag{61}$$

The indices i and j are defined as above and the factors in Eqs. (60) and (61) are determined by

$$\tau_{k\ell}^{ij} := p^2 \det\left[\boldsymbol{c}_{k+1}^i - \boldsymbol{c}_k^i,\, \boldsymbol{c}_{\ell+1}^j - \boldsymbol{c}_\ell^j\right], \tag{62}$$

$$\alpha_1 := \max_{k=0,\ldots,p}\left\{\mu||\left(\boldsymbol{c}_k^4 - \boldsymbol{c}_k^3\right) + \psi_k\hat{c} + \left(\boldsymbol{c}_0^1 - \boldsymbol{c}_p^1\right)||\right\}, \tag{63}$$

$$\alpha_2 := \max_{k=0,\ldots,p}\left\{\mu||\left(\boldsymbol{c}_k^2 - \boldsymbol{c}_k^1\right) + \psi_k\hat{c} + \left(\boldsymbol{c}_0^1 - \boldsymbol{c}_0^2\right)||\right\}, \tag{64}$$

with $\hat{c} = (\boldsymbol{c}_0^2 - \boldsymbol{c}_p^2 + \boldsymbol{c}_p^1 - \boldsymbol{c}_0^1)$. Finally, the constant β is defined such that

$$p\left\|\psi_k\left(\boldsymbol{c}_{\ell+1}^2 - \boldsymbol{c}_\ell^2 + \boldsymbol{c}_\ell^1 - \boldsymbol{c}_{\ell+1}^1\right) + \left(\boldsymbol{c}_{\ell+1}^1 - \boldsymbol{c}_\ell^1\right)\right\| \leq \beta, \tag{65}$$

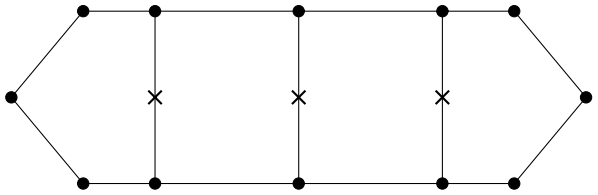


Fig. 42 Converting a pair of odd faces to even ones. *Circles* indicate the initial vertices of the faces and *crosses* mark those vertices that had been added to obtain faces with an even number of vertices

$$p\left\|\psi_k\left(\boldsymbol{c}^4_{\ell+1}-\boldsymbol{c}^4_{\ell}+\boldsymbol{c}^3_{\ell}-\boldsymbol{c}^3_{\ell+1}\right)+\left(\boldsymbol{c}^3_{\ell+1}-\boldsymbol{c}^3_{\ell}\right)\right\|\leq\beta, \tag{66}$$

for all $\ell = 0, \dots, p-1$ and $k = 0, \dots, p$. Based on these definitions, the Coons patch mapping is regular if

$$2\alpha\beta+\alpha^2<\tau \quad \text{and} \quad \tau>0. \tag{67}$$

These are only sufficient conditions. If they are not fulfilled a subdivision procedure is employed. Therefore, another sufficient condition is derived. Assuming that a Coons patch is represented as Bézier surface with control points $\boldsymbol{c}^C_{i,j}$, the Jacobian can also be described as a Bézier function of degree $2p$ with the control points

$$\boldsymbol{c}^J_{m,n}=\sum_{\substack{i+k=m\\ j+\ell=n}} C(i,j,k,\ell)\frac{\binom{p}{i}\binom{p}{k}}{\binom{2p}{i+k}}\frac{\binom{p}{j}\binom{p}{\ell}}{\binom{2p}{j+\ell}}, \tag{68}$$

with $m, n = 0, \dots, 2p$ and the coefficients

$$\begin{aligned} C(i,j,k,\ell) := \frac{\ell}{p}\Bigg[&\frac{i}{p}D(i-1,j,k,\ell-1)\\ &+\left(1-\frac{i}{p}\right)D(i,j,k,\ell-1)\Bigg]\\ &+\left(1-\frac{\ell}{p}\right)\Bigg[\frac{i}{p}D(i-1,j,k,\ell)\\ &+\left(1-\frac{i}{p}\right)D(i,j,k,\ell)\Bigg], \end{aligned} \tag{69}$$

where

$$D(i,j,k,\ell) := p^2\det\left[\boldsymbol{c}^C_{i+1,j}-\boldsymbol{c}^C_{i,j},\ \boldsymbol{c}^C_{k,\ell+1}-\boldsymbol{c}^C_{k,\ell}\right]. \tag{70}$$

If the coefficients $\boldsymbol{c}^J_{m,n}$ have the same sign the Coons patch mapping is a diffeomorphism. In case of unequal signs, the patch is adaptively subdivided and for each sub-patch the coefficients $\boldsymbol{c}^{J_{sub}}_{m,n}$ are computed. The procedure stops as soon as the signs of $\boldsymbol{c}^{J_{sub}}_{m,n}$ do not change within every sub-patch. The overall Coons patch is not regular, if it consists of sub-patches with different signs. It is emphasized that the subdivision is only performed to determine the regularity of the mapping. The corresponding proofs and more information can be found in [113, 237].

In case of multiple patches, a matching parameterization between adjacent surfaces is sought. The connectivity is described by a graph. During the decomposition procedure, the polygon vertices of the adjacent patches $\boldsymbol{S}_i$ and $\boldsymbol{S}_j$ are computed so that

$$\mathcal{X}_i(\boldsymbol{x}^i_k)=\mathcal{X}_j(\boldsymbol{x}^j_\ell), \tag{71}$$

where $\boldsymbol{x}^i_k$ and $\boldsymbol{x}^j_\ell$ are the vertices along the common edge and $\mathcal{X}$ denotes the geometrical mapping (23). If a vertex $\boldsymbol{x}^i_k$ is added to obtain an even-numbered approximation $\boldsymbol{C}^l$, a corresponding $\boldsymbol{x}^j_\ell$ needs to be added in the adjacent patch. Thus, odd faces can be converted to even ones only in pairs as illustrated in Fig. 42. It should be noted that the inserted vertices propagate through faces which are even already. In order to minimize the affected faces, the shortest path connecting two faces is computed by the application of Dijkstra's algorithm [72] to the connectivity graph. In addition to matching vertices, trimming curves $\boldsymbol{C}^t \in [a, b]$ are parameterized by means of the chord length of the corresponding intersection curve in model space. The trimming curve segment of quadrilaterals $\mathcal{R}_i$ is initially defined by $\bar{\boldsymbol{C}}^t_i(t\cdot a_i+(1-t)b_i)\in[a_i,b_i]\subset[a,b]$. The new representation $\hat{\boldsymbol{C}}^t_i$ is given by

$$\hat{\boldsymbol{C}}^t_i := \bar{\boldsymbol{C}}^t_i\circ\phi_i, \quad \phi_i=\left(\lambda_i\right)^{-1}, \tag{72}$$

with ϕ_i denoting the inverse of the length function

$$\lambda_i(t) := \int_a^t \left\|\frac{d(\mathcal{X}\circ\bar{\boldsymbol{C}}^t_i)}{dt}(\theta)\right\| \mathrm{d}\theta. \tag{73}$$

Hence, the images of the trimming curve $\hat{\boldsymbol{C}}^t$ of adjacent patches match at the same parametric values, i.e., the same chord length. This procedure is applied before the Coons patch construction. For details on the computation of the reparameterization the interested reader is referred to [238].

In contrast to the approach described in the previous subsection, this reconstruction scheme addresses the partitioning of trimmed domains into several four-sided regions, the regularity of these regions, and the connection of adjacent patches. Yet, some aspects are unresolved. For instance, the compatibility condition (71) implies that trimming curves coincide which is usually not the case as discussed in Sect. 3. Since the trimming curves do not describe the same curve in model space, the chord length parameterization may lead to diverse results. Thus, a robust implementation is required that treats the tolerances involved. As

pointed out in Randrianarivony's thesis [237], the reconstruction algorithm might fail if the inaccuracies of CAGD models are too large. User interaction is required to find an adequate tolerance threshold. IGES data has been used as exchange format in the related publications, which makes the tolerance treatment even more delicate since the topology of the objects has to be reconstructed as elaborated in Sect. 4. Finally, the problem of integrating over C^∞-continuous regions in case of B-spline patches is not addressed.

5.2.3 Reconstruction by Isocurves

Recently, Urick [298] presented a reconstruction approach based on isocurves (25). In contrast to the previous schemes, the trimmed patch is replaced by a new parameterization and a new set of control points. The overall procedure consists of several steps including (i) topology detection, (ii) parameter space analysis and determination of knot vector superset, (iii) reparameterization of trimmed parameter spaces, (iv) computation of corresponding control points, and (v) the treatment of multiple trimming curves.

In order to identify the topology of the trimmed domain $\mathcal{A}^{\mathrm{v}}$, characteristic points $\boldsymbol{x}$ of the trimming curves $\boldsymbol{C}^t$ are determined. The points considered are summarized in Table 3. They represent characteristic points commonly used in surface-to-surface intersection schemes (i.e., types 0, 2, and 3) [221], along with an additional point previously not utilized (type 1). The main purpose of this classification is to detect portions γ of a trimming curve that are associated to either the u-direction, i.e., γ^u, or the v-direction, i.e., γ^v, of the parameter space. With this in mind, the most significant points are those of types 0 and 1, because they indicate a possible transition from γ^u to γ^v. Each γ together with its opposing edge of the parameter space specifies a four-sided regions $\mathcal{R}$. An example of a segmentation of $\mathcal{A}^{\mathrm{v}}$ is illustrated in Fig. 43. Note that not every characteristic point of type 1, i.e., $\boldsymbol{x}^1$, yields a new portion. Hence, the sequence of characteristic points has to be examined rather than the classification of individual points.

Once all reconstruction regions $\mathcal{R}$ are detected, the parameterization of adjacent patches is aligned. The following knot cross-seeding procedure establishes a one-to-one relation of points along the intersection curve $\hat{\boldsymbol{C}}(s)$ in model space and the related trimming curves of the surfaces $\boldsymbol{S}_i(u, v)$. Firstly, $\hat{\boldsymbol{C}}(s)$ is refined so that it is defined by Bézier segments, i.e., the multiplicity of all knots is equal to the curve's degree. Furthermore, each $\boldsymbol{S}_i(u, v)$ is subjected to knot insertion in the u-direction and v-direction at their characteristic points $\boldsymbol{x}$ of γ^u and γ^v portions, respectively. Thus, the knot vectors of the surfaces incorporate the locations of all $\boldsymbol{x}$. In the next step, the knot information is exchanged across the different objects involved in order to define a knot vector superset. During this process, new Bézier segments are introduced to $\hat{\boldsymbol{C}}(s)$. In particular, knots are added at the parametric values $\hat{s}$ that correspond to the knots of $\boldsymbol{S}_i(u, v)$. The values $\hat{s}$ are determined by minimizing the distance of $\hat{\boldsymbol{C}}(s)$ to isocurves $\boldsymbol{C}^{iso}(u)$ and $\boldsymbol{C}^{iso}(v)$ of each $\boldsymbol{S}_i(u, v)$ and the fixed parameters of these isocurves are determined by the knot values of the surfaces. As a result, the refined intersection curve and its superset knot vector reflect the knots and topological characteristics of itself, the related surfaces, and their trimming curves. Finally, this information is passed on to the surfaces, i.e., all $\boldsymbol{S}_i(u, v)$ are refined at the interior knots of $\hat{\boldsymbol{C}}(s)$ including the knots of the adjacent $\boldsymbol{S}_j(u, v)$, $j \neq i$. This is done by minimizing the distance between the points of $\hat{\boldsymbol{C}}(s)$ and $\boldsymbol{S}_i(u, v)$. The exchange of knot data is necessary in order to guarantee that patches are connected along their intersection after the reconstruction.

Reparameterization is required to obtain a conforming basis for each four-sided region $\mathcal{R}$. Suppose $\mathcal{R}$ is related to a γ^v-portion[19] of a trimming curve, then it is described by a set of isocurves $\{\boldsymbol{C}_k^{iso}(u)\}_{k=1}^K$ along fixed parameter values

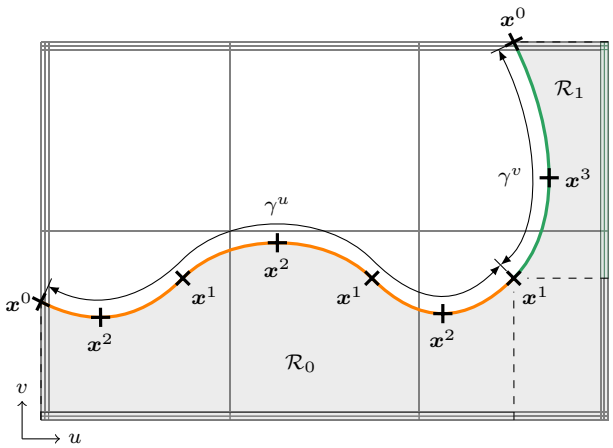


Fig. 43 Determination of the trimming curve portions γ^u and γ^v which are associated to the parametric direction u and v, respectively. The boundary of the related regions $\mathcal{R}_i$ within the trimmed domain are indicated by *dashed lines*. Characteristic points $\boldsymbol{x}$ are marked by *crosses* which correspond to the sloped of the curve. The point type is denoted by the related superscript

Table 3 Characteristic points $\boldsymbol{x}$ of a trimming curve

Types	Description
0	End points, kinks, and cusps
1	Slope relative to u is 1 or -1
2	Slope relative to u is 0
3	Slope relative to u is ∞

[19] Regions related to a γ^u-portion are treated in an analogous manner.

s_k^{iso}. Note that the values s_k^{iso} correspond to the parameterization of the intersection curve $\hat{\boldsymbol{C}}(s)$ rather than the v-direction of the trimmed surface. Thus, the reparameterized region $\tilde{\mathcal{R}}$ will be conformal with $\hat{\boldsymbol{C}}(s)$ and the reparameterized counterpart of an adjacent surface will be conformal as well. Due to the cross-seeding process, γ^v is linked to at least one Bézier segment of $\hat{\boldsymbol{C}}(s)$. The positions s_k^{iso} of the isocurves $\boldsymbol{C}_k^{iso}(u)$ are determined by the endpoints and Greville abscissae of these Bézier segments. The corresponding parametric values in the u-direction are labeled $\hat{u}_k$ and represent the locations where the distance of $\boldsymbol{C}_k^{iso}(u)$ is minimal to $\hat{\boldsymbol{C}}(s)$. Knot insertion at $\hat{u}_k$ is applied to extract the part of $\boldsymbol{C}_k^{iso}(u)$ that is within the domain of interest, i.e., the current four-sided region $\mathcal{R}$. The values $\hat{u}_k$ vary for each $\boldsymbol{C}_k^{iso}(u)$ since they are distributed close to the trimming curves. In other words, parameter intervals of the isocurves within $\mathcal{R}$ do not match in general. To overcome this issue, all $\boldsymbol{C}_k^{iso}(u)$ are reparameterized to be specified by the *same* basis.

The simplest way to establish the reparameterization is to use a linear coordinate transformation so that all isocurves are defined over a common range, combined with a subsequent accumulation of the shifted interior knots of each knot vector. However, this technique yields a large number of basis function since interior knots of isocurves with different initial intervals do not coincide after the transformation. Furthermore, the size of the resulting knot spans may vary excessively because the alteration of shifted interior knots can be arbitrarily small.

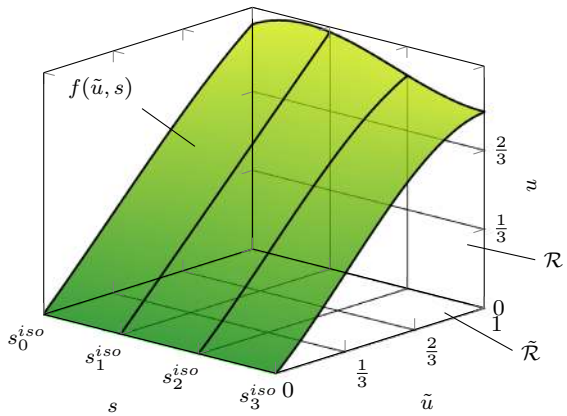
Hence, a nonlinear reparameterization is preferred. A set of functions $\{f_k(\tilde{u})\}_{k=1}^K$ is sought that maps the corresponding $\boldsymbol{C}_k^{iso}(u) \in [a_k, b_k]$ to a common range $\boldsymbol{C}_k^{iso}(\tilde{u}) \in [c, d]$ without modifying interior knots. These functions can be represented by univariate splines with scalar coefficients c_i^k, i.e.,

$$f_k(\tilde{u}) = \sum_{i=0}^{I-1} B_{i,q}(\tilde{u}) c_i^k, \quad q > 1, \quad k = 1, \dots, K. \tag{74}$$

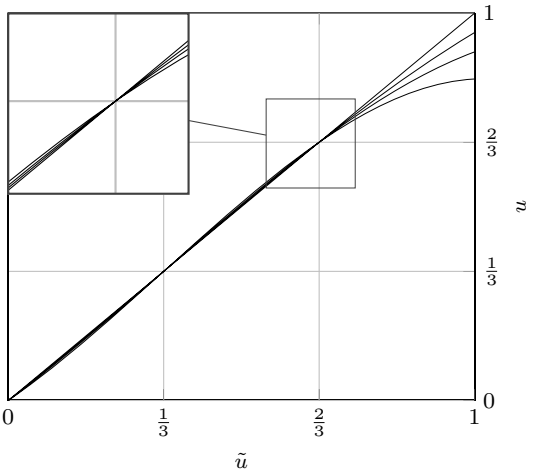
Note that the linear coordinate transformation is a special case of this formulation where the degree of the function is set to $q = 1$. The composite of an isocurve $\boldsymbol{C}_k^{iso}(u)$ and its $f_k(\tilde{u})$ determines the reparameterization

$$\boldsymbol{C}_k^{iso}(\tilde{u}) = \boldsymbol{C}_k^{iso}(f_k(\tilde{u})), \quad \tilde{u} \in [c, d]. \tag{75}$$

The degree of the resulting curve is given by $\tilde{p} = pq$ where p refers to the original degree of the isocurve. If the longest isocurve is taken as target parameterization, it can be adjusted by using conventional degree elevation to $\tilde{p}$. The other curves are subjected to a nonlinear reparameterization based on their $f_k(\tilde{u})$. For technical details on nonlinear reparameterization of curves the interested reader is referred to the textbook [230].



(a) Reparameterization surface



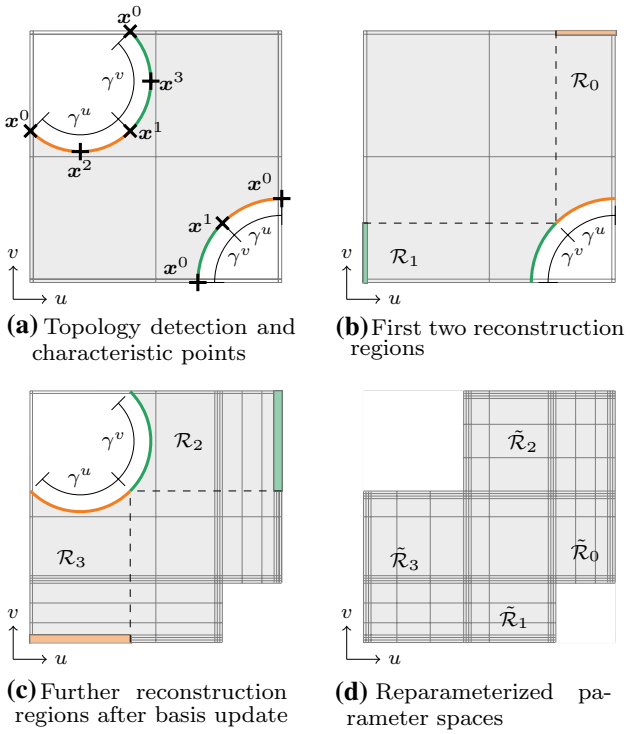
(b) Isocurves

Fig. 44 Reparameterization of a trimmed parameter space of a bicubic Bézier patch: **a** surface $f(\tilde{u}, s)$ that reparameterizes the trimmed parameter space u to a regular one $\tilde{u}$ indicated by the vertical and plane grid, respectively. *Lines* on the surface mark the associated isocurve along s_k^{iso}. **b** Profile of the isocurves $f_k(\tilde{u})$

It remains to find a way to coordinate the individual $f_k(\tilde{u})$ to obtain a *global* reparameterization for the whole reconstruction domain $\mathcal{R}$ that yields a new valid tensor product parameter space $\tilde{\mathcal{R}}$. The key idea is to represent the global transformation as a spline surface $f(\tilde{u}, s)$. This surface includes all $f_k(\tilde{u})$ as isocurves, i.e., $f(\tilde{u}, s_k^{iso}) = f_k(\tilde{u})$. The bivariate reparameterization is given by

$$f(\tilde{u}, s) = \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} B_{i,\tilde{p}}(\tilde{u}) B_{j,p_s}(s) c_{i,j}, \tag{76}$$

with the degree p_s of the intersection curve segment and a grid of scalar control coefficients $c_{i,j}$. If the degree in the v -direction of the trimmed surface varies from p_s, the degree of the segment may be adjusted by means of degree elevation. Equation (76) can be represented as a non-parametric



(a) Topology detection and characteristic points

(b) First two reconstruction regions

(c) Further reconstruction regions after basis update

(d) Reparameterized parameter spaces

Fig. 45 Successive evolution of the isocurve reconstruction of a quadratic trimmed patch using a quadratic reconstruction function

surface by linking the coefficients $c_{i,j}$ to their Greville abscissae (22).

The corresponding control points are defined as

$$c_{i,j} = \begin{bmatrix} (\tilde{u}_{i+1} + \tilde{u}_{i+2} + \cdots + \tilde{u}_{i+\tilde{p}})/\tilde{p} \\ (s_{j+1} + s_{j+2} + \cdots + s_{j+p_s})/p_s \\ c_{i,j} \end{bmatrix}. \tag{77}$$

The coefficients $c_{i,j}$ can be defined by the user as long as the following restrictions are met:

(i) The function $f(\tilde{u}, s)$ must be *strictly monotonic* in the $\tilde{u}$-direction so that intervals do not overlap.
(ii) The spline surface must employ the same target knot vector $\tilde{\Xi}$ in the $\tilde{u}$-direction.
(iii) Each knot value u_i of the initial knot vectors must be mapped to a distinct $\tilde{u}_i \in \tilde{\Xi}$ for all isocurves, i.e., $u_i = f(\tilde{u}_i, s_k^{iso})$, $k = 1, \ldots, K$.

An illustration of such a bivariate reparameterization function $f(\tilde{u}, s)$ is provided in Fig. 44a and the corresponding isocurves are shown in Fig. 44b. It should be noted that the parameter space spanned by the $\tilde{u}$-axis and s-axis is defined by straight parameter lines only, in contrast to the original basis spanned by the u-axis and s-axis. It is emphasized that the graphs in Fig. 44b intersect at the common interior knots $\tilde{u}_i = u_i = \left\{\frac{1}{3}, \frac{2}{3}\right\}$.

The final step of the reconstruction scheme is to determine the control points of the reparameterized regions $\tilde{\mathcal{R}}$. Therefore, we recall the specification of the control points $\tilde{\boldsymbol{c}}_i^k$ of isocurves $\boldsymbol{C}_k^{iso}(u)$ as a weighted combination of surface control points $\boldsymbol{c}_{i,j}$

$$\tilde{\boldsymbol{c}}_i^k = \sum_{j=0}^{J-1} B_{j,q}(s_k^{iso}) \boldsymbol{c}_{i,j}, \quad k = 1, \ldots, K. \tag{78}$$

Isocurves have been introduced at the Greville abscissae of the Bézier segments along the reconstruction boundary γ^v. Hence, the number of isocurves is equal to the number of unknowns, i.e., $J = K$, and the control points $\boldsymbol{c}_{i,j}$ can be computed based on the known isocurve control points $\tilde{\boldsymbol{c}}_i^k$ by inverting the system of equations (78). The control points along the boundary γ^v are already known beforehand, i.e., the control points of $\hat{\boldsymbol{C}}(s)$, and do not need to be computed explicitly.

It is quite astonishing that the procedure described remains the same when multiple trimming curves are involved. Instead of assessing the topology of all trimming curves at once, the trimming curve or more precisely each γ is processed successively and independently of each other. In fact, it does not matter if the portions γ originate from one or several trimming curves. After each reparameterization the parameter space is updated and the next region is addressed. The iterative evolution of the reconstructed regions $\tilde{\mathcal{R}}$ is displayed in Fig. 45. To be clear, the regions $\mathcal{R}_0$ and $\mathcal{R}_1$ shown in Fig. 45b and the regions $\mathcal{R}_2$ and $\mathcal{R}_3$ displayed in Fig. 45c are not constructed at the same time.

The final outcome of the reconstruction is a new set of patches with aligned parameter spaces that share the control point of their intersection curves. Thus, the reconstructed object is watertight. It is emphasized that this holds true even for non-manifolds. These benefits come at the price of an alteration of the initial geometry and an increase of the degrees of freedom. Since the concept has been presented just recently, there are several open research topics to explore. For instance, an estimation of the geometrical error introduced with respect to the degree of the reparameterization function and the number of isocurves would be of great interest. This could be the basis for an optimization procedure for the definition of the reparameterization function. Another topic might be the quality of the resulting elements in model space, especially at the transitions from γ^u to γ^v regions.

We close the discussion of the isocurve reconstruction approach with some application remarks. Firstly, the concept can also be applied locally. In this paper it is focused on the tensor product case where refinement propagates

through the whole domain for the sake of simplicity. A locally reconstructed parameter space may be represented by any local basis like T-splines, hierarchical B-splines, or LR-splines. Secondly, the degree of the resulting patches may become large, depending on the degree of the reparameterization function. It might be beneficial to apply a degree reduction technique after the reconstruction, but this introduces additional approximation errors. Finally, the intersection curves should have a good parameterization since they play an essential role during the reconstruction. Therefore, it might be advisable to reparameterize the intersection curve, e.g., by its chord length, at the beginning of the overall procedure.

5.2.4 Reconstruction by Triangular Bézier Splines

Another recent attempt has been proposed by Xia and Qian [314]. They employ triangular Bézier patches (36) to convert trimmed models to watertight representations. The convergence behavior of these splines has been assessed by these authors and co-workers in [315]. The conversion involves the following steps: (i) subdivision of all surfaces into tensor product Bézier patches, (ii) exact representation of non-trimmed patches by two Bézier triangles, (iii) knot cross-seeding between adjacent patches, (iv) approximation of the region along the trimming curve using Bézier triangles, and (v) substitution of the resulting control points of the approximate trimming curve by corresponding control points of the intersection curve in model space.

The first step can be easily accomplished by means of knot insertion. The second one is performed following Goldman and Filip [100]. In particular, a non-trimmed tensor product patch with control points $\boldsymbol{c}^{\square}_{m,n}$ can be converted to two triangular Bézier patches by

$$\boldsymbol{c}^{\triangle}_{i,j,k} = \frac{1}{\binom{p+q}{q}} \sum_{m=0}^{i} \sum_{n=\max\{0,j-p+m\}}^{\min\{j,q-i+m\}} \boldsymbol{c}^{\square}_{m,n} \binom{i}{m} \binom{j}{n} \times \binom{p+q-i-j}{p+n-m-j}, \tag{79}$$

where $i+j+k = p+q$ and $\binom{\alpha}{\beta}$ are binomial coefficients defined as

$$\binom{\alpha}{\beta} := \frac{\alpha!}{(\alpha-\beta)!\beta!}. \tag{80}$$

Equation (79) yields the control points $\boldsymbol{c}^{\triangle}_{i,j,k}$ of one triangular patch using $\boldsymbol{c}^{\square}_{m,n}: 0 \leqslant m \leqslant p; 0 \leqslant n \leqslant q$. The control points of the other triangular patch are obtained by reversing the order of the original control points, i.e., $\boldsymbol{c}^{\square}_{p-m,q-n}: 0 \leqslant m \leqslant p; 0 \leqslant n \leqslant q$. The degree of the resulting patches is determined by $p+q$. It is emphasized that this transformation does not introduce an approximation error.

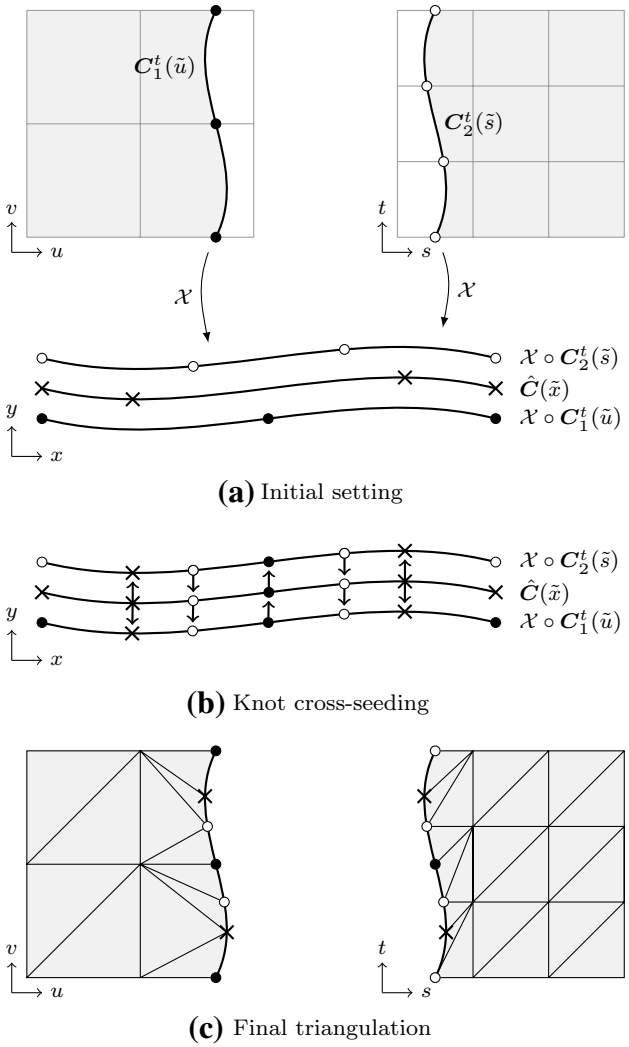


Fig. 46 Generation of conforming triangulations along the intersection of two patches: **a** definition of Bézier segments of the intersection curve $\hat{\boldsymbol{C}}(\tilde{x})$ and the trimming curves $\boldsymbol{C}^t_1(\tilde{u})$ and $\boldsymbol{C}^t_2(\tilde{s})$. **b** Closest point projection to find corresponding points on the other curves. **c** Addition of Bézier segments due to the exchanged points and specification of associated triangular regions. Segments are marked by *crosses*, *black points*, and *white points* based on their origin. The offset between $\hat{\boldsymbol{C}}(\tilde{x})$ and the images $\mathcal{X} \circ \boldsymbol{C}^t_1(\tilde{u})$ and $\mathcal{X} \circ \boldsymbol{C}^t_2(\tilde{s})$ shall emphasize that they do not coincide in model space. In **b**, *arrows* indicate the projections performed

Next, the relationship of adjacent patches along the intersection is established. This is done similar to the knot cross-seeding procedure described in the previous subsection. Hence, we adopt this term here as well. Figure 46 summarizes the basic procedure. Firstly, the intersection curve $\hat{\boldsymbol{C}}(\tilde{x})$ in model space and the trimming curves $\boldsymbol{C}^t_1(\tilde{u})$

and $\boldsymbol{C}_2^t(\tilde{s})$ are subdivided into Bézier segments at their knot values and intersections with the trimmed parameter space. Then, the endpoints of these segments are projected to the other curves and the corresponding parametric values are computed. In other words, the trimming curve are refined based on the knot information of the other trimming curve and the intersection curve in model space. Consequently, the resulting Bézier segments of a curve have corresponding counterparts in the other curves. However, the distinct segments do not coincide in model space. At this point, the purpose of the knot cross-seeding is to obtain an aligned triangulation along the intersection. Triangular patches are specified within each trimmed surface so that one of their boundaries represents a Bézier segment.

The construction of these triangular Bézier patches which are arbitrarily located within the trimmed surface is performed accordingly to Lasser [182]. In general, a Bézier triangle T of degree $\tilde{p}$ in a tensor product basis of a surface $\boldsymbol{R}(u, v)$ of degrees p and q yields a triangular patch $\boldsymbol{S}^{\triangle}(r, s, t)$ of degree $\tilde{p}(p+q)$. Xia and Qian [314] focus on the linear case, i.e., $\tilde{p} = 1$, meaning that the trimming curve is approximated by linear segments. Thus, the composition $\boldsymbol{S}^{\triangle}(r, s, t) = \boldsymbol{R}(T(r, s, t))$ is given by

$$\boldsymbol{S}^{\triangle}(r, s, t) = \sum_{|\mathbf{I}|=p+q} B_{\mathbf{I},p+q}(r, s, t)\boldsymbol{c}_{\mathbf{I}}^{\triangle}, \tag{81}$$

where $B_{\mathbf{I},p+q}$ refer to the Bernstein basis (37) of the triangular patch, $\mathbf{I}$ is an index triplet (i, j, k), and $|\mathbf{I}| = i + j + k$. It remains to determine the corresponding control points $\boldsymbol{c}_{\mathbf{I}}^{\triangle}$. Therefore, the construction points $\boldsymbol{R}_{0,0}^{p,q}(u_{\mathbf{I}^u}^p, v_{\mathbf{I}^v}^q)$ of the blossom of $\boldsymbol{R}(u, v)$ are needed. Regarding the u-direction, for instance, these points are recursively defined by

$$\begin{aligned}\boldsymbol{R}_{i,j}^{a,b}(u_{\mathbf{I}^u}^a, v_{\mathbf{I}^v}^b) = &\left(1 - u_{\mathbf{I}_a^u}\right)\boldsymbol{R}_{i,j}^{a-1,b}(u_{\mathbf{I}^u}^{a-1}, v_{\mathbf{I}^v}^b) \\ &+ u_{\mathbf{I}^u{}_a}\boldsymbol{R}_{i+1,j}^{a-1,b}(u_{\mathbf{I}^u}^{a-1}, v_{\mathbf{I}^v}^b),\end{aligned} \tag{82}$$

where the control points of the surface $\boldsymbol{R}(u, v)$ are used as initial values $\boldsymbol{R}_{i,j}^{0,0}$. The construction in the v-direction is performed in an analogous manner. The superscripts a and b denote distinct steps of the recurrence relation in the u -direction and v-direction, respectively. Note that Eq. (82) is an adaptation of the de Casteljau algorithm that allows employing new parameter values $u_{\mathbf{I}_a^u}$ in every iteration. Likewise, the index tuples are given by $\mathbf{I}^v = \mathbf{I}_1^v + \cdots + \mathbf{I}_b^v$ and $\mathbf{I}^u = \mathbf{I}_1^u + \cdots + \mathbf{I}_\alpha^u$ with α referring to the related superscript, i.e., a or $a-1$. Using this recursion, the control points of the triangular patch $\boldsymbol{S}^{\triangle}(r, s, t)$ are obtained by

$$\boldsymbol{c}_{\mathbf{I}}^{\triangle} = \sum_{\mathbf{I}^u+\mathbf{I}^v=\mathbf{I}} \frac{1}{\binom{p+q}{\mathbf{I}}}\boldsymbol{R}_{0,0}^{p,q}(u_{\mathbf{I}^u}^p, v_{\mathbf{I}^v}^q), \tag{83}$$

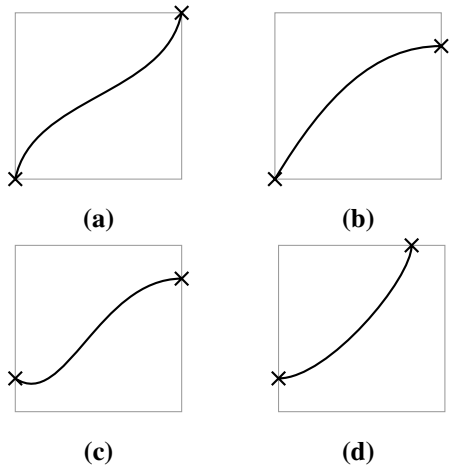


Fig. 47 Illustration of the most common valid cutting patterns of a single knot span. The actual element type is determined by the direction of the trimming curve. The *crosses* highlight the intersection points of the trimming curve with the element

with

$$\mathbf{I}^u = \mathbf{I}_1^u + \cdots + \mathbf{I}_q^u \quad \text{and} \quad \mathbf{I}^v = \mathbf{I}_1^v + \cdots + \mathbf{I}_p^v, \tag{84}$$

where each of these index triples consists of

$$\mathbf{I}_\beta^\alpha = \left(i_\beta^\alpha, j_\beta^\alpha, k_\beta^\alpha\right) \quad \text{with} \quad i_\beta^\alpha, j_\beta^\alpha, k_\beta^\alpha \in \{0, 1\}, \tag{85}$$

and further

$$|\mathbf{I}^u| = \left|\mathbf{I}_1^u\right| + \cdots + \left|\mathbf{I}_p^u\right| = p, \tag{86}$$

$$|\mathbf{I}^v| = \left|\mathbf{I}_1^v\right| + \cdots + \left|\mathbf{I}_q^v\right| = q, \tag{87}$$

$$|\mathbf{I}| = |\mathbf{I}^u| + |\mathbf{I}^v| = p + q. \tag{88}$$

This procedure is applied to cover the valid area of every trimmed Bézier surface by a set of triangular Bézier patches. Each Bézier segment of a trimming curve is represented by an edge of such a triangular patch. Finally, those edges are replaced by the corresponding Bézier segment of the intersection curve in model space. Since this substitution is carried out for all patches, a seamless join between adjacent surfaces is obtained. The approximation error introduced may be controlled by refining the patches along the trimming curves.

It is worth noting that Xia and Qian [314] use their reconstruction procedure as an intermediate step in order to set up a volumetric parameterization of B-Rep models. The watertight triangular Bézier surface representation provides the starting point for a construction of volumetric Bézier tetrahedra.

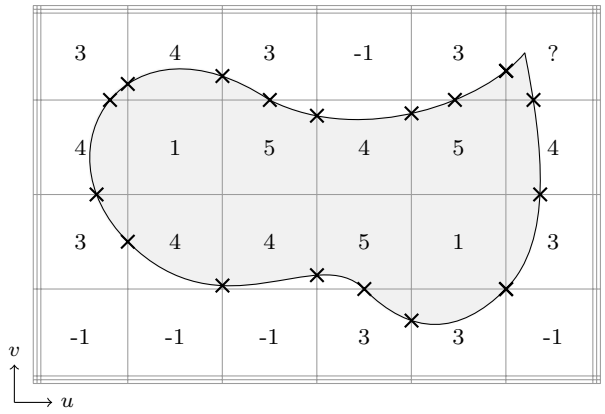


Fig. 48 Trimmed parameter space and corresponding element types: *1* labels untrimmed knot spans whereas -1 denotes knot spans which are outside of the computational domain. In case of trimmed knot spans the element type indicates the number of interior edges, i.e., *3*, *4*, or *5*. The *question mark* indicates a special case. The intersections of the trimming curve with the parameter lines are highlighted by *crosses*

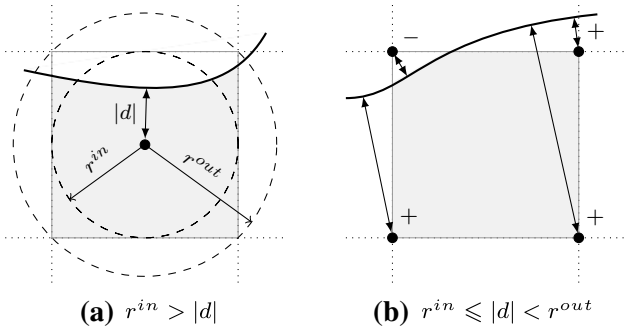


Fig. 49 Detection of cut elements according to Kim et al. [160, 161]: **a** first assessment based on the inscribed and circumscribed *circles* of the element and if necessary, **b** further comparison of the signed distance of the element corners

5.3 Local Approaches

Local techniques employ a completely different philosophy than their global counterparts, that is, the geometry model is not modified but the analysis has to deal with all deficiencies of trimmed solid models. Thereby, the trimmed parameter space is used as background parameterization for the simulation while the trimming curves determine the domain of interest. Hence, the analyzed area is embedded in a regular grid of knot spans which consists of *interior*, *exterior*, and *cut* elements. The following subsections discuss (i) the detection of theses distinct element sets, (ii) the integration of cut elements, (iii) the treatment of multipatch geometries, and (iv) the stability of a trimmed basis.

5.3.1 Element Detection

Before the actual analysis can be performed, the various element types and their position within the trimmed basis need to be identified. Interior elements are defined by non-zero knot spans that are completely within the valid domain and can be treated as in regular isogeometric analysis. Exterior ones, on the other hand, can be ignored since their entire support is outside of the domain of interest. Cut elements require special attention. One of the advantages of local approaches is that the cutting patterns of these elements are relatively simple compared to the complexity of the overall trimming curve. Figure 47 depicts topological cases of cut elements that are usually considered, e.g., [160, 161, 199, 202, 260]. It should be pointed out that other cases may exits as well, e.g., an element containing more than one trimming curve. These situations occur especially when the basis is very coarse. In general, the complexity of a trimming curve's topology within an element decreases as the fineness of the parameter space increases. Hence, (local) refinement is a common way to resolve invalid cutting patterns. This refinement may be performed for integration purpose only. Thus, no new knots are introduced, but the invalid element is subdivided in several valid integration regions. An alternative is to extend the valid cutting patterns as suggested by Wang et al. [307] or the construction of tailored integration rules for each cut element as proposed by Nagy and Benson [211]. However, the benefit of a restricted number of trimming cases facilitates the subsequent integration process.

Considering the situations shown in Fig. 47, cut elements have either 3, 4, or 5 edges, where one of them is a portion of the trimming curve. In this paper, we adapt the notation of Schmidt et al. [260] and label the type of cut elements by their number of edges. Interior and exterior elements are referred to as elements of type 1 and -1, respectively. Figure 48 illustrates a trimmed parameter space and the related element types. Note that the knot span in the upper right corner is an example of an invalid case since smooth element edges are usually assumed for the numerical integration. Possible strategies to deal with this element include subdivision into several integration regions, treatment as a type 4 element with two curved edges, or knot refinement through the kink of the curve. In general, kinks and straight trimming curves that are aligned with parameter lines are usual suspects for introducing special cases.

The portions of the trimming curve which are within each element have to be determined in order to get a proper description of cut knot spans. In particular, the intersections of the parameter grid with the trimming curve $\boldsymbol{C}^t(\tilde{u})$ are required, together with the corresponding parametric values $\tilde{u}^+$. The overall element detection task consists of the classification of knot span with respect to the trimming

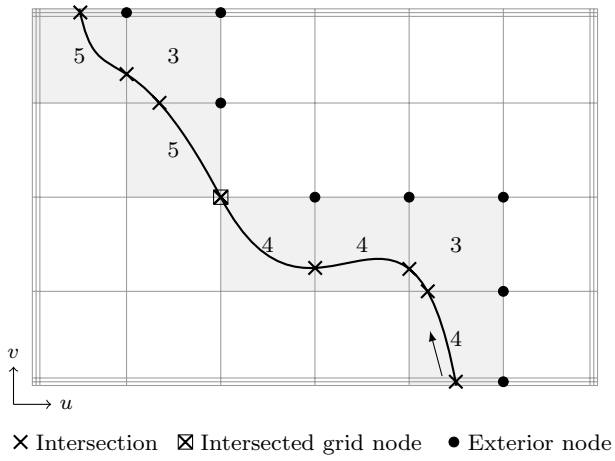


Fig. 50 Starting point for the separation of interior and exterior elements following the procedure of Schmidt et al. [260]. White knot spans are not classified yet. The *arrow* indicates the direction of the trimming curve

curves and the determination of trimming curve portions related to cut elements.

Kim et al. [160, 161] and Schmidt et al. [260] presented two different algorithmic solutions for the element detection problem. The procedure suggested by the former can be summarized by:

(i) Compute the minimal signed distance $d_{i,j}$ of the center of each non-zero knot span to all trimming curves to separate *interior* and *exterior* elements.
(ii) Identify *cut* elements by comparing $\left|d_{i,j}\right|$ with the radii $r^{in}_{i,j}$ and $r^{out}_{i,j}$ of the inscribed and circumscribed circles of the element. If $r^{in}_{i,j} \leqslant \left|d_{i,j}\right| < r^{out}_{i,j}$, the signed distance of the element corner nodes to the trimming curve are computed and compared as well.
(iii) Compute *intersection points* for each element cut by the trimming curve.

Both cases of the second step which specify cut elements are illustrated in Fig. 49. In Fig. 49a, the distance of the element's center to the trimming curve is smaller than the radius of the inscribed circle, whereas in Fig. 49b, the cut element is identified since the signed distances of its corner nodes are positive and negative.

On the other hand, Schmidt et al. [260] recommend to label all non-zero knot spans as interior elements as starting point for the following procedure:

(i) Determine all *intersection points* of the trimming curve and the grid produced by the tensor product of the knot vectors and sort them in a nondecreasing order with respect to the related values $\tilde{u}^+$.
(ii) Assign the element type of *cut* elements based on the position of successive intersection points.
(iii) Detect *exterior* elements based on their position relative to the cut elements.

It should be noted that successive intersection points mark start and end of trimming curve portions within an element. For the last task, the exterior nodes of cut elements can be used to initialize an incremental algorithm setting adjacent elements which are not labeled as cut elements to -1 [286]. Figure 50 illustrates the situation described. The nodes of these exterior elements are then used to determine further exterior elements. The procedure stops as soon as there are any adjacent elements of type 1 left.

On this basis, it may be concluded that the present approaches tackle the problem from two different directions. The former puts the element type in the focus with a subsequent calculation of the intersections of cut elements with the trimming curve, whereas the latter computes all intersections and derives the type of the elements afterwards. In general, the most important property of an element detection algorithm is its robustness since it hardly effects the overall efficiency of the simulation. Both approaches require a robust implementation of the curve-to-grid intersection computation. The Bézier clipping technique described in Sect. 3.5.2 could be used. The treatment of invalid cutting patterns applies also to both algorithms and depends on the subsequent integration procedure. The main difference between the approaches is that the former relies on a robust implementation of a point projection algorithm in order to determine the signed distance of a point to the trimming curve, while the latter requires a robust technique for detecting exterior elements based on the intersection information. There is perhaps no objective way to prefer one scheme to the other, but we would like to share our experiences with both schemes by mentioning some possible pitfalls in the following paragraphs.

The first point addresses the detection of exterior elements based on their relative position to cut elements. The starting point is illustrated in Fig. 50. Elements of type 1 are changed to type -1 if they are adjacent to the exterior nodes of cut and exterior elements. The search for adjacent elements can be performed in an incremental manner as it is done in "flood fill" algorithms, which are commonly used in graphics software [286]. It should, however, be emphasized that intersected grid nodes need special attention since the search for adjacent elements might propagate at these points to the valid domain. Furthermore, a proper treatment of zero knot spans is required.

The other note is concerned with the calculation of the signed distance to trimming curves. The shortest distance d_t of a test point $\boldsymbol{x}_t$ to a trimming curve $\boldsymbol{C}^t(\tilde{u})$ is defined as

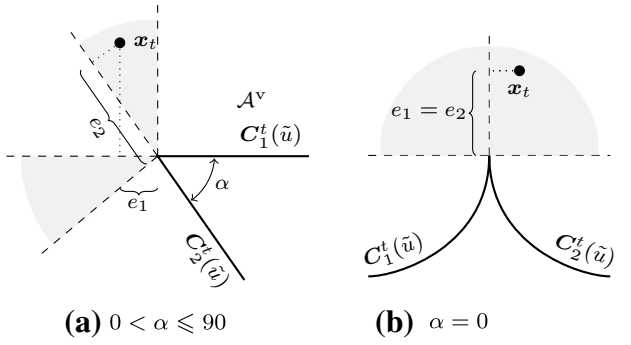


Fig. 51 Determination of the correct sign in case of multiple trimming curves $\boldsymbol{C}_i^t(\tilde{u})$ which describe an acute angle. The area which returns ambiguous signs is indicated in *gray*. (Courtesy of Jakob W. Steidl)

$$d_t = \min\left\{\left\|\boldsymbol{C}^t(\tilde{u}) - \boldsymbol{x}_t\right\|\right\} = \left\|\boldsymbol{C}^t(\tilde{u}^*) - \boldsymbol{x}_t\right\|. \tag{89}$$

A Newton–Raphson iteration scheme is employed to determine the parametric values $\tilde{u}^*$ [192, 230]. The corresponding sign s indicates on which side of $\boldsymbol{C}^t(\tilde{u})$ the point $\boldsymbol{x}_t$ is located. It can be computed by the cross product of the tangent vector $\boldsymbol{t} = (t_u, t_v)^\intercal$ at the projected point $\boldsymbol{x}_p = \boldsymbol{C}^t(\tilde{u}^*)$ and the direction vector $\boldsymbol{d} = (d_u, d_v)^\intercal$ from $\boldsymbol{x}_p$ to $\boldsymbol{x}_t$. In two dimensions, the analog to the cross product is given by the determinant, hence the sign is calculated by

$$s = t_u d_v - t_v d_u. \tag{90}$$

In case of *non-smooth* trimming curves, more than one minimum might exist as pointed out in [287]. From a practical point of view this is only relevant if these minima have different signs. Such cases appear in the vicinity of sharp corners as shown in Fig. 51. The correct sign can be determined by the projected distance calculated by the dot product

$$e_i = \boldsymbol{v} \cdot \boldsymbol{t}_i, \quad i = 1, 2, \tag{91}$$

$$s = \operatorname{sign}\left\{\min\left\{e_1, e_2\right\}\right\}. \tag{92}$$

If the angle $\alpha = 0$, i.e., $e_1 = e_2$, the curvatures of the curves may be compared

$$s = \begin{cases} 1 & \textit{if} \kappa_1 > \kappa_2, \\ -1 & \textit{otherwise}, \end{cases} \tag{93}$$

where κ_1 denotes the curvature of the curve that ends at the corner.

5.3.2 Integration

Various strategies to integrate cut elements $\tilde{\tau} \in \mathcal{A}^v$ are outlined in this subsection. In general, numerical integration is performed using conventional Gauss–Legendre quadrature. The integral over each $\tilde{\tau}$

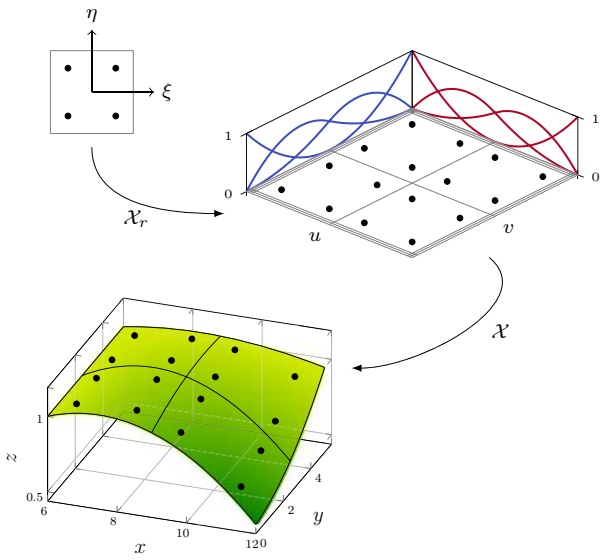


Fig. 52 Distribution of quadrature points indicated by *black* points over a regular patch

$$I_{\tilde{\tau}} = \int_{\Omega_{\tilde{\tau}}} f(\boldsymbol{x}) \, \mathrm{d}\Omega_{\tilde{\tau}} \tag{94}$$

is substituted by a weighted sum of point evaluations

$$I_{\tilde{\tau}} \approx \sum_{g=1}^{n} f\left(\boldsymbol{y}_g\right) G(u_g, v_g) J_{\hat{\tau}}(\xi_g, \eta_g) w_g. \tag{95}$$

The related quadrature points $\boldsymbol{y}$ and the corresponding weights w are specified in the reference element $\hat{\tau} = [-1, 1]^2$. The coordinates for the pointwise evaluation of the integrand f are determined by the integral transformation $\mathcal{X}_r(\xi, \eta)\colon \mathbb{R}^2 \mapsto \mathbb{R}^2$ from $\hat{\tau}$ to $\tilde{\tau}$ and the geometrical mapping $\mathcal{X}(u, v)\colon \mathbb{R}^2 \mapsto \mathbb{R}^3$, i.e., $\boldsymbol{y}_g = \mathcal{X}(u_g, v_g) = \mathcal{X}(\mathcal{X}_r(\xi_g, \eta_g))$ as illustrated in Fig. 52. In order to take these mappings into account, the quadrature weights w_g are multiplied by the Gram's determinant

$$G(u, v) := \sqrt{\det\left(\mathbf{J}_{\mathcal{X}}^\intercal(u, v)\mathbf{J}_{\mathcal{X}}(u, v)\right)}, \tag{96}$$

given by the Jacobian matrix $\mathbf{J}_{\mathcal{X}}$ of the geometrical mapping. The Jacobian determinant of $\mathcal{X}_r$

$$J_{\hat{\tau}}(\xi_g, \eta_g) = \det\left(\mathbf{J}(\xi_g, \eta_g)\right), \tag{97}$$

is evaluated with respect to the reference coordinates ξ_g and η_g of the integration point $\boldsymbol{y}_g$. The definition of the integral transformation $\mathcal{X}_r$ and the related $J_{\hat{\tau}}$ is straightforward in case of regular elements. However, the domain of cut

elements is more complex and thus, the definition of $\mathcal{X}_r$ is more involved.

Numerical integration of cut elements is required in various simulation schemes. Besides the analysis of trimmed geometries, it is also needed in the context of fictitious domain methods and the extended finite element method. There are numerous approaches and a vast body of literature proposing strategies to specify a proper integration of $\tilde{\tau}$. It may be performed so that the trimming curve is taken into account in an exact or approximate manner. In this paper, the main focus is on techniques presented in the context of trimmed NURBS objects. They can be broadly classified into the following categories: (i) local reconstruction, (ii) approximate treatment, and (iii) exact treatment. The former is performed in model space, while the others operate in the parameter space in general.

5.3.2.1 Local Reconstruction Schmidt et al. [260] suggested to perform the adjustment of the integration region by a local reconstruction of the trimmed patch. Therefore, each cut element in the model space τ is remodeled as a single reconstruction patch $\hat{\tau}$. In particular, $\hat{\tau}$ is specified as a Bézier patch with degrees $\hat{p} \geqslant p$ and $\hat{q} \geqslant q$, where p and q refer to the degrees of the origin surface. The key idea is to represent $\hat{\tau}$ in terms of the original control points of τ. A transformation matrix $\mathbf{T}$ provides the relationship between the control points of the original and reconstructed patch. It can be computed by means of a least squares approximation where the system of equations is given by

$$\hat{\mathbf{B}}\hat{\mathbf{P}} = \mathbf{S} = \mathbf{B}\mathbf{P}. \tag{98}$$

This equation consists of

$$\hat{\mathbf{P}} = \begin{bmatrix} \hat{\boldsymbol{c}}_0 \\ \vdots \\ \hat{\boldsymbol{c}}_{\hat{n}-1} \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} \boldsymbol{c}_0 \\ \vdots \\ \boldsymbol{c}_{n-1} \end{bmatrix}, \quad \text{and} \quad \mathbf{S} = \begin{bmatrix} \boldsymbol{x}_0^s \\ \vdots \\ \boldsymbol{x}_l^s \end{bmatrix}, \tag{99}$$

representing the (unknown) control points of $\hat{\tau}$, the (known) control points of τ, and a set of sampling points $\boldsymbol{x}^s$ interpolated by both patches. The total number of control points involved is determined by the number of non-zero basis functions, i.e., $\hat{n} = (\hat{p}+1)(\hat{q}+1)$ and $n = (p+1)(q+1)$. The number of sampling points, i.e., $l+1$, is arbitrary but larger than $\hat{n}$. The basis function values of $\hat{\tau}$ at $\boldsymbol{x}^s$ are provided by

$$\hat{\mathbf{B}} = \begin{bmatrix} \hat{B}_{0,\hat{p}}(\hat{u}_0^s)\hat{B}_{0,\hat{q}}(\hat{v}_0^s) & \cdots & \hat{B}_{\hat{p},\hat{p}}(\hat{u}_0^s)\hat{B}_{\hat{q},\hat{q}}(\hat{v}_0^s) \\ \vdots & \ddots & \vdots \\ \hat{B}_{0,\hat{p}}(\hat{u}_l^s)\hat{B}_{0,\hat{q}}(\hat{v}_l^s) & \cdots & \hat{B}_{\hat{p},\hat{p}}(\hat{u}_l^s)\hat{B}_{\hat{q},\hat{q}}(\hat{v}_l^s) \end{bmatrix}, \tag{100}$$

and the corresponding values of τ are given by

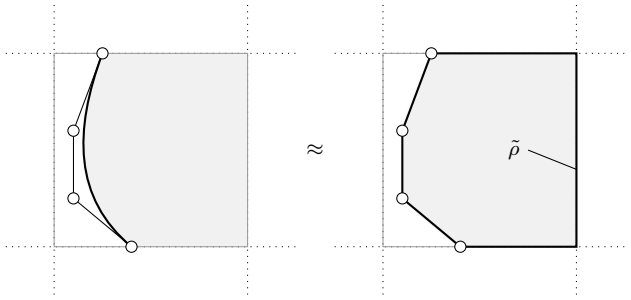


Fig. 53 Approximation of the cut element by a polytope $\tilde{\rho}$. The control points of the trimming curve are marked by *circles*

$$\mathbf{B} = \begin{bmatrix} B_{0,p}(u_0^s)B_{0,q}(v_0^s) & \cdots & B_{p,p}(u_0^s)B_{q,q}(v_0^s) \\ \vdots & \ddots & \vdots \\ B_{0,p}(u_l^s)B_{0,q}(v_l^s) & \cdots & B_{p,p}(u_l^s)B_{q,q}(v_l^s) \end{bmatrix}, \tag{101}$$

Note that a correlation between the parametric values has to be established so that $\boldsymbol{x}_i^s = \tau(u_i^s, v_i^s) = \hat{\tau}(\hat{u}_i^s, \hat{v}_i^s)$, $i = 0, \ldots, l$. The system of equations (98) is overdetermined consisting of $l+1$ equations and $\hat{n}$ unknowns. In general, it cannot be solved exactly, but a good approximation of the solution can be found by forming

$$\hat{\mathbf{B}}^\intercal\hat{\mathbf{B}}\hat{\mathbf{P}} = \hat{\mathbf{B}}^\intercal\mathbf{S} = \hat{\mathbf{B}}^\intercal\mathbf{B}\mathbf{P}, \tag{102}$$

which yields the definition of the transformation matrix

$$\mathbf{T} = \left(\hat{\mathbf{B}}^\intercal\hat{\mathbf{B}}\right)^{-1}\hat{\mathbf{B}}^\intercal\mathbf{B}, \tag{103}$$

and the relation between the control points

$$\hat{\mathbf{P}} = \mathbf{T}\mathbf{P}. \tag{104}$$

As a result, numerical integration can be performed based on the regular reconstruction patch $\hat{\tau}$ and the simple mapping of the regular integration can be applied. The values obtained are distributed to the control points of the original patch using the transformation matrix $\mathbf{T}$. For more details, the interested reader is referred to [260].

This procedure can be directly applied to cut elements of types 3 and 4 as specified in Sect. 5.3.1. Type 5 elements may be subdivided into two four-sided regions. A drawback of the local reconstruction scheme is that it introduces an additional approximation error since the system of equations (102) cannot be solved exactly. Moreover, the stability of the computation of the transformation matrix might be affected if only a very small region of a cut element needs to be reconstructed.

5.3.2.2 Approximated Trimming Curve The following two schemes approximate the trimming curve $\boldsymbol{C}^t$ in order to define proper integration points within the parameter space.

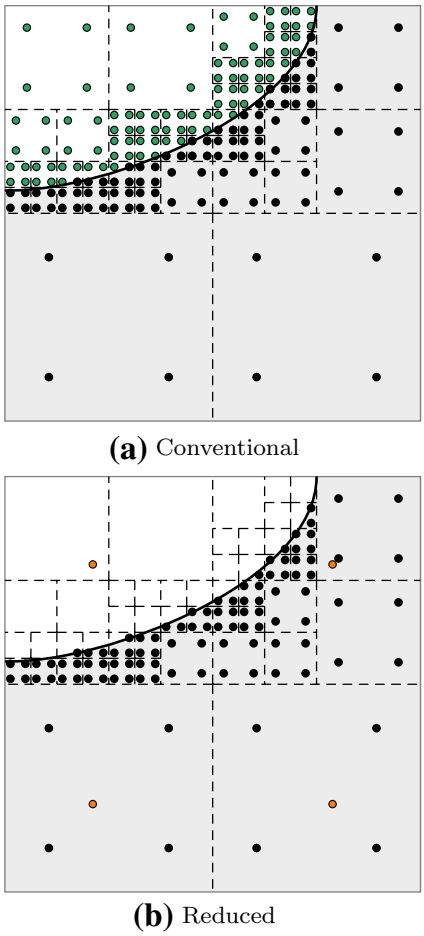


Fig. 54 Sub-cell structure of a single cut element: **a** conventional approach with quadrature points distributed within the valid (*black points*) and exterior (*green points*) domain and **b** reduced approach integrating over the whole element (*orange points*) and the valid domain (*black points*). The sub-cells are indicated by *dashed lines*. (Color figure online)

One uses a linear approximation of $\boldsymbol{C}^t$ to set up a tailored integration rule, whereas the other applies an adaptive subdivision to approximate the domain of the cut element.

A *tailored integration* rule can be established for each cut element $\tilde{\tau}$ as proposed in [211, 305, 306]. The control polygon[20] of $\boldsymbol{C}^t$ is used to represent $\tilde{\tau}$ by a polytope $\tilde{\rho}$ as shown in Fig. 53. The integral over $\tilde{\rho}$ can be reduced to a sum of line integrals over the edges of $\tilde{\rho}$ using Lasserre's theorems [183]. Therefore, the integration domain $\Omega_{\tilde{\rho}}$ has to be convex. Thus, a preprocessing step is applied to represent non-convex regions by a combination of convex ones. The line integrals provide reference solutions for the right-hand side of a set of moment-fitting equations given by

$$\sum_{i=1}^{m} f_j(u_i, v_i) w_i = \int_{\Omega_{\tilde{\rho}}} f_j(u, v) \, \mathrm{d}\Omega_{\tilde{\rho}}, \quad j = 1, \ldots, n. \tag{105}$$

They are used to computed a tailored quadrature rule, i.e., points $\boldsymbol{y}_i = (u_i, v_i)^\intercal$ and weights w_i, for all functions f_j of the desired functions space, e.g., monomials up to a certain degree. The goal is to find the lowest number of $\boldsymbol{y}_i$ so that the Eq. (105) are satisfied up to a certain tolerance, for each $\tilde{\tau}$ or rather $\tilde{\rho}$. The algorithm proposed in [211] starts with an initial set of $\boldsymbol{y}_i$ and successively eliminates one superfluous point after another. In each step, the reduced set of points is used to solve the system of equations (105) in the least squares sense.

The construction of a tailored quadrature has two main benefits: (i) the number of integration points per $\tilde{\tau}$ is optimized and (ii) *all* cutting patterns are covered by a single technique, including cases which had been labeled as invalid in Sect. 5.3.1. Of course, this comes at the price of a more involved preprocessing phase since every cut element has to be treated individually. Furthermore, an error is introduced due to the approximation of $\tilde{\tau}$ by $\tilde{\rho}$. This error can be reduced by refinement of the trimming curve since the control polygon converges to it. Still, the smooth higher degree representation is removed by a linear one. Finally, it should be pointed out that the reduction of integration points does not take the smoothness of the basis into account, as it is done in case of optimized quadrature rules for regular splines, see e.g., [118].

A completely different strategy for the integration of cut elements in based on *adaptive subdivision*. Researchers who developed the *finite cell method* applied this technique to trimmed geometries [107, 239, 253, 254]. The basic idea is to use a composed Gauss quadrature that aggregates integration points along the trimming curve. A cut element $\tilde{\tau}$ is decomposed into axis-aligned sub-cells $\tilde{\tau}^{\boxplus}$ based on a tree-structure, i.e., a quadtree in two dimensions. Starting from the initial cut element, each sub-cell is further subdivided into equally spaced sub-cells if it contains the trimming curve as displayed in Fig. 54a. This recursive procedure is performed up to a user-defined maximal depth. Following the spirit of fictitious domain methods the integral I^c over the complete element is defined as

$$I^c = \sum_{i=1}^{I} I^{\mathrm{v}}_{\tilde{\tau}_i^{\boxplus}}(\alpha^{\mathrm{v}}) + \sum_{j=1}^{J} I^{-}_{\tilde{\tau}_j^{\boxplus}}(\alpha^{-}). \tag{106}$$

The factors $I^{\mathrm{v}}_{\tilde{\tau}_i^{\boxplus}}$ and $I^{-}_{\tilde{\tau}_j^{\boxplus}}$ are the integrals over the valid domain $\mathcal{A}^{\mathrm{v}}$ and the complementary exterior domain $\mathcal{A}^{-}$, respectively. Integration points in the interior of $\mathcal{A}^{\mathrm{v}}$ are

[20] To be precise, it is suggested to use the control points of the intersection curves in the model space and to apply point inversion to determine their location in the parameter space. However, there is no particular reason why the control polygon of the trimming curve cannot be used directly.

multiplied by $\alpha^{\mathrm{v}} = 1$, whereas exterior integration points are multiplied by a value that is almost zero, e.g., $\alpha^{-} = 10^{-14}$ as suggested in [253]. The integration procedure can be improved with respect to the number of quadrature points by

$$I^{c} = I^{-}_{\tilde{\tau}}(\alpha^{-}) + \sum_{i=1}^{I} I^{\mathrm{v}}_{\tilde{\tau}_i^{\boxplus}}(\alpha^{\mathrm{v}} - \alpha^{-}), \tag{107}$$

where $I^{-}_{\tilde{\tau}}(\alpha^{-})$ represents the integral over the whole cut element without taken the trimming curve into account. The integration over the valid domain is performed as before by the composite quadrature, yet with another weighting factor, i.e., $(\alpha^{\mathrm{v}} - \alpha^{-})$. Such an improved sub-cell integration is illustrated in Fig. 54b.

The key features of this approach are its simplicity and generality. The definition of integral transformation $\mathcal{X}_r$ and its Jacobian is straightforward, due to the axis-aligned shape of the sub-cells. Again, *all* cutting patterns (including invalid ones) can be addressed with a single algorithm. Moreover, the algorithm can be easily extended to higher dimensions. The downside is that the trimming curve is only approximated. Consequently, the integration region is not represented exactly and an additional approximation error is introduced. In fact, the accuracy of the integral ceases at a certain threshold [173, 175]. This threshold may be improved by the subdivision depth, but a fine resolution of sub-cells results in a vast number of quadrature points. Further, refined sub-cells do not converge to the trimming curve in contrast to the previous approach. One of the great successes of the finite cell method was the demonstrated ability to achieve higher rates of convergence for higher-order elements and splines, and even exponential rates in the context of the *p*-method.

5.3.2.3 Exact Trimming Curve The following techniques focus on defining a proper mapping $\mathcal{X}_r$ from the reference element $\hat{\tau}$ to the cut element $\tilde{\tau} \in \mathcal{A}^{\mathrm{v}}$ so that the trimming curve is exactly represented. Depending on the cutting pattern, $\tilde{\tau}$ may be represented by a disjointed set of integration regions $\tilde{\tau}^{\boxdot}$ such that

$$\tilde{\tau} = \bigcup_{i=1}^{I} \tilde{\tau}_i^{\boxdot}. \tag{108}$$

In contrast to the sub-cells of the previous scheme, the regions $\tilde{\tau}^{\boxdot}$ are not aligned with the axes of the parameter space and at least one $\tilde{\tau}^{\boxdot}$ has an edge which is described by the portion of the trimming curve $\boldsymbol{C}^t$ within $\tilde{\tau}$.

There are various ways to specify $\mathcal{X}_r$. Ruled surface (26) and Coons patch (35) interpolation may be applied, where the portion of the trimming curve within $\tilde{\tau}$ is considered for the construction [199, 307]. An example of local ruled surface mappings for various element types are shown in Fig. 55a. These methods may be interpreted as local counterparts of the global reconstruction schemes presented in Sects. 5.2.1 and 5.2.2. It is worth noting that approaches based on the *blending function method* [95, 173, 174] can be included into this category, because this method also employs a transfinite mapping [103]. In the *nested Jacobian approach*, integral transformation is also defined by a local NURBS surface combined with a nested subdivision [38, 227]. Thus, $\mathcal{X}_r$ consists of the local surface mapping and an additional transformation to the subregion. A corresponding distribution of quadrature points is shown in Fig. 55b. In contrast to both previous references, i.e., [199, 307], type 5 elements are not decomposed into three triangular ones, but a bisection of the knot span is performed. Recently, an *adaptive Gaussian integration procedure* has been proposed [37]. This variation of the nested Jacobian approach defines the local surface parameterization within the reference space instead of the trimmed parameter space as illustrated in Fig. 55c. Therefore, the trimming curve is transformed to the reference space by scaling and rotation. The integration points and their weights are adapted by scaling the η-direction such that the points are located within the region described by the transformed trimming curve. The motivation for the adaptive Gaussian integration procedure is to treat the various cutting patterns by a single approach.

Another very common strategy is to adopt the integration scheme developed in the context of the *NURBS-enhanced finite element method* [147, 148, 160, 161, 272, 273]. Using this scheme, every cut element is subdivided into a set of triangles. Those triangles that only consist of straight edges are subjected to conventional integration rules for linear triangles. The other triangles are treated by a series of mappings that take the curved edge into account

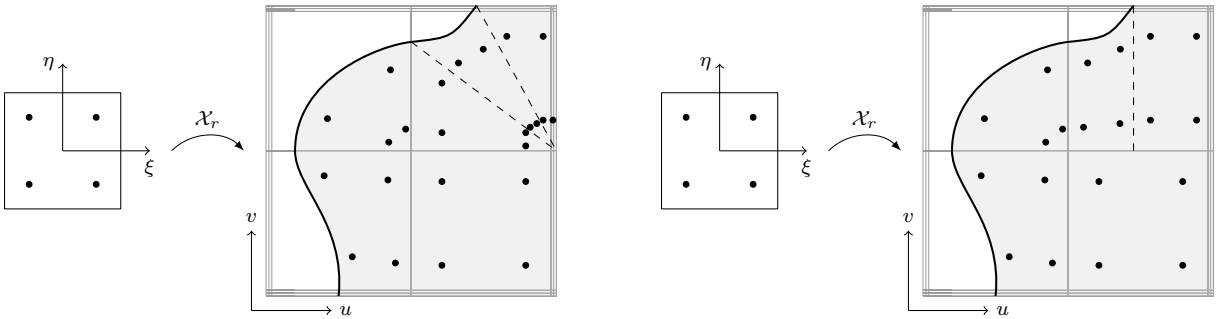
$$\mathcal{X}_r := \mathcal{X}_{s,t}(\mathcal{X}_{\tilde{u},\zeta}(\mathcal{X}_{\xi,\eta}(\xi,\eta))). \tag{109}$$

Figure 55d displays the components of this series. Suppose the corner nodes of the triangle in the trimmed parameter space are labeled $\boldsymbol{x}_1^{\triangle}$ to $\boldsymbol{x}_3^{\triangle}$, where the beginning and the end of the trimming curve portion within the considered triangle are denoted by $\boldsymbol{x}_2^{\triangle}$ and $\boldsymbol{x}_3^{\triangle}$, respectively. The transformation $\mathcal{X}_{s,t}: \boldsymbol{x}(s,\, t) \mapsto \boldsymbol{x}(u,\, v)$ describes the mapping of a linear three node element

$$\mathcal{X}_{s,t} := \boldsymbol{x}(u,\, v) = t\boldsymbol{x}_1^{\triangle} + (1 - s - t)\boldsymbol{x}_2^{\triangle} + s\boldsymbol{x}_3^{\triangle}. \tag{110}$$

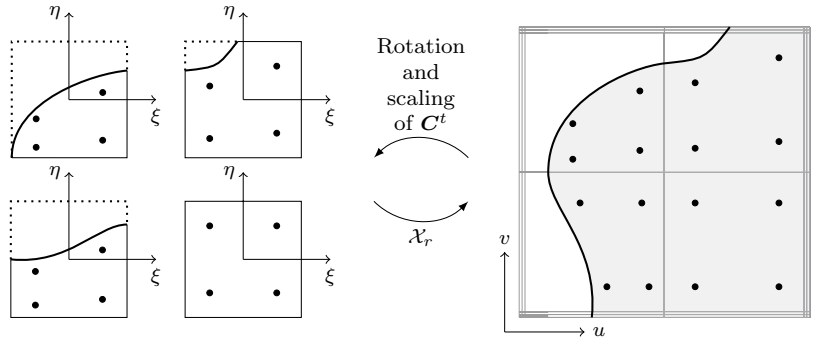
In order to address the curved edge, the trimming curve is transformed into the s,t-coordinate system by

$$\begin{aligned} \phi(\tilde{u}) &= \mathcal{X}_{s,t}^{-1} \cdot \boldsymbol{C}^t(\tilde{u}) \\ &= \begin{bmatrix} x_3^{\triangle} - x_2^{\triangle} & x_1^{\triangle} - x_2^{\triangle} \end{bmatrix}^{-1} \left(C^t(\tilde{u}) - x_2^{\triangle} \right). \end{aligned} \tag{111}$$

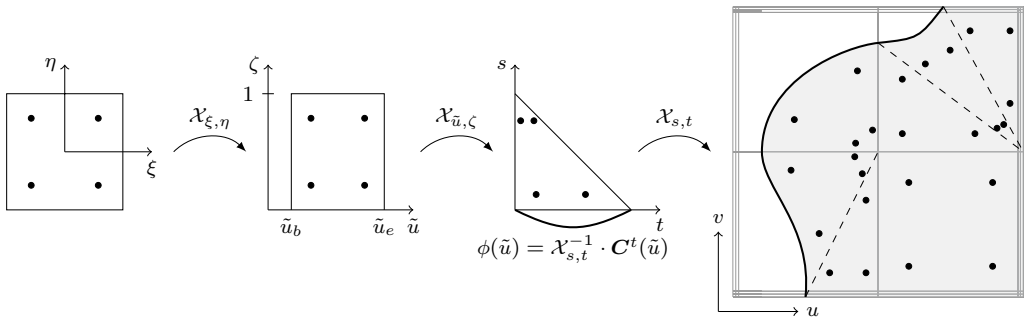


(a) Local ruled surface mapping

(b) Nested Jacobian approach



(c) Adaptive Gaussian integration procedure



(d) NURBS enhanced FEM integration

Fig. 55 Distribution of quadrature points due to various approaches which represent the trimming curve exactly. *Dashed lines* indicate a subdivision of a cut element into integration regions

The next mapping $\mathcal{X}_{\tilde{u},\zeta}: \boldsymbol{x}(\tilde{u}, \zeta) \mapsto \boldsymbol{x}(s, t)$ converts the triangular domain into a rectangular one which possesses straight edges only. It is given by

$$\mathcal{X}_{\tilde{u},\zeta} := \begin{cases} s &= \phi_s(\tilde{u})\ (1-\zeta), \\ t &= \phi_t(\tilde{u})\ (1-\zeta) + \zeta. \end{cases} \tag{112}$$

Finally, the transformation $\mathcal{X}_{\xi,\eta}: \boldsymbol{x}(\xi, \eta) \mapsto \boldsymbol{x}(\tilde{u}, \zeta)$ of the reference space $[-1, 1]^2$ to the rectangular region is performed by

$$\mathcal{X}_{\xi,\eta} := \begin{cases} \tilde{u} &= \frac{\xi}{2}(\tilde{u}_e - \tilde{u}_b) + \frac{1}{2}(\tilde{u}_e + \tilde{u}_b) \\ \zeta &= \frac{\eta}{2} + \frac{1}{2}, \end{cases} \tag{113}$$

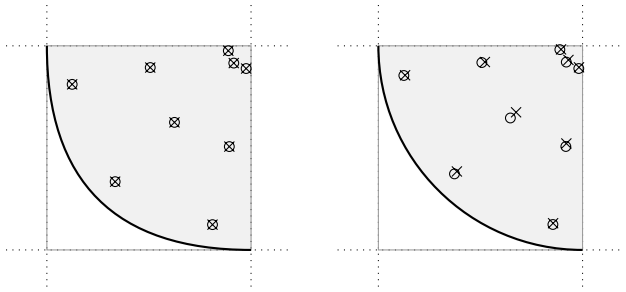
where $\tilde{u}_b$ and $\tilde{u}_e$ are the parametric values of the beginning and the end of the trimming curve portion within the triangle, i.e., $\boldsymbol{C}^t(\tilde{u}_b) = \boldsymbol{x}_2^\Delta$ and $\boldsymbol{C}^t(\tilde{u}_e) = \boldsymbol{x}_3^\Delta$. The Jacobian determinant of the overall mapping $\mathcal{X}_r$ is determined by

$$J_{\hat{\tau}} = \det\left(\mathbf{J}_{s,t}\right) \det\left(\mathbf{J}_{\tilde{u},\zeta}\right) \det\left(\mathbf{J}_{\xi,\eta}\right), \tag{114}$$

with

$$\mathbf{J}_{s,t} = \begin{bmatrix} u_3^\Delta - u_2^\Delta & u_3^\Delta - v_2^\Delta \\ u_1^\Delta - u_2^\Delta & v_1^\Delta - v_2^\Delta \end{bmatrix}, \tag{115}$$

$$\mathbf{J}_{\tilde{u},\zeta} = \begin{bmatrix} \frac{\partial \phi_s(\tilde{u})}{\partial \tilde{u}}(1-\zeta) & \frac{\partial \phi_t(\tilde{u})}{\partial \tilde{u}}(1-\zeta) \\ -\phi_s(\tilde{u}) & 1 - \phi_t(\tilde{u}) \end{bmatrix}, \tag{116}$$



(a) B-spline curve **(b)** NURBS curve

Fig. 56 Comparison of the distribution of Gauss points within a cut element of type 3 based on the NURBS enhanced FEM mapping (*circles*) and ruled surface parameterization (*crosses*). The trimming curve is described by either **a** a B-spline curve or **b** a NURBS curve

$$\mathbf{J}_{\xi,\eta} = \begin{bmatrix} \frac{1}{2}(\tilde{u}_e - \tilde{u}_b) & 0 \\ 0 & \frac{1}{2} \end{bmatrix}. \tag{117}$$

The coefficients $u_i^{\triangle}$ and $v_i^{\triangle}$ refer to the coordinates of the corner nodes $\boldsymbol{x}_i^{\triangle}$ and the derivative of the transformed trimming curve is calculated by

$$\frac{\partial \phi(\tilde{u})}{\partial \tilde{u}} = \begin{bmatrix} \boldsymbol{x}_3^{\triangle} - \boldsymbol{x}_2^{\triangle} & \boldsymbol{x}_1^{\triangle} - \boldsymbol{x}_2^{\triangle} \end{bmatrix}^{-1} \left(\frac{\partial \boldsymbol{C}^t(\tilde{u})}{\partial \tilde{u}} \right). \tag{118}$$

The various integration schemes are summarized in Fig.55. Their common and most essential feature is that the integration region is exactly represented. The main difference between the strategies is the partitioning of a cut element $\tilde{\tau}$ into integration regions $\tilde{\tau}^{\square}$. In fact, the series of mappings (109) shown in Fig. 55d yields the same distribution of quadrature points over a triangular element as a ruled surface interpolation (26) illustrated in Fig. 55a, if the trimming curve is a B-spline curve. In case of NURBS curves, on the other hand, different distributions are obtained. These two cases are compared in Fig. 56.

In general, it seems that good results can be obtained with either of these concepts, especially for moderate degrees. However, it has been demonstrated that the properties of coordinate mappings and the corresponding placement of interior nodes is crucial for the convergence behavior of conventional higher degree ($p > 3$) finite elements [216]. With this in mind, additional research might be useful to assess the quality of the mapping schemes presented with respect to their performance for higher degree.

5.3.3 Multipatch Geometries

A robust treatment of multiple patches is the most challenging part of analyzing trimmed geometries. While single patch analysis can exploit the benefits of trimmed representations, all the deficiencies elaborated in Sect. 3 surface as soon as solid models described by several trimmed patches are considered. In case of finite element methods, the main ingredients to cope with this situation are: (i) a weak coupling formulation and (ii) a robust procedure linking the degrees of freedom of adjacent patches to each other. This subsection closes with some general statements regarding the continuity of trimmed multipatch geometries.

5.3.3.1 Weak Coupling Weak enforcement of constraints is a common problem in computational mechanics, see e.g., [138, 166, 312] and the references cited therein. Such techniques are required in several contexts like mesh-independent imposing of essential boundary conditions and domain decomposition methods. The latter covers a versatile field of applications including contact problems, parallelization, and coupling of subdomains described by different physics or non-conforming discretizations. Numerous approaches have been developed and each one possesses different benefits and disadvantages. The most popular schemes are based on Lagrange multipliers [12], the penalty method [13, 139], or Nitsche's method [112, 215]. These methods are separated by a fine line: the penalty method may be viewed as an approximation of the Lagrange multiplier method [139]. Furthermore, the Nitsche method may be referred to as a consistent penalty method [252]. In addition, the close relationship of the Nitsche method to the stabilized Lagrange multiplier method [17, 18] has been outlined in [288].

The use of Lagrange multipliers is a very general way to enforce constraints to a system of equations which is applicable to all kinds of problems. Following Huerta et al. [138] the main disadvantages are: (i) the system of equations increases due to the Lagrange multipliers which are incorporated as additional degrees of freedom, (ii) the resulting system is not positive definite, and (iii) the introduction of a separate field for the Lagrange multipliers yields a saddle-point problem which must satisfy a stability condition known as the inf–sup or Babuška–Brezzi condition. In order to fulfill the last point, the interpolation fields of the unknowns and the Lagrange multipliers must be coordinated, which is not a trivial task, examples of choices for the interpolation functions can be found in [139].

The penalty method is easy to implement and avoids the problems mentioned above. However, uniform convergence to the solution can only be guaranteed if the applied penalty parameter increases as the mesh is refined [7]. This is crucial since the system matrix becomes ill-conditioned when the penalty parameter gets large. Usually, a fixed parameter value is chosen and as a result, the quality of the approximation cannot be improved below a certain error.

Nitsche's method introduces a penalty term too, but it is considerably smaller than in the penalty method [80,

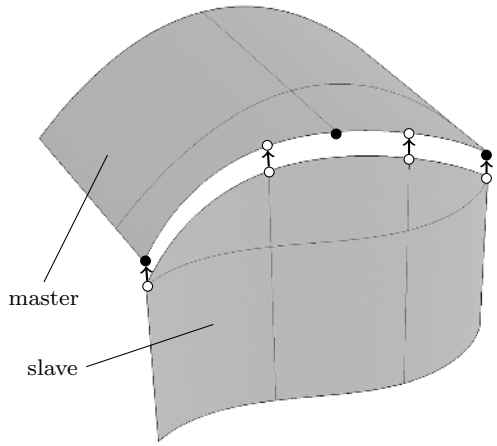


Fig. 57 Closest point projections of a slave patch to a master patch. The *lines* on the surfaces represent the grid of the underlying parameter space. The intersections of these *lines* with the trimming curves are illustrated by *white* and *black dots* for the slave and master patch, respectively. The projections themselves are indicated by *arrows*

254]. According to Huerta et al. [138] the only problem of Nitsche's method is that it is not as general as the other procedure. Thus, it is not straightforward to provide an implementation for some problem types.

These techniques have been successfully applied to various isogeometric analysis applications, e.g. [23, 24, 39, 80, 81, 163, 213]. A comparison of the three schemes can be found in [7]. Also in the context of coupling trimmed patches, the Lagrange multiplier method [307], the penalty method [37, 38], and the Nitsche method [107, 164, 254] have been successfully applied already. In none of these publications, the surface type motivated the choice of the weak coupling strategy. In other words, trimmed patches do not introduce additional arguments to prefer one approach to the other. Nevertheless, it is important to emphasize again that trimming curves of adjacent patches do *not* describe the same curve in model space. Thus, adjacent patches have non-conforming parameterizations as well as gaps and overlaps along their intersection.

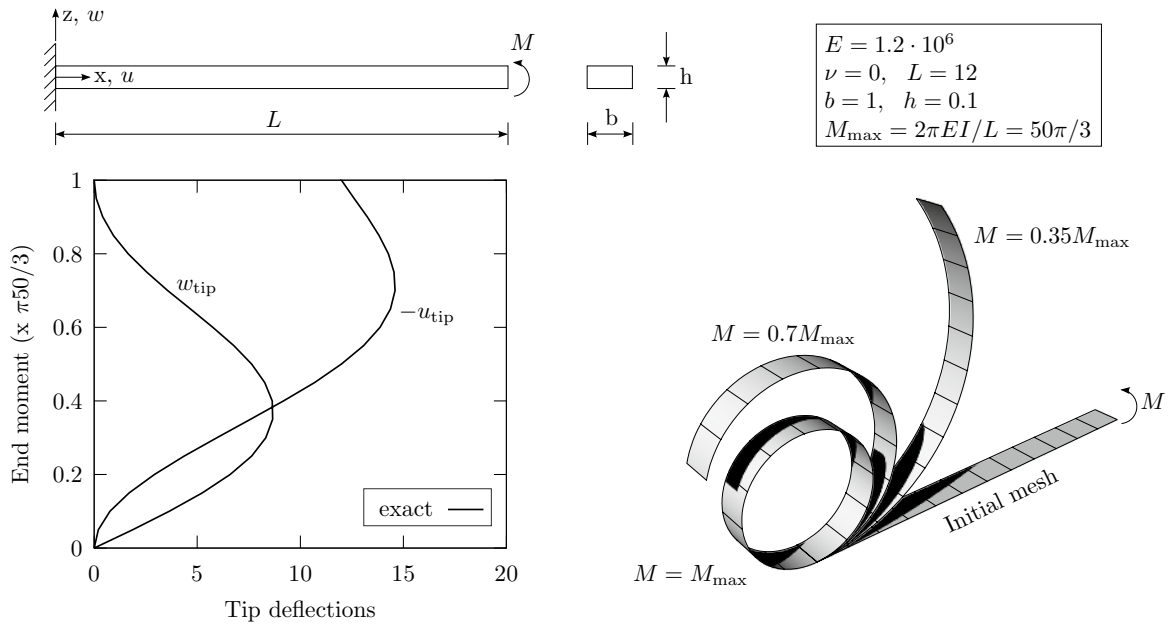
5.3.3.2 Linking of Degrees of Freedom Breitenberger et al. [38] presented a procedure that is able to deal with complex design models and it has been discussed in more detail in the related thesis [37]. In addition to a weak coupling formulation, trimming curves of adjacent patches are connected by so-called edge elements that contain the required topological information. To be precise, the trimming curves are treated by a *master–slave* concept where points of the slave curve are mapped to the master curve. These points are the intersections of the slave trimming curve with the grid lines of its own parameter space. The mapping to the master curve is performed in model space by means of a *point inversion algorithm* [192, 230]. The algorithm is usually carried out by a Newton–Raphson scheme and provides the closest projection of a point to a curve as shown in Fig. 57. In addition, the related parametric values of the master curve are provided by the point inversion scheme. The accumulation of these values and the original grid intersections of the master curve define a set of integration regions. Within each region, quadrature points are specified and the corresponding points of the slave curve can again be computed by the point inversion algorithm. To sum up, the relation of two related trimming curves is establish by an iterative procedure in model space which computes the shortest distance of a point defined by one curve to the other curve. This is indeed the same concept as for the knot cross-seeding procedures presented in Sect. 5.2 in the context of global approaches. In theory, this is a straightforward task, but its robust implementation is challenging and crucial for the overall performance of an analysis.

In the following we would like to highlight the importance of a robust association of adjacent patches by showing an example presented in [37, 38]. The basic setting of the problem is shown in Fig. 58a. This benchmark for geometric nonlinear shell analysis describes a cantilever that is subjected to an end moment. If the maximal moment M_{max} is applied, the cantilever deforms to a closed circular ring. Figure 58b illustrates the numerical solution of this problem for various parameterizations. Note the different level of complexity along the edges of adjacent patches. It clearly demonstrates the vast diversity of situations that my occur in case of multipatch geometries even if they represent the same geometry.

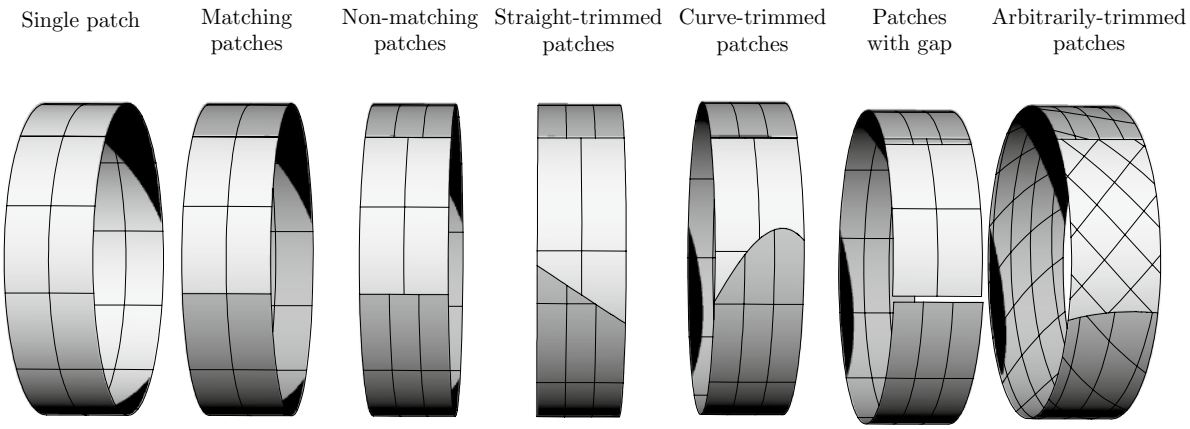
Another important aspect studied by this example is the influence of the gap and overlap size between patches. Consider the geometrical discretization illustrated in Fig. 59. A gap–overlap function f is introduced to specify a user-defined inaccuracy along the curved intersection. Positive and negative values of f represent gaps and overlaps, respectively. The trimming curves are linked by the point inversion algorithm as described before. The resulting vertical displacements at the cantilever's end u_{Tip} of representations with different f are related to a reference solution u_{ref} obtained with $f = 0$. The difference is calculated by

$$d_1 = \left| u_{Tip}(f) - u_{ref} \right|, \tag{119}$$

and the related results are summarized in Fig. 60. Based on the corresponding graph, it can be concluded that small gaps which are within CAD tolerance, i.e., 0.001 units, barely influence the quality of the simulation. The different behavior of gaps and overlaps can be explained by the minimal distance computation: in contrast to gaps, the assignment of points of the slave curve to the master curve is not unique in case of overlaps.



(a) Test setting



(b) Results for various parameterizations with $M = M_{\max}$

Fig. 58 Different geometry models for analyzing a cantilever subjected to an end moment: **a** definition of the problem and **b** resulting solutions. In **b**, different *gray scales* indicate the distinct patches. Note the various complexities of the connection of adjacent patches. (Courtesy of Breitenberger [37, 38])

5.3.3.3 Continuity Considerations The continuity along the intersection of two trimmed patches is usually not higher than C^0. The construction of corresponding C^0 isogeometric spaces with optimal approximation properties is well understood for conforming parameterizations [299]. Brivadis et al. [39] showed this also for weakly imposed C^0 conditions. However, their isogeometric mortar method focuses on regular patches and a modification of the basis functions at the boundary is required to obtain stability, if the same degree is used for the primal and dual spaces. Although the influence of non-matching interfaces is discussed as well, the application in the context of trimmed surfaces has yet to be investigated in more detail.

The construction of smooth isogeometric spaces for trimmed models is an even more complicated open topic. In fact, smooth isogeometric spaces on unstructured geometries are a challenging and open problem in general [141, 296]. Locking effects may occur even for regular planar multipatch configurations [63, 150]. At this point, it should be noted that T-splines or subdivision surfaces provide geometric models which are globally smooth almost everywhere. Nevertheless, these representations seem to lack

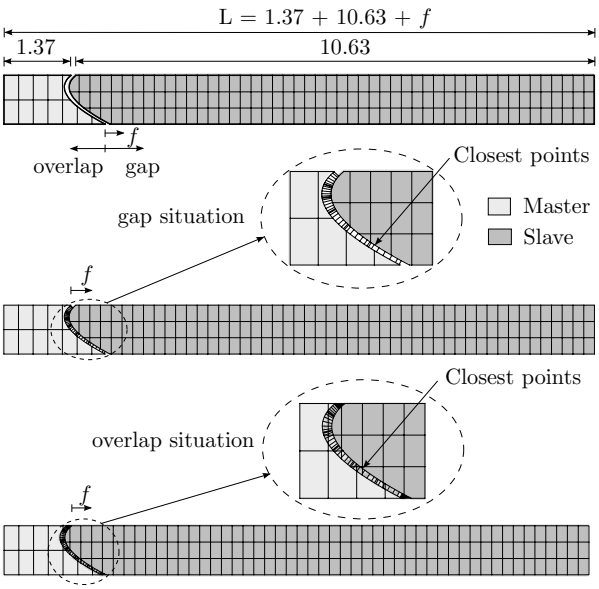


Fig. 59 Geometry representation and definition of the gap–overlap parameter f for the investigation of the effect of non-watertight geometries on numerical results. (Courtesy of Michael Breitenberger [37, 38])

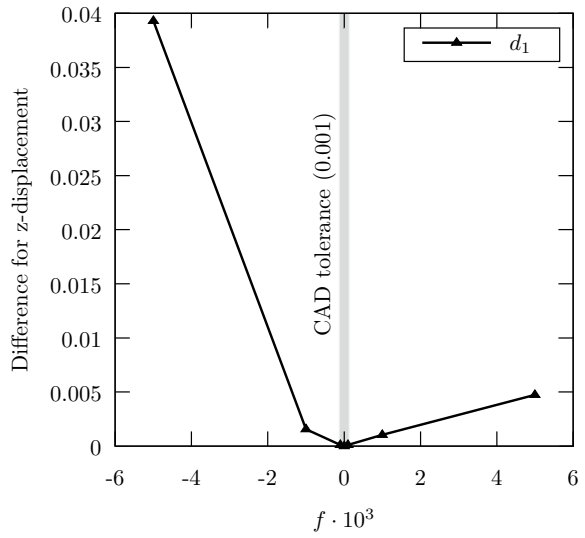


Fig. 60 Comparison of the relative vertical displacement d_1 related to the gap–overlap parameter f. Gaps and overlaps are indicated by positive and negative values, respectively. The *gray area* of the diagram indicates the default tolerance of the CAD software used. (Courtesy of Breitenberger [37, 38])

optimal approximation properties due to the existence of extraordinary vertices [145, 212].

5.3.4 Stabilization

A trimmed basis contains basis functions which are cut by the trimming curve and exist only partially within the valid area $\mathcal{A}^{\mathrm{v}}$. In order to clarify the problem statement, Fig. 61 illustrates a trimmed univariate basis. It should be noted that the Greville abscissae of cut basis functions may be located outside of $\mathcal{A}^{\mathrm{v}}$. In the example given, this is the case for $B_{4,2}$. These points cannot be used for collocation or spline interpolation problems, despite the fact that they are the preferred choice for setting up a stable system of equations (see Sect. 2.3). Furthermore, the support of cut basis functions may be arbitrary small, e.g., this would be the case for $B_{4,2}$ as the trimming location t approaches the knot value 2. Thus, the condition number of the resulting system matrices can become very large. In other words, a trimmed basis is not guaranteed to be stable.

In order to emphasize this stability issue an interpolation problem is examined: a given function

$$f(u, v) = \frac{1}{\sqrt{(-1.2-u)^2 + (-1.2-v)^2}}, \tag{120}$$

shall be approximated by a B-spline surface $\boldsymbol{S}_h$. They agree at k interpolation points $\bar{\boldsymbol{x}}_{i,j} = (\bar{u}_i, \bar{v}_j)^{\intercal}$, where k represents the total number of bivariate basis functions involved. The further components of the corresponding system of equations are the unknown coefficients $c_{i,j}$ and the bivariate spline collocation matrix $\mathbf{A}$. The matrix is defined by

$$\mathbf{A}[i + j \cdot J,\, m + n \cdot J] = B_{i,p}(\bar{u}_m) B_{j,q}(\bar{v}_n), \tag{121}$$

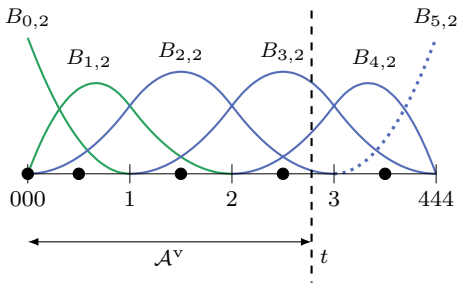


Fig. 61 Univariate basis trimmed at a parameter t. There are basis functions which are fully inside (*green*), partially inside (*blue*), and completely outside (*dotted*) of $\mathcal{A}^{\mathrm{v}}$. The Greville abscissae of the considered basis functions are marked by *circles*. (Color figure online)

with $i, m = 0, \dots, I$ and $j, n = 0, \dots, J$, where I and J are the number of basis functions in each parametric directions. The initial parameter space is given by an open knot vector with a uniform discretization from -1 to 1 in both directions, i.e., $u, v \in [-1, 1]$, and the knot span size is specified by $h = 0.125$. A trimming parameter $t \in [0.5, 1)$ determines the square domain $\mathcal{A}^{\mathrm{v}} \in [-1, t]^2$ considered for the interpolation problem. The interpolation points $\bar{\boldsymbol{x}}$ of cut basis functions may have to be shifted into $\mathcal{A}^{\mathrm{v}}$. Exterior basis functions that are completely outside of $\mathcal{A}^{\mathrm{v}}$ are not involved in the interpolation process. The quality and stability of the approximation $\boldsymbol{S}_h$ are specified by the relative interpolation error measured in the L_2-norm $\|\epsilon_{rel}\|_{L_2}$ as well as the condition number of the spline collocation matrix $\kappa(\mathbf{A})$. The results are summarized in Fig. 62 for various degree with $p = q$.

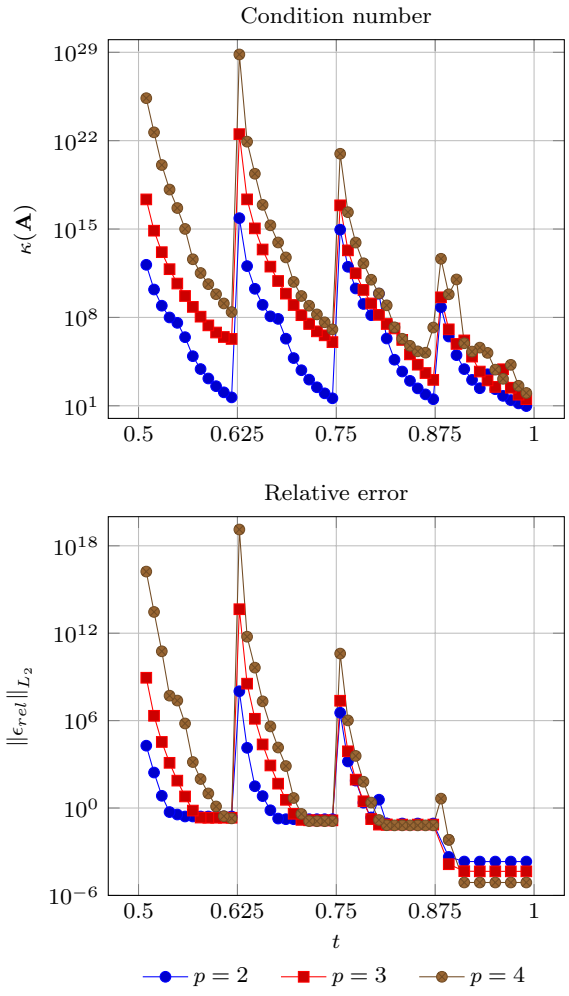


Fig. 62 Condition number $\kappa(\mathbf{A})$ and relative interpolation error $\|\epsilon_{rel}\|_{L_2}$ of the bivariate basis for several degrees p in both parametric directions related to the trimming parameter t. The subdivision of the *horizontal axis* corresponds to the knot values of the trimmed basis

It can be observed that the condition number of $\mathbf{A}$ is considerably influenced by the trimming parameter t. In particular, a peak is reached as soon as t approaches a knot value, i.e., a support of cut basis functions becomes very small. Furthermore, the approximation quality is affected. The peaks of the relative error $\|\epsilon_{rel}\|_{L_2}$ near knot values are in fact disastrous. Hence, it is evident that the straightforward application of a trimmed basis negatively affects the condition number and subsequently the quality of the approximation.

The stability aspect of local approaches for the analysis of trimmed geometries has scarcely been considered in previous works. It is worth noting that Nitsche formulations may incorporate parameters which take cut elements into account, see e.g., [42, 80, 289]. A method-independent alternative that exploit the properties of B-splines is outlined in Sect. 6.

5.4 Summary and Discussion

Various approaches to incorporate trimmed geometries into an analysis have been described in this review. While Sect. 5.1 addresses an early attempt which combines trimmed patches with Lagrange interpolation, recent research is the focus of the subsequent Sects. (5.2) and (5.3). To recapitulate the findings of the current approaches: there are two fundamentally different philosophies to deal with trimmed models. One seeks to resolve the deficiencies of trimmed models by a reconstruction of the geometric representation. This is performed as a preprocessing step before the actual analysis. Since these procedures affect entire patches and their connection, they are referred to as global approaches in this work. The other philosophy is to accept the flaws of trimmed models, implying that the analysis has to be adaptable enough to cope with them. This capability is accomplished by treating the occurring trimming situations on the knot span level. Hence, we classify such techniques as local approaches.

Global approaches address the core of the problem and aim to solve it at its origin. In fact, they are similar to the remodeling schemes of CAGD outlined in Sect. 3.4. They share the same shortcomings such as an increased number of control points and the dependence on a four-sided domain if regular tensor product surfaces are used for the reconstruction. It can be argued that global approaches are more related to CAGD than analysis. Consequently, their success is also determined by the acceptance in the design community. However, a compelling global scheme could eventually lead to design models which can be directly applied not only to analysis but *all* downstream applications, which is the holy grail of the trimming problem.

Local approaches focus on enhancing the analysis and thus, may seem more feasible for researchers in the field of computational mechanics. In fact, the majority of the publications on isogeometric analysis of trimmed geometries employ such concepts. There is a close relation to fictitious domain, or immersed, methods since the trimmed parameter space is used as a background parameterization. Hence, similar challenges have to be addressed: (i) detection of elements cut be the trimming curve, (ii) special integration schemes for these elements, (iii) weak coupling of adjacent patches, and (iv) the stability issue induced by the trimmed basis. The main difference is that additional effort has to be made to associate the degrees of freedom of adjacent patches, keeping in mind that their intersections possess non-matching parameterizations, gaps, and overlaps. These distinct tasks are clearly separated from each other. For example, weak coupling is mandatory for finite element methods but may be neglected if a boundary element method is applied. Most researchers have drawn their attention to the integration of cut elements. The application of weak formulations has also been addressed by several authors. On the other hand, the stability of a trimmed basis and the robust association of adjacent patches are barely discussed in the literature, despite the fact that the latter task is crucial for the analysis of practical design models. Another issue of using a trimmed basis for the analysis is that the Greville abscissae of cut basis functions are not guaranteed to be located inside of the domain of interest. Consequently, an application to interpolation and collocation methods requires further considerations. However, the modular structure of local approaches is indeed a benefit compared to global approaches which require a self-contained concept which becomes more and more sophisticated with its capabilities.

6 Stabilization of a Trimmed Basis

There are two reasons for presenting a distinct section on the stabilization of trimmed parameter spaces: first and foremost, we want to draw attention to this issue which has been scarcely discussed so far, and, in addition, some of our recent research is focused on this topic allowing a more detailed observation of it. The general problem statement has already been given in Sect. 5.3.4, where it has been demonstrated that basis functions cut by a trimming curve can yield ill-conditioned system matrices. Further, Greville abscissae of such basis functions may be outside of the valid domain and thus, they cannot be applied to methods which employ these points like isogeometric collocation [11, 258]. In order to identify the troublesome components, we classify the basis functions of a trimmed parameter space as stable, degenerate, or exterior. The support of the latter is completely outside of the valid domain $\mathcal{A}^{\mathrm{v}}$ and hence, it can be neglected for the analysis. The distinguishing feature of the other types is that the Greville abscissae of stable B-splines are within $\mathcal{A}^{\mathrm{v}}$ whereas the Greville abscissae of degenerate ones are outside of $\mathcal{A}^{\mathrm{v}}$.

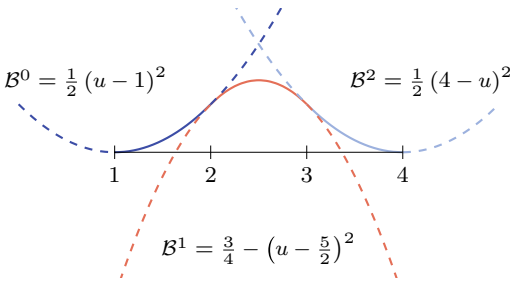


Fig. 63 Polynomial segments $\mathcal{B}^s$ of a B-spline. The extensions of the segments are indicated by *dashed lines*

The following stabilization scheme resolves the issues induced by degenerate basis functions in a simple and flexible manner. The concept is referred to as *extended B-splines*. Originally, these splines have been developed by Höllig and co-workers in the context of a B-spline based fictitious domain method [124–127]. Here, the main aspects of extended B-splines are outlined based on the findings provided in [199, 202].

6.1 Definition of Extended B-splines

We start the description of extended B-splines by recalling two fundamental properties of conventional B-spline: (i) B-splines $B_{i,p}$ are represented by a set of polynomial segments $\mathcal{B}_i^s$ and (ii) B-splines form a basis of a space $\mathbb{S}_{p,\Xi}$ which contains every piecewise polynomial $f_{p,\Xi}$ of degree p over a knot sequence Ξ. The former property is illustrated in Fig. 63. It should be noted that each polynomial segment $\mathcal{B}^s$ may be *extended* beyond its associated knot span s. With this in mind, it is straightforward to grasp the essential idea of extended B-splines, namely to re-established the stability of a trimmed basis by substituting degenerate, and therefore potentially unstable, B-splines by extensions of stable ones. These extensions can be exactly represented by the basis since they are within $\mathbb{S}_{p,\Xi}$ by definition.

The overall construction procedure of extended B-splines is summarized in Fig. 64. Firstly, it is determined if the Greville abscissae of non-exterior B-splines are located inside or outside of $\mathcal{A}^{\mathrm{v}}$. In the latter case the basis function is labeled as degenerate and the corresponding index is stored in the index-set $\mathbb{J}$. Secondly, the polynomial segments of trimmed knot spans are replaced by the extensions of the polynomial segments of the closest non-trimmed knot span that contains stable B-splines only. These extensions together with the polynomial segments of the non-trimmed knot spans form the extended

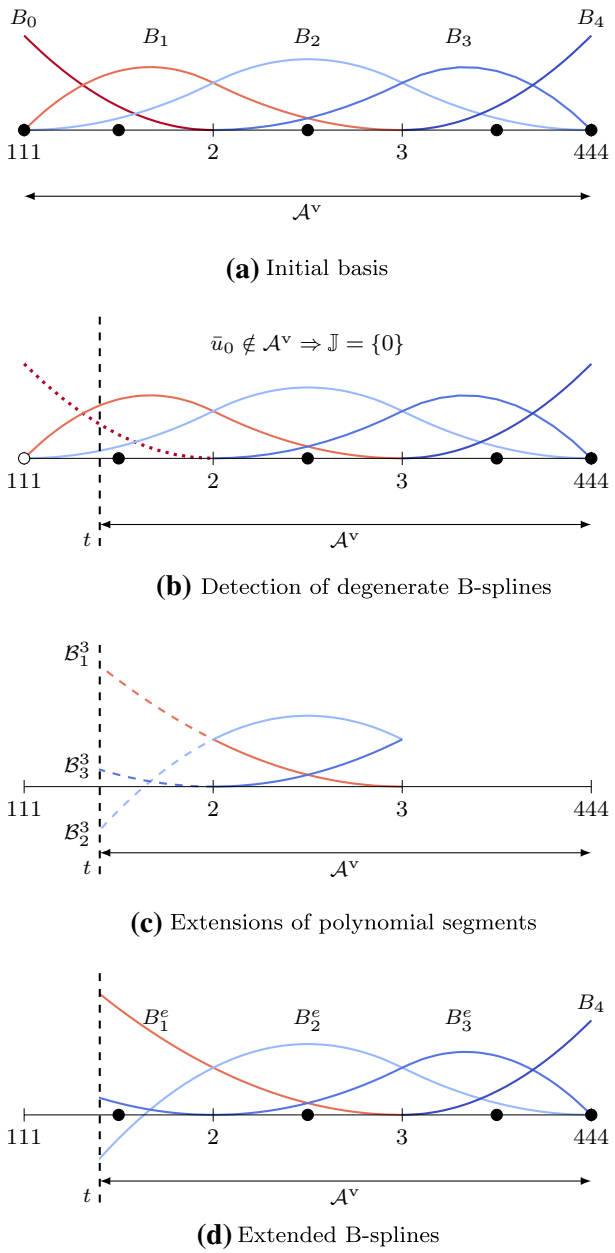


Fig. 64 Basic procedure to get from **a** conventional to **d** extended B-splines: **b** determination of degenerate B-splines and substitution of trimmed polynomial segments by **c** extensions of non-trimmed ones

B-spline basis. The final step is to represent the extended B-splines by a linear combination of the original B-splines. An extended B-spline is defined by

$$B_{i,p}^e = B_{i,p} + \sum_{j \in \mathbb{J}_i} e_{i,j} B_{j,p}, \tag{122}$$

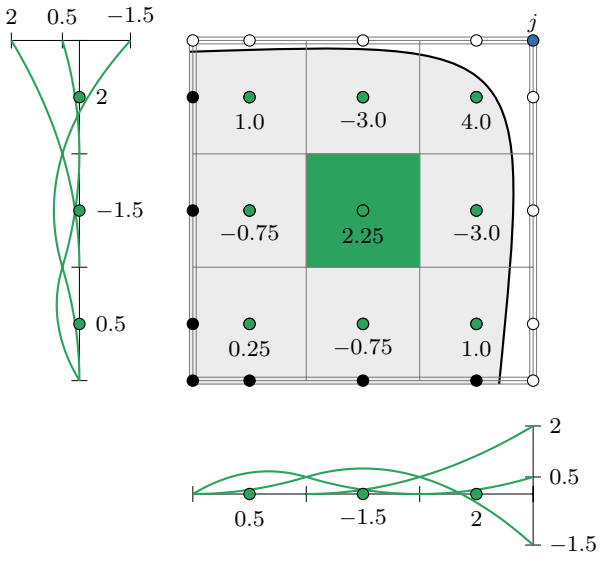


Fig. 65 The construction of bivariate extrapolation weights $e_{i,j}$ for a biquadratic basis. Stable B-splines are marked by *black* and *green circles*. The shown values of $e_{i,j}$ are related to the degenerate basis function marked by the *blue circle* in the upper right corner of the parameter space. B-splines of the closest non-trimmed knot span are indicated by *green circles*. (Color figure online)

where $B_{i,p}$ is the stable B-spline that provides the extension and $\mathbb{J}_i$ is the index-set of all degenerate B-splines related to the current $B_{i,p}^e$. The extrapolation weights $e_{i,j}$ can be determined by solving an interpolation problem. To be precise, the given polynomial function f of the extension over the trimmed knot span shall be represented by means of the basis functions $B_{j,p}$. The coefficient $e_{i,i}$ is trivial since $B_{i,p}^e$ must be equal to $B_{i,p}$ within the non-trimmed knot spans, thus $e_{i,i} = 1$.

Spline interpolation as described in Sect. 2.3 is not optimal to compute $e_{i,j}$ because the Greville abscissae of $B_{j,p}$ are not located within the trimmed knot span in general. Hence, a *quasi interpolation* scheme is preferred which allows an explicit computation of B-spline coefficients. In particular, the so-called *de Boor–Fix* or dual functional $\lambda_{j,p}$ [34, 35] is used: for any piecewise polynomial $f \in \mathbb{S}_{p,\Xi}$,

$$f = \sum_{j=0}^{J-1} \lambda_{j,p}(f) B_{j,p}, \tag{123}$$

with

$$\lambda_{j,p}(f) = \frac{1}{p!} \sum_{k=0}^{p} (-1)^k \psi_{j,p}^{(p-k)}(\mu_j) f^{(k)}(\mu_j), \tag{124}$$

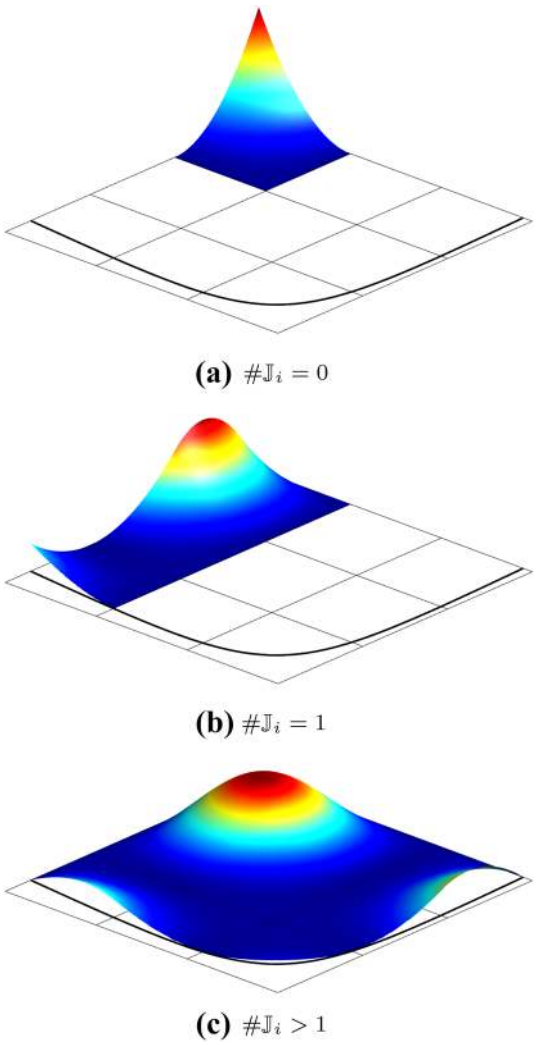


Fig. 66 Bivariate extended B-splines $B^e_{i,p}$ with various cardinalities of the index-set $\mathbb{J}_i$ which indicates the number of related degenerate B-splines. Note that **a** is in fact a conventional B-spline, i.e., $B^e_{i,p} \equiv B_{i,p}$, since $\mathbb{J}_i$ is empty

$$\psi_{j,p}(u) = \prod_{m=1}^{p} \left(u - u_{j+m}\right). \tag{125}$$

The evaluation point μ_j can be chosen arbitrarily within $[u_j, u_{j+p+1}]$. Substituting f of Eq. (124) by $\mathcal{B}^s_i$ yields the extrapolation weights

$$e_{i,j} = \frac{1}{p!} \sum_{k=0}^{p} (-1)^k \psi^{(p-k)}_{j,p}(\mu_j) \mathcal{B}^{s^{(k)}}_i(\mu_j). \tag{126}$$

When the polynomials $\psi_{j,p}$ and $\mathcal{B}^s_i$ are expressed in power basis form

$$\psi_{j,p}(u) = \sum_{k=0}^{p} \beta_k u^k \quad \text{and} \quad \mathcal{B}^s_i(u) = \sum_{k=0}^{p} \alpha_k u^k, \tag{127}$$

expression (126) simplifies to

$$e_{i,j} = \frac{1}{p!} \sum_{k=0}^{p} (-1)^k (p-k)! \beta_{p-k} k! \alpha_k. \tag{128}$$

The interested reader is referred to [202] for details on the conversion to power basis form and further details regarding the evaluation of the dual functional. In case of a uniform knot vector, a simplified formula can be derived which solely relies on the indices of the B-splines involved, see e.g., [124].

Bivariate extrapolation weights are simply obtained by the tensor product of their univariate counterparts calculated for each parametric direction as illustrated in Fig. 65. Note that the degenerate B-spline is distributed to $(p+1)(q+1)$ stable ones.

6.2 Properties of Extended B-splines

Extended B-splines inherit most essential properties of conventional B-splines [124, 125, 127]. They are linearly independent and polynomial precision is guaranteed. Thus, they form a basis for a spline space. Each knot span has exactly $p+1$ non-vanishing basis functions which span the space of all polynomials of degree $\leqslant p$ over $\mathcal{A}^{\mathrm{v}}$. Furthermore, approximation estimates have the same convergence order as conventional B-splines. Extended B-splines have local support in the sense that only B-splines near the trimming curve are subjected to the extension procedure. The actual size of the affected region depends on the fineness of the parameter space, the degree of its basis functions, and the number of degenerate $B_{j,p}$ related to the stable $B_{i,p}$. The latter is given by the cardinality of the corresponding index-set $\#\mathbb{J}_i$. Figure 66 illustrates various examples of extended B-splines. The basis function shown in Fig. 66a is in fact a conventional B-spline since it is far away from the trimming curve.

However, there are also some differences. It is important to note that the extrapolation weights may be *negative*, hence the evaluation of extended B-splines may lead to negative values. Conventional B-splines, on the other hand, are strictly non-negative. This property is exploited in some contact formulations [295] and structural optimization [210], for instance. In such cases, the application

of extended B-splines requires further considerations. The main difference in favor of extended B-splines is the *stability* of the corresponding basis. The condition number of a system is independent of the location of the trimming curve due to the substitution of B-splines with small support. Another benefit is that all Greville abscissae are located within $\mathcal{A}^{\mathrm{v}}$ by construction.

6.3 Assembling

Extended B-splines can be applied to an analysis in a very convenient manner. Suppose we have a linear system of n equations, one for each stable B-spline, set up by all basis functions m which are at least partially inside $\mathcal{A}^{\mathrm{v}}$. This yields

$$\mathbf{K}\mathbf{u} = \mathbf{f} \quad \text{where} \quad \mathbf{u}, \mathbf{f} \in \mathbb{R}^n \quad \text{and} \quad \mathbf{K} \in \mathbb{R}^{n \times m}, \tag{129}$$

with $m > n$. Upon this point, only conventional B-splines have been used to compute the system matrix $\mathbf{K}$. In other words, $\mathbf{K}$ is set up as usual but equations related to degenerate B-splines are neglected. In order to obtain a square matrix an *extension matrix* $\mathbf{E} \in \mathbb{R}^{m \times n}$ is introduced [124]. This sparse matrix $\mathbf{E}$ contains all extrapolation weights $e_{i,j}$ including the trivial ones, i.e., $e_{i,i} = 1$. The transformation of the original to the stable extended B-spline basis is performed by multiplying the extension matrix to the system matrix. The resulting stable system

$$\mathbf{K}_e\mathbf{u} = \mathbf{f} \quad \text{with} \quad \mathbf{K}_e = \mathbf{K}\mathbf{E}, \quad \mathbf{K}_e \in \mathbb{R}^{n \times n}, \tag{130}$$

is subsequently solved and the obtained solution $\mathbf{u}$ corresponds to the extended B-splines of the unknown field. In case of multi-patch geometries, the extrapolation weights $e_{i,j}$ of each patch have to be assembled to $\mathbf{E}$ with respect to the global degrees of freedom. The application of the extension operator is particularly convenient, if extended B-splines are added to an existing code.

6.4 Application to NURBS

The stabilization described is tailored to B-spline functions where it is exploited that the extensions of any polynomial segment $\mathcal{B}_i^s$ can be exactly represented by a linear combination of basis functions of the trimmed knot span. In case of NURBS this property is not guaranteed due to the local influence of the weights. In order to apply extended B-splines to a trimmed NURBS basis, two different approaches may be used: (i) conversion of the CAGD model to a B-spline representation or (ii) application of an independent field approximation [199–201]. The benefit of the latter is that it allows performing the analysis based on the original NURBS model without any geometrical approximations. The key idea of independent field approximation is to use different basis functions for the representation of the geometry and the approximations of the physical fields. Hence, conventional B-splines can be used for the discretization of the field variables over NURBS patches. This allows the straightforward application of extended B-splines. In addition, the combination of NURBS for the geometry description and B-splines for the approximations has been shown to be more efficient [199] and does not lead to a loss of accuracy [184, 201, 299]. There is one caveat: independent fields are inconsistent with the isoparametric concept in mechanics and can upset the precise representation of constant strain states and rigid body motions [139].

6.5 Assessment of Stability

In order to assess the approximation quality and stability of extended B-splines the same interpolation problem as in Sect. 5.3.4 is considered. Again, the relative interpolation

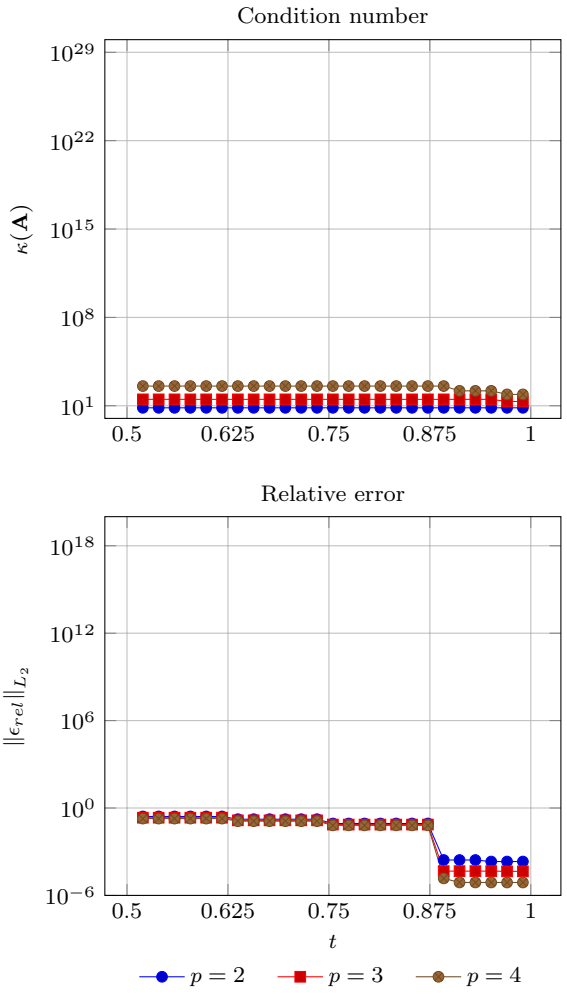


Fig. 67 Spline interpolation problem with extended B-splines for several degrees p. The condition number $\kappa(\mathbf{A})$ and the relative interpolation error $\|\epsilon_{rel}\|_{L_2}$ are related to the trimming parameter t. The labels of the *horizontal axis* indicate knots of the trimmed basis

error measured in the L_2-norm $\|\epsilon_{rel}\|_{L_2}$ and the condition number of the spline collocation matrix $\kappa(\mathbf{A})$ are examined. The results are summarized in Fig. 67.

Comparing Figs. 62 and 67 shows the significant improvement of extended B-splines. If extended B-splines are used, $\kappa(\mathbf{A})$ hardly changes and is independent of the trimming parameter t. In other words, the extended B-spline basis is stable. Consequently, the approximation quality is significantly improved. In Fig. 67, the reduction of the approximation accuracy occurs only due to the reduction of the degrees of freedom n, i.e., number of extended B-spline, as the trimming parameter $t \to 0.5$.

6.6 Summary and Discussion

The concept of extended B-splines substitutes unstable basis functions by extensions of stable ones. It is established in a very flexible manner and requires only the presence of a sufficient number of stable basis functions. In general, this requirement is non-restrictive and can be fulfilled by refinement of the basis. Still, it may be an issue if the design object contains very small fillets. Only B-splines close to the trimming curve are affected by the stabilization procedure. The number of B-splines depends on the distance of the trimming curve to the knot span which provides the stable B-splines. This correlates with the degree p of the basis function since the size of its support extends over $p + 1$ knot spans.

7 Final Remarks and Conclusions

The present work accumulates several topics related to the treatment of trimmed models and the interoperability problem between CAD and downstream applications in general.

It is apparent that trimming is a fundamental technique for geometric design. Most importantly, it enables the computation of intersections between free-form surfaces. However, intersection curves cannot be determined exactly, which leads to various problems. As a result, an intersection is usually approximated by several independent curves, one in model space and one in the parameter space of each surface involved. Their images in model space do not coincide and there is no link between these curves. The resulting gaps and overlaps between the surfaces yield robustness issues due to a lack of exact topological consistency. These problems are still unresolved, despite the fact that they have been the focus of an enormous amount of research.

Since the robustness issues of trimmed models are particularly crucial for downstream applications, the exchange of CAD data is examined as well. Neutral exchange standards seem to be the most comprehensive strategy, but it is important to note that all translations lead to loss of information. Moreover, the capabilities of the various exchange formats are *not* equivalent. It is demonstrated that STEP is superior to IGES. STEP should be preferred in general and especially if the topology of a model needs to be extracted.

In the context of analysis, the current approaches can be divided into two different philosophies. On the one hand, global approaches aim to fix the problems of trimmed models before the simulation by a reconstruction of the geometric representation. Using local approaches, on the other hand, trimmed models are directly employed, but the analysis has to be enhanced in order to deal with all the flaws of the geometric representation. It may be argued that the former addresses the issue from a CAD point of view, whereas the latter utilizes an analysis perspective. The fact that the problem can be tackled by these diverse directions emphasizes the central role of trimmed models for the integration of design and analysis.

The main conclusions of the present review can be summarized as follows:

- Trimming seems to be a simple and benign procedure at first glance, but its consequences are profound.
- Robustness issues are the price for the flexibility of trimmed models.
- Flaws of trimmed models are usually hidden from the user, but surface as soon as they are applied to a downstream application.
- To overcome these issues is a crucial aspect regarding the integration of design and analysis.
- The success of CAD data exchange depends on the quality of the design model *and* the capability of the exchange format.
- There is no canonical way to deal with trimmed models, neither in analysis nor in design, at least so far.

It is hoped that this review provides a helpful introduction to the topic and an impetus for further research activities. There are indeed several open issues worth exploring: optimization of the reparameterization of the global approach based on isocurves, assessment of the affect of gaps on the analysis in case of local schemes, and application to isogeometric collocation, just to name a few. In general, the step from academic examples to practical multipatch models is perhaps the most challenging task. Robust algorithms should be able to take the tolerances of a design model into account. On the other hand, it may be an unrealistic aim to find a solution that can deal with every possible trimmed geometry. Similar to the quality of conventional meshes,

an isogeometric simulation is effected by the quality of the design model. Hence, the specification of distinct properties that classify a design model to be analysis-suitable are needed so that a designer can get a direct feedback if a model requires an improvement—providing the right information to the right person at the right time. We close this review by emphasizing that a holistic treatment of the engineering design process requires the aligned efforts of both the design and the analysis communities.

Acknowledgements Open access funding provided by Austrian Science Fund (FWF). We thank Michael Breitenberger and Ben Urick for their useful advise and fruitful discussions. This research was supported by the Austrian Science Fund (FWF): J 3884-N32, and the Office of Naval Research (ONR): N00014-17-1-2039. This support is gratefully acknowledged.

Compliance with Ethical Standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix: Exchange Data File Examples

The source files of the neutral exchange format example presented in Sect. 4.2.3 are given in this section. The models have been constructed with the commercial CAD software `Rhinoceros 5`.

It should be pointed out that the IGES examples, i.e., Files 1 and 2, provide the *same* information although the topological data of the two models was different before the extraction procedure. In particular, these files differ only in the representation of some floating point values, e.g., 0.0D0 and 8.881D-16, and the sequence of the numbering of a few parametric data entities, e.g., 0000029P of File 1 is equal to 0000037P of File 2. The corresponding STEP examples with the correct topology data are given in Files 3 and 4.

The interested reader is referred to the homepage of `STEP Tools, Inc.`[21] for further examples of STEP files covering various application protocols.

File 1: IGES trimming example – surface model

```
                                                                        S      1
1H,,1H;,,                                                               G      1
65HC:\Users\Marussig\Desktop\LinuxSync\simpleTrimmingExampleTrim.igs,    G      2
26HRhinoceros ( Sep 27 2012 ),31HTrout Lake IGES 012 Sep 27 2012,       G      3
32,38,6,308,15,                                                         G      4
,                                                                       G      5
1.0D0,2,2HMM,1,0.254D0,13H160826.214535,                                G      6
0.001D0,                                                                G      7
5D0,                                                                    G      8
,                                                                       G      9
,                                                                       G     10
10,0,13H160826.214535;                                                  G     11
     314       1       0       0       0       0       0000002000D      1
     314       0       1       1       0       0       0   COLOR       0D      2
     406       2       0       0       1       0       0       000003000D      3
     406       0      -1       1       3       0       0LEVELDEF       0D      4
     128       3       0       0       1       0       0       000010000D      5
     128       0      -1       7       8       0       0 TrimSrf       0D      6
     126      10       0       0       1       0       0       000010000D      7
     126       0      -1       5       1       0       0 TrimSrf       1D      8
     126      15       0       0       1       0       0       000010500D      9
     126       0      -1       4       1       0       0 TrimSrf       1D     10
     126      19       0       0       1       0       0       000010000D     11
     126       0      -1       5       1       0       0 TrimSrf       2D     12
     126      24       0       0       1       0       0       000010500D     13
     126       0      -1       5       1       0       0 TrimSrf       2D     14
     126      29       0       0       1       0       0       000010000D     15
     126       0      -1       5       1       0       0 TrimSrf       3D     16
     126      34       0       0       1       0       0       000010500D     17
     126       0      -1       4       1       0       0 TrimSrf       3D     18
     126      38       0       0       1       0       0       000010000D     19
     126       0      -1       4       1       0       0 TrimSrf       4D     20
     126      42       0       0       1       0       0       000010500D     21
     126       0      -1       3       1       0       0 TrimSrf       4D     22
     141      45       0       0       1       0       0       000010000D     23
     141       0      -1       1       0       0       0 TrimSrf       1D     24
     143      46       0       0       1       0       0       000000000D     25
     143       0      -1       1       0       0       0 TrimSrf       0D     26
     128      47       0       0       1       0       0       000010000D     27
     128       0      -1       4       8       0       0 TrimSrf       0D     28
     126      51       0       0       1       0       0       000010000D     29
     126       0      -1       3       1       0       0 TrimSrf       1D     30
     126      54       0       0       1       0       0       000010500D     31
     126       0      -1       6       1       0       0 TrimSrf       1D     32
     126      60       0       0       1       0       0       000010000D     33
     126       0      -1       2       1       0       0 TrimSrf       2D     34
     126      62       0       0       1       0       0       000010500D     35
     126       0      -1       2       1       0       0 TrimSrf       2D     36
     126      64       0       0       1       0       0       000010000D     37
     126       0      -1       2       1       0       0 TrimSrf       3D     38
     126      66       0       0       1       0       0       000010500D     39
     126       0      -1       2       1       0       0 TrimSrf       3D     40
     141      68       0       0       1       0       0       000010000D     41
     141       0      -1       1       0       0       0 TrimSrf       1D     42
     143      69       0       0       1       0       0       000000000D     43
     143       0      -1       1       0       0       0 TrimSrf       0D     44
314,0.0,0.0,0.0,20HRGB(   0,    0,    0 );                              0000001P      1
406,2,1,7HDefault;                                                      0000003P      2
128,1,1,1,1,0,0,1,0,0,0.0D0,0.0D0,7.071067811865475D0,                  0000005P      3
7.071067811865475D0,0.0D0,0.0D0,9.999999999999998D0,                    0000005P      4
9.999999999999998D0,1.0D0,1.0D0,1.0D0,1.0D0,5.0D0,0.0D0,-5.0D0,         0000005P      5
8.881784197001252D-16,4.999999999999999D0,-5.0D0,5.0D0,0.0D0,           0000005P      6
4.999999999999998D0,8.881784197001252D-16,4.999999999999999D0,         0000005P      7
4.999999999999998D0,0.0D0,7.071067811865475D0,0.0D0,                    0000005P      8
9.999999999999998D0;                                                    0000005P      9
126,1,1,1,0,1,0,0.0D0,0.0D0,7.071067811865475D0,                        0000007P     10
7.071067811865475D0,1.0D0,1.0D0,5.0D0,0.0D0,                            0000007P     11
8.881784197001252D-16,8.881784197001252D-16,4.999999999999999D0,        0000007P     12
8.881784197001252D-16,0.0D0,7.071067811865475D0,0.0D0,0.0D0,            0000007P     13
1.0D0;                                                                  0000007P     14
126,1,1,1,0,1,0,0.0D0,0.0D0,7.071067811865475D0,                        0000009P     15
7.071067811865475D0,1.0D0,1.0D0,0.0D0,5.0D0,0.0D0,                      0000009P     16
7.071067811865475D0,5.000000000000001D0,0.0D0,0.0D0,                    0000009P     17
7.071067811865475D0,0.0D0,0.0D0,1.0D0;                                  0000009P     18
126,1,1,1,0,1,0,5.000000000000001D0,5.000000000000001D0,                0000011P     19
9.999999999999998D0,9.999999999999998D0,1.0D0,1.0D0,                    0000011P     20
8.881784197001252D-16,4.999999999999999D0,8.881784197001252D-16,        0000011P     21
8.881784197001252D-16,4.999999999999999D0,4.999999999999998D0,         0000011P     22
5.000000000000001D0,9.999999999999998D0,1.0D0,0.0D0,0.0D0;              0000011P     23
126,1,1,1,0,1,0,5.000000000000001D0,5.000000000000001D0,                0000013P     24
9.999999999999998D0,9.999999999999998D0,1.0D0,1.0D0,                    0000013P     25
7.071067811865475D0,5.000000000000001D0,0.0D0,                          0000013P     26
7.071067811865475D0,9.999999999999998D0,0.0D0,                          0000013P     27
5.000000000000001D0,9.999999999999998D0,0.0D0,0.0D0,1.0D0;              0000013P     28
126,1,1,1,0,1,0,-7.071067811865475D0,-7.071067811865475D0,              0000015P     29
-0.0D0,-0.0D0,1.0D0,1.0D0,8.881784197001252D-16,                        0000015P     30
4.999999999999999D0,4.999999999999998D0,5.0D0,0.0D0,                    0000015P     31
4.999999999999998D0,-7.071067811865475D0,-0.0D0,0.0D0,0.0D0,            0000015P     32
1.0D0;                                                                  0000015P     33
126,1,1,1,0,1,0,0.0D0,0.0D0,7.071067811865475D0,                        0000017P     34
7.071067811865475D0,1.0D0,1.0D0,7.071067811865475D0,                    0000017P     35
9.999999999999998D0,0.0D0,0.0D0,9.999999999999998D0,0.0D0,0.0D0,        0000017P     36
7.071067811865475D0,0.0D0,0.0D0,1.0D0;                                  0000017P     37
126,1,1,1,0,1,0,-9.999999999999998D0,-9.999999999999998D0,              0000019P     38
-5.000000000000001D0,-5.000000000000001D0,1.0D0,1.0D0,5.0D0,            0000019P     39
0.0D0,4.999999999999998D0,5.0D0,0.0D0,8.881784197001252D-16,            0000019P     40
-9.999999999999998D0,-5.000000000000001D0,1.0D0,0.0D0,0.0D0;            0000019P     41
126,1,1,1,0,1,0,0.0D0,0.0D0,4.999999999999997D0,                        0000021P     42
4.999999999999997D0,1.0D0,1.0D0,0.0D0,9.999999999999998D0,0.0D0,        0000021P     43
0.0D0,5.0D0,0.0D0,0.0D0,4.999999999999997D0,0.0D0,0.0D0,1.0D0;          0000021P     44
141,1,3,5,4,7,1,1,9,11,1,1,13,15,1,1,17,19,1,1,21;                     0000023P     45
143,1,5,1,23;                                                           0000025P     46
128,1,1,1,1,0,0,1,0,0,0.0D0,0.0D0,5.0D0,5.0D0,0.0D0,0.0D0,5.0D0,       0000027P     47
5.0D0,1.0D0,1.0D0,1.0D0,1.0D0,0.0D0,0.0D0,0.0D0,0.0D0,5.0D0,           0000027P     48
0.0D0,5.0D0,0.0D0,0.0D0,5.0D0,5.0D0,0.0D0,0.0D0,5.0D0,0.0D0,           0000027P     49
5.0D0;                                                                  0000027P     50
```

[21] http://www.steptools.com, 8 2016.

```
126,1,1,1,0,1,0,-7.071067811865475D0,-7.071067811865475D0,       0000029P     51
-0.0D0,-0.0D0,1.0D0,1.0D0,0.0D0,5.0D0,0.0D0,5.0D0,0.0D0,0.0D0,   0000029P     52
-7.071067811865475D0,-0.0D0,0.0D0,0.0D0,1.0D0;                   0000029P     53
126,3,3,1,0,1,0,-7.071067811865475D0,-7.071067811865475D0,       0000031P     54
-7.071067811865475D0,-7.071067811865475D0,-0.0D0,-0.0D0,-0.0D0,  0000031P     55
-0.0D0,1.0D0,1.0D0,1.0D0,1.0D0,5.0D0,0.0D0,0.0D0,                0000031P     56
3.3333333333333334D0,1.666666666666666D0,0.0D0,                  0000031P     57
1.666666666666666D0,3.3333333333333334D0,0.0D0,0.0D0,5.0D0,0.0D0, 0000031P     58
-7.071067811865475D0,-0.0D0,0.0D0,0.0D0,1.0D0;                   0000031P     59
126,1,1,1,0,1,0,-5.0D0,-5.0D0,-0.0D0,-0.0D0,1.0D0,1.0D0,5.0D0,   0000033P     60
0.0D0,0.0D0,0.0D0,0.0D0,0.0D0,-5.0D0,-0.0D0,0.0D0,0.0D0,1.0D0;   0000033P     61
126,1,1,1,0,1,0,0.0D0,0.0D0,5.0D0,5.0D0,1.0D0,1.0D0,0.0D0,5.0D0, 0000035P     62
0.0D0,0.0D0,0.0D0,0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,1.0D0;           0000035P     63
126,1,1,1,0,1,0,0.0D0,0.0D0,5.0D0,5.0D0,1.0D0,1.0D0,0.0D0,0.0D0, 0000037P     64
0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,1.0D0;           0000037P     65
126,1,1,1,0,1,0,0.0D0,0.0D0,5.0D0,5.0D0,1.0D0,1.0D0,0.0D0,0.0D0, 0000039P     66
0.0D0,5.0D0,0.0D0,0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,1.0D0;           0000039P     67
141,1,3,27,3,29,1,1,31,33,1,1,35,37,1,1,39;                      0000041P     68
143,1,27,1,41;                                                   0000043P     69
S0000001G0000011D0000044P0000069                                        T      1
```

File 2: IGES trimming example – solid model

```
                                                                        S      1
1H,,1H;,,                                                               G      1
72HC:\Users\Marussig\Desktop\LinuxSync\simpleTrimmingExampleBooleanIGES.G      2
igs,                                                                    G      3
26HRhinoceros ( Sep 27 2012 ),31HTrout Lake IGES 012 Sep 27 2012,       G      4
32,38,6,308,15,                                                         G      5
,                                                                       G      6
1.0D0,2,2HMM,1,0.254D0,13H160826.214018,                                G      7
0.001D0,                                                                G      8
5D0,                                                                    G      9
,                                                                       G     10
,                                                                       G     11
10,0,13H160826.214018;                                                  G     12
     314       1       0       0       0       0       0       000000200D     1
     314       0       1       1       0       0       0   COLOR       0D     2
     406       2       0       0       1       0       0       000000300D     3
     406       0      -1       1       3       0       0LEVELDEF       0D     4
     128       3       0       0       1       0       0       000010000D     5
     128       0      -1       7       8       0       0   Shell       0D     6
     126      10       0       0       1       0       0       000010000D     7
     126       0      -1       3       1       0       0   Shell       1D     8
     126      13       0       0       1       0       0       000010500D     9
     126       0      -1       4       1       0       0   Shell       1D    10
     126      17       0       0       1       0       0       000010000D    11
     126       0      -1       5       1       0       0   Shell       2D    12
     126      22       0       0       1       0       0       000010500D    13
     126       0      -1       5       1       0       0   Shell       2D    14
     126      27       0       0       1       0       0       000010000D    15
     126       0      -1       5       1       0       0   Shell       3D    16
     126      32       0       0       1       0       0       000010500D    17
     126       0      -1       4       1       0       0   Shell       3D    18
     126      36       0       0       1       0       0       000010000D    19
     126       0      -1       3       1       0       0   Shell       4D    20
     126      39       0       0       1       0       0       000010500D    21
     126       0      -1       3       1       0       0   Shell       4D    22
     141      42       0       0       1       0       0       000010000D    23
     141       0      -1       1       0       0       0   Shell       1D    24
     143      43       0       0       1       0       0       000000000D    25
     143       0      -1       1       0       0       0   Shell       1D    26
     128      44       0       0       1       0       0       000010000D    27
     128       0      -1       4       8       0       0   Shell       0D    28
     126      48       0       0       1       0       0       000010000D    29
     126       0      -1       2       1       0       0   Shell       1D    30
     126      50       0       0       1       0       0       000010500D    31
     126       0      -1       2       1       0       0   Shell       1D    32
     126      52       0       0       1       0       0       000010000D    33
     126       0      -1       2       1       0       0   Shell       2D    34
     126      54       0       0       1       0       0       000010500D    35
     126       0      -1       2       1       0       0   Shell       2D    36
     126      56       0       0       1       0       0       000010000D    37
     126       0      -1       3       1       0       0   Shell       3D    38
     126      59       0       0       1       0       0       000010500D    39
     126       0      -1       6       1       0       0   Shell       3D    40
     141      65       0       0       1       0       0       000010000D    41
     141       0      -1       1       0       0       0   Shell       1D    42
     143      66       0       0       1       0       0       000000000D    43
     143       0      -1       1       0       0       0   Shell       2D    44
314,0.0,0.0,0.0,20HRGB(   0,   0,   0 );                          0000001P     1
406,2,1,7HDefault;                                                0000003P     2
128,1,1,1,1,0,0,1,0,0,0.0D0,0.0D0,7.071067811865475D0,            0000005P     3
7.071067811865475D0,0.0D0,0.0D0,9.999999999999998D0,              0000005P     4
9.999999999999998D0,1.0D0,1.0D0,1.0D0,1.0D0,5.0D0,0.0D0,-5.0D0,   0000005P     5
8.881784197001252D-16,4.999999999999999D0,-5.0D0,5.0D0,0.0D0,     0000005P     6
4.999999999999998D0,8.881784197001252D-16,4.999999999999999D0,   0000005P     7
4.999999999999998D0,0.0D0,7.071067811865475D0,0.0D0,              0000005P     8
9.999999999999998D0;                                              0000005P     9
126,1,1,1,0,1,0,0.0D0,0.0D0,7.071067811865475D0,                  0000007P    10
7.071067811865475D0,1.0D0,1.0D0,5.0D0,0.0D0,0.0D0,0.0D0,5.0D0,   0000007P    11
0.0D0,0.0D0,7.071067811865475D0,0.0D0,0.0D0,1.0D0;                0000007P    12
126,1,1,1,0,1,0,0.0D0,0.0D0,7.071067811865475D0,                  0000009P    13
7.071067811865475D0,1.0D0,1.0D0,0.0D0,5.0D0,0.0D0,               0000009P    14
7.071067811865475D0,5.000000000000001D0,0.0D0,0.0D0,             0000009P    15
7.071067811865475D0,0.0D0,0.0D0,1.0D0;                           0000009P    16
126,1,1,1,0,1,0,5.000000000000001D0,5.000000000000001D0,         0000011P    17
9.999999999999998D0,9.999999999999998D0,1.0D0,1.0D0,0.0D0,5.0D0, 0000011P    18
0.0D0,8.881784197001252D-16,4.999999999999999D0,                 0000011P    19
4.999999999999998D0,5.000000000000001D0,9.999999999999998D0,     0000011P    20
1.0D0,0.0D0,0.0D0;                                               0000011P    21
126,1,1,1,0,1,0,5.000000000000001D0,5.000000000000001D0,         0000013P    22
9.999999999999998D0,9.999999999999998D0,1.0D0,1.0D0,             0000013P    23
7.071067811865475D0,5.000000000000001D0,0.0D0,                   0000013P    24
7.071067811865475D0,9.999999999999998D0,0.0D0,                   0000013P    25
5.000000000000001D0,9.999999999999998D0,0.0D0,0.0D0,1.0D0;       0000013P    26
126,1,1,1,0,1,0,-7.071067811865475D0,-7.071067811865475D0,       0000015P    27
-0.0D0,-0.0D0,1.0D0,1.0D0,8.881784197001252D-16,                 0000015P    28
4.999999999999999D0,4.999999999999998D0,5.0D0,0.0D0,             0000015P    29
4.999999999999998D0,-7.071067811865475D0,-0.0D0,0.0D0,0.0D0,     0000015P    30
1.0D0;                                                           0000015P    31
126,1,1,1,0,1,0,0.0D0,0.0D0,7.071067811865475D0,                 0000017P    32
7.071067811865475D0,1.0D0,1.0D0,7.071067811865475D0,             0000017P    33
9.999999999999998D0,0.0D0,0.0D0,9.999999999999998D0,0.0D0,0.0D0, 0000017P    34
7.071067811865475D0,0.0D0,0.0D0,1.0D0;                           0000017P    35
126,1,1,1,0,1,0,-9.999999999999998D0,-9.999999999999998D0,       0000019P    36
-5.0D0,-5.0D0,1.0D0,1.0D0,5.0D0,0.0D0,4.999999999999998D0,5.0D0, 0000019P    37
0.0D0,0.0D0,-9.999999999999998D0,-5.0D0,1.0D0,0.0D0,0.0D0;       0000019P    38
126,1,1,1,0,1,0,0.0D0,0.0D0,4.999999999999998D0,                 0000021P    39
4.999999999999998D0,1.0D0,1.0D0,1.0D0,0.0D0,9.999999999999998D0,0.0D0, 0000021P 40
0.0D0,5.0D0,0.0D0,0.0D0,4.999999999999998D0,0.0D0,0.0D0,1.0D0;   0000021P    41
141,1,3,5,4,7,1,1,9,11,1,1,13,15,1,1,17,19,1,1,21;              0000023P    42
143,1,5,1,23;                                                    0000025P    43
128,1,1,1,1,1,0,0,1,0,0,0.0D0,0.0D0,5.0D0,5.0D0,0.0D0,0.0D0,5.0D0, 0000027P 44
5.0D0,1.0D0,1.0D0,1.0D0,1.0D0,0.0D0,0.0D0,0.0D0,0.0D0,5.0D0,     0000027P    45
0.0D0,5.0D0,0.0D0,0.0D0,5.0D0,5.0D0,0.0D0,0.0D0,5.0D0,0.0D0,     0000027P    46
5.0D0;                                                           0000027P    47
126,1,1,1,0,1,0,-5.0D0,-5.0D0,-0.0D0,-0.0D0,1.0D0,1.0D0,5.0D0,   0000029P    48
0.0D0,0.0D0,0.0D0,0.0D0,0.0D0,-5.0D0,-0.0D0,0.0D0,0.0D0,1.0D0;   0000029P    49
126,1,1,1,0,1,0,0.0D0,0.0D0,5.0D0,5.0D0,1.0D0,1.0D0,0.0D0,5.0D0, 0000031P    50
0.0D0,0.0D0,0.0D0,0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,1.0D0;           0000031P    51
126,1,1,1,0,1,0,0.0D0,0.0D0,5.0D0,5.0D0,1.0D0,1.0D0,0.0D0,0.0D0, 0000033P    52
0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,1.0D0;           0000033P    53
126,1,1,1,0,1,0,0.0D0,0.0D0,5.0D0,5.0D0,1.0D0,1.0D0,0.0D0,0.0D0, 0000035P    54
0.0D0,5.0D0,0.0D0,0.0D0,0.0D0,5.0D0,0.0D0,0.0D0,1.0D0;           0000035P    55
126,1,1,1,0,1,0,-7.071067811865475D0,-7.071067811865475D0,       0000037P    56
-0.0D0,-0.0D0,1.0D0,1.0D0,0.0D0,5.0D0,0.0D0,5.0D0,0.0D0,0.0D0,   0000037P    57
-7.071067811865475D0,-0.0D0,0.0D0,0.0D0,1.0D0;                   0000037P    58
126,3,3,1,0,1,0,-7.071067811865475D0,-7.071067811865475D0,       0000039P    59
-7.071067811865475D0,-7.071067811865475D0,-0.0D0,-0.0D0,-0.0D0,  0000039P    60
-0.0D0,1.0D0,1.0D0,1.0D0,5.0D0,0.0D0,0.0D0,                      0000039P    61
3.333333333333334D0,1.666666666666666D0,0.0D0,                   0000039P    62
1.666666666666666D0,3.333333333333334D0,0.0D0,0.0D0,5.0D0,0.0D0, 0000039P    63
-7.071067811865475D0,-0.0D0,0.0D0,0.0D0,1.0D0;                   0000039P    64
141,1,3,27,3,29,1,1,31,33,1,1,35,37,1,1,39;                      0000041P    65
143,1,27,1,41;                                                   0000043P    66
S0000001G0000012D0000044P0000066                                        T      1
```

File 3: STEP trimming example – surface model

```
ISO-10303-21;
HEADER;
/* Generated by software containing ST-Developer
 * from STEP Tools, Inc. (www.steptools.com)
 */
/* OPTION: using custom schema-name function */

FILE_DESCRIPTION(
/* description */ (''),
/* implementation_level */ '2;1');

FILE_NAME(
/* name */ 'Trim',
/* time_stamp */ '2016-09-06T17:08:41+02:00',
/* author */ (''),
/* organization */ (''),
/* preprocessor_version */ 'ST-DEVELOPER␣v15',
/* originating_system */ '',
/* authorisation */ '');

FILE_SCHEMA (('AUTOMOTIVE_DESIGN'));
ENDSEC;

DATA;
#10=SHAPE_REPRESENTATION_RELATIONSHIP('','',#100,#15);
#11=PRESENTATION_LAYER_ASSIGNMENT('Default','',(#13));
#12=PRESENTATION_LAYER_ASSIGNMENT('Default','',(#14));
#13=SHELL_BASED_SURFACE_MODEL('shell_1',(#16));
#14=SHELL_BASED_SURFACE_MODEL('shell_2',(#17));
#15=MANIFOLD_SURFACE_SHAPE_REPRESENTATION('shell_rep_0',(#13,#14,#102),
#99);
#16=OPEN_SHELL('',(#18));
#17=OPEN_SHELL('',(#19));
#18=ADVANCED_FACE('',(#20),#80,.T.);
#19=ADVANCED_FACE('',(#21),#81,.T.);
#20=FACE_OUTER_BOUND('',#22,.T.);
#21=FACE_OUTER_BOUND('',#23,.T.);
#22=EDGE_LOOP('',(#24,#25,#26,#27));
#23=EDGE_LOOP('',(#28,#29,#30));
#24=ORIENTED_EDGE('',*,*,#52,.T.);
#25=ORIENTED_EDGE('',*,*,#53,.T.);
#26=ORIENTED_EDGE('',*,*,#54,.T.);
#27=ORIENTED_EDGE('',*,*,#55,.T.);
#28=ORIENTED_EDGE('',*,*,#56,.T.);
#29=ORIENTED_EDGE('',*,*,#57,.T.);
#30=ORIENTED_EDGE('',*,*,#58,.T.);
#31=PCURVE('',#80,#38);
#32=PCURVE('',#80,#39);
#33=PCURVE('',#80,#40);
#34=PCURVE('',#80,#41);
#35=PCURVE('',#81,#42);
#36=PCURVE('',#81,#43);
#37=PCURVE('',#81,#44);
#38=DEFINITIONAL_REPRESENTATION('',(#60),#152);
#39=DEFINITIONAL_REPRESENTATION('',(#62),#152);
#40=DEFINITIONAL_REPRESENTATION('',(#64),#152);
#41=DEFINITIONAL_REPRESENTATION('',(#66),#152);
#42=DEFINITIONAL_REPRESENTATION('',(#68),#152);
#43=DEFINITIONAL_REPRESENTATION('',(#70),#152);
#44=DEFINITIONAL_REPRESENTATION('',(#72),#152);
#45=SURFACE_CURVE('',#59,(#31),.PCURVE_S1.);
#46=SURFACE_CURVE('',#61,(#32),.PCURVE_S1.);
#47=SURFACE_CURVE('',#63,(#33),.PCURVE_S1.);
#48=SURFACE_CURVE('',#65,(#34),.PCURVE_S1.);
#49=SURFACE_CURVE('',#67,(#35),.PCURVE_S1.);
#50=SURFACE_CURVE('',#69,(#36),.PCURVE_S1.);
#51=SURFACE_CURVE('',#71,(#37),.PCURVE_S1.);
#52=EDGE_CURVE('',#75,#76,#45,.T.);
```

```
#53=EDGE_CURVE('',#76,#73,#46,.T.);
#54=EDGE_CURVE('',#73,#74,#47,.T.);
#55=EDGE_CURVE('',#74,#75,#48,.T.);
#56=EDGE_CURVE('',#78,#79,#49,.T.);
#57=EDGE_CURVE('',#79,#77,#50,.T.);
#58=EDGE_CURVE('',#77,#78,#51,.T.);
#59=B_SPLINE_CURVE_WITH_KNOTS('',1,(#116,#117),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,7.07106781186547),.UNSPECIFIED.);
#60=B_SPLINE_CURVE_WITH_KNOTS('',1,(#118,#119),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,7.07106781186547),.UNSPECIFIED.);
#61=B_SPLINE_CURVE_WITH_KNOTS('',1,(#120,#121),.UNSPECIFIED.,.F.,.F.,(2,
2),(5.,10.),.UNSPECIFIED.);
#62=B_SPLINE_CURVE_WITH_KNOTS('',1,(#122,#123),.UNSPECIFIED.,.F.,.F.,(2,
2),(5.,10.),.UNSPECIFIED.);
#63=B_SPLINE_CURVE_WITH_KNOTS('',1,(#124,#125),.UNSPECIFIED.,.F.,.F.,(2,
2),(-7.07106781186547,0.),.UNSPECIFIED.);
#64=B_SPLINE_CURVE_WITH_KNOTS('',1,(#126,#127),.UNSPECIFIED.,.F.,.F.,(2,
2),(-7.07106781186547,0.),.UNSPECIFIED.);
#65=B_SPLINE_CURVE_WITH_KNOTS('',1,(#128,#129),.UNSPECIFIED.,.F.,.F.,(2,
2),(-10.,-5.),.UNSPECIFIED.);
#66=B_SPLINE_CURVE_WITH_KNOTS('',1,(#130,#131),.UNSPECIFIED.,.F.,.F.,(2,
2),(-10.,-5.),.UNSPECIFIED.);
#67=B_SPLINE_CURVE_WITH_KNOTS('',1,(#139,#140),.UNSPECIFIED.,.F.,.F.,(2,
2),(-7.07106781186547,0.),.UNSPECIFIED.);
#68=B_SPLINE_CURVE_WITH_KNOTS('',1,(#141,#142),.UNSPECIFIED.,.F.,.F.,(2,
2),(-7.07106781186547,0.),.UNSPECIFIED.);
#69=B_SPLINE_CURVE_WITH_KNOTS('',1,(#143,#144),.UNSPECIFIED.,.F.,.F.,(2,
2),(-5.,0.),.UNSPECIFIED.);
#70=B_SPLINE_CURVE_WITH_KNOTS('',1,(#145,#146),.UNSPECIFIED.,.F.,.F.,(2,
2),(-5.,0.),.UNSPECIFIED.);
#71=B_SPLINE_CURVE_WITH_KNOTS('',1,(#147,#148),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,5.),.UNSPECIFIED.);
#72=B_SPLINE_CURVE_WITH_KNOTS('',1,(#149,#150),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,5.),.UNSPECIFIED.);
#73=VERTEX_POINT('',#112);
#74=VERTEX_POINT('',#113);
#75=VERTEX_POINT('',#114);
#76=VERTEX_POINT('',#115);
#77=VERTEX_POINT('',#136);
#78=VERTEX_POINT('',#137);
#79=VERTEX_POINT('',#138);
#80=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#108,#109),(#110,#111)),
 .UNSPECIFIED.,.F.,.F.,.F.,(2,2),(2,2),(0.,7.07106781186547),(0.,10.),
 .UNSPECIFIED.);
#81=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#132,#133),(#134,#135)),
 .UNSPECIFIED.,.F.,.F.,.F.,(2,2),(2,2),(0.,5.),(0.,5.),.UNSPECIFIED.);
#82=SHAPE_DEFINITION_REPRESENTATION(#83,#100);
#83=PRODUCT_DEFINITION_SHAPE('Document','',#85);
#84=PRODUCT_DEFINITION_CONTEXT('3D␣Mechanical␣Parts',#89,'design');
#85=PRODUCT_DEFINITION('A','First␣version',#86,#84);
#86=PRODUCT_DEFINITION_FORMATION_WITH_SPECIFIED_SOURCE('A',
'First␣version',#91,.MADE.);
#87=PRODUCT_RELATED_PRODUCT_CATEGORY('tool','tool',(#91));
#88=APPLICATION_PROTOCOL_DEFINITION('Draft␣International␣Standard',
'automotive_design',1999,#89);
#89=APPLICATION_CONTEXT(
'data␣for␣automotive␣mechanical␣design␣processes');
#90=PRODUCT_CONTEXT('3D␣Mechanical␣Parts',#89,'mechanical');
#91=PRODUCT('Document','Document','Rhino␣converted␣to␣STEP',(#90));
#92=(
LENGTH_UNIT()
NAMED_UNIT(*)
SI_UNIT(.MILLI.,.METRE.)
);
#93=(
NAMED_UNIT(*)
PLANE_ANGLE_UNIT()
SI_UNIT($,.RADIAN.)
);
#94=DIMENSIONAL_EXPONENTS(0.,0.,0.,0.,0.,0.,0.);
#95=PLANE_ANGLE_MEASURE_WITH_UNIT(PLANE_ANGLE_MEASURE(0.01745329252),#93);
#96=(
CONVERSION_BASED_UNIT('DEGREES',#95)
NAMED_UNIT(#94)
PLANE_ANGLE_UNIT()
);
#97=(
NAMED_UNIT(*)
SI_UNIT($,.STERADIAN.)
SOLID_ANGLE_UNIT()
);
#98=UNCERTAINTY_MEASURE_WITH_UNIT(LENGTH_MEASURE(0.001),#92,
'DISTANCE_ACCURACY_VALUE',
'Maximum␣model␣space␣distance␣between␣geometric␣entities␣at␣asserted␣c
onnectivities');
#99=(
GEOMETRIC_REPRESENTATION_CONTEXT(3)
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT((#98))
GLOBAL_UNIT_ASSIGNED_CONTEXT((#97,#96,#92))
REPRESENTATION_CONTEXT('ID1','3D')
);
#100=SHAPE_REPRESENTATION('Document',(#101,#102),#99);
#101=AXIS2_PLACEMENT_3D('',#107,#103,#104);
#102=AXIS2_PLACEMENT_3D('',#151,#105,#106);
#103=DIRECTION('',(0.,0.,1.));
#104=DIRECTION('',(1.,0.,0.));
#105=DIRECTION('',(0.,0.,1.));
#106=DIRECTION('',(1.,0.,0.));
#107=CARTESIAN_POINT('',(0.,0.,0.));
#108=CARTESIAN_POINT('',(5.,0.,-5.));
#109=CARTESIAN_POINT('',(5.,0.,5.));
#110=CARTESIAN_POINT('',(8.88178419700125E-16,5.,-5.));
#111=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#112=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#113=CARTESIAN_POINT('',(5.,0.,5.));
#114=CARTESIAN_POINT('',(5.,0.,8.88178419700125E-16));
#115=CARTESIAN_POINT('',(8.88178419700125E-16,5.,8.88178419700125E-16));
#116=CARTESIAN_POINT('',(5.,0.,8.88178419700125E-16));
#117=CARTESIAN_POINT('',(8.88178419700125E-16,5.,8.88178419700125E-16));
#118=CARTESIAN_POINT('',(0.,5.));
#119=CARTESIAN_POINT('',(7.07106781186547,5.));
#120=CARTESIAN_POINT('',(8.88178419700125E-16,5.,8.88178419700125E-16));
#121=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#122=CARTESIAN_POINT('',(7.07106781186547,5.));
#123=CARTESIAN_POINT('',(7.07106781186547,10.));
#124=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#125=CARTESIAN_POINT('',(5.,0.,5.));
#126=CARTESIAN_POINT('',(7.07106781186547,10.));
#127=CARTESIAN_POINT('',(0.,10.));
#128=CARTESIAN_POINT('',(5.,0.,5.));
#129=CARTESIAN_POINT('',(5.,0.,8.88178419700125E-16));
#130=CARTESIAN_POINT('',(0.,10.));
#131=CARTESIAN_POINT('',(0.,5.));
#132=CARTESIAN_POINT('',(0.,0.,0.));
#133=CARTESIAN_POINT('',(5.,0.,0.));
#134=CARTESIAN_POINT('',(0.,5.,0.));
#135=CARTESIAN_POINT('',(5.,5.,0.));
#136=CARTESIAN_POINT('',(0.,0.,0.));
#137=CARTESIAN_POINT('',(0.,5.,0.));
#138=CARTESIAN_POINT('',(5.,0.,0.));
#139=CARTESIAN_POINT('',(0.,5.,0.));
#140=CARTESIAN_POINT('',(5.,0.,0.));
#141=CARTESIAN_POINT('',(5.,0.));
#142=CARTESIAN_POINT('',(0.,5.));
#143=CARTESIAN_POINT('',(5.,0.,0.));
#144=CARTESIAN_POINT('',(0.,0.,0.));
#145=CARTESIAN_POINT('',(0.,5.));
#146=CARTESIAN_POINT('',(0.,0.));
#147=CARTESIAN_POINT('',(0.,0.,0.));
#148=CARTESIAN_POINT('',(0.,5.,0.));
#149=CARTESIAN_POINT('',(0.,0.));
#150=CARTESIAN_POINT('',(5.,0.));
#151=CARTESIAN_POINT('',(0.,0.,0.));
#152=(
GEOMETRIC_REPRESENTATION_CONTEXT(2)
PARAMETRIC_REPRESENTATION_CONTEXT()
REPRESENTATION_CONTEXT('pspace','')
);
ENDSEC;
END-ISO-10303-21;
```

File 4: STEP trimming example – solid model

```
ISO-10303-21;
HEADER;
/* Generated by software containing ST-Developer
 * from STEP Tools, Inc. (www.steptools.com)
 */
/* OPTION: using custom schema-name function */

FILE_DESCRIPTION(
/* description */ (''),
/* implementation_level */ '2;1');

FILE_NAME(
/* name */ 'Boolean',
/* time_stamp */ '2016-09-06T17:07:00+02:00',
/* author */ (''),
/* organization */ (''),
/* preprocessor_version */ 'ST-DEVELOPER␣v15',
/* originating_system */ '',
/* authorisation */ '');

FILE_SCHEMA ((''AUTOMOTIVE_DESIGN''));
ENDSEC;

DATA;
#10=SHAPE_REPRESENTATION_RELATIONSHIP('','',#92,#13);
#11=PRESENTATION_LAYER_ASSIGNMENT('Default','',(#12));
#12=SHELL_BASED_SURFACE_MODEL('shell_1',(#14));
#13=MANIFOLD_SURFACE_SHAPE_REPRESENTATION('shell_rep_0',(#12,#94),#91);
#14=OPEN_SHELL('',(#15,#16));
#15=ADVANCED_FACE('',(#17),#72,.T.);
#16=ADVANCED_FACE('',(#18),#73,.T.);
#17=FACE_OUTER_BOUND('',#19,.T.);
#18=FACE_OUTER_BOUND('',#20,.T.);
#19=EDGE_LOOP('',(#21,#22,#23,#24));
#20=EDGE_LOOP('',(#25,#26,#27));
#21=ORIENTED_EDGE('',*,*,#48,.T.);
#22=ORIENTED_EDGE('',*,*,#49,.T.);
#23=ORIENTED_EDGE('',*,*,#50,.T.);
#24=ORIENTED_EDGE('',*,*,#51,.T.);
#25=ORIENTED_EDGE('',*,*,#52,.T.);
#26=ORIENTED_EDGE('',*,*,#53,.T.);
#27=ORIENTED_EDGE('',*,*,#48,.F.);
#28=PCURVE('',#72,#35);
#29=PCURVE('',#72,#36);
#30=PCURVE('',#72,#37);
#31=PCURVE('',#72,#38);
#32=PCURVE('',#73,#39);
#33=PCURVE('',#73,#40);
#34=PCURVE('',#73,#41);
#35=DEFINITIONAL_REPRESENTATION('',(#55),#140);
#36=DEFINITIONAL_REPRESENTATION('',(#57),#140);
#37=DEFINITIONAL_REPRESENTATION('',(#59),#140);
#38=DEFINITIONAL_REPRESENTATION('',(#61),#140);
#39=DEFINITIONAL_REPRESENTATION('',(#63),#140);
#40=DEFINITIONAL_REPRESENTATION('',(#65),#140);
#41=DEFINITIONAL_REPRESENTATION('',(#66),#140);
#42=SURFACE_CURVE('',#54,(#28,#34),.PCURVE_S1.);
#43=SURFACE_CURVE('',#56,(#29),.PCURVE_S1.);
#44=SURFACE_CURVE('',#58,(#30),.PCURVE_S1.);
#45=SURFACE_CURVE('',#60,(#31),.PCURVE_S1.);
#46=SURFACE_CURVE('',#62,(#32),.PCURVE_S1.);
#47=SURFACE_CURVE('',#64,(#33),.PCURVE_S1.);
#48=EDGE_CURVE('',#69,#68,#42,.T.);
#49=EDGE_CURVE('',#68,#70,#43,.T.);
```

```
#50=EDGE_CURVE('',#70,#71,#44,.T.);
#51=EDGE_CURVE('',#71,#69,#45,.T.);
#52=EDGE_CURVE('',#69,#67,#46,.T.);
#53=EDGE_CURVE('',#67,#68,#47,.T.);
#54=B_SPLINE_CURVE_WITH_KNOTS('',1,(#113,#114),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,7.07106781186547),.UNSPECIFIED.);
#55=B_SPLINE_CURVE_WITH_KNOTS('',1,(#115,#116),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,7.07106781186547),.UNSPECIFIED.);
#56=B_SPLINE_CURVE_WITH_KNOTS('',1,(#117,#118),.UNSPECIFIED.,.F.,.F.,(2,
2),(5.,10.),.UNSPECIFIED.);
#57=B_SPLINE_CURVE_WITH_KNOTS('',1,(#119,#120),.UNSPECIFIED.,.F.,.F.,(2,
2),(5.,10.),.UNSPECIFIED.);
#58=B_SPLINE_CURVE_WITH_KNOTS('',1,(#121,#122),.UNSPECIFIED.,.F.,.F.,(2,
2),(-7.07106781186547,0.),.UNSPECIFIED.);
#59=B_SPLINE_CURVE_WITH_KNOTS('',1,(#123,#124),.UNSPECIFIED.,.F.,.F.,(2,
2),(-7.07106781186547,0.),.UNSPECIFIED.);
#60=B_SPLINE_CURVE_WITH_KNOTS('',1,(#125,#126),.UNSPECIFIED.,.F.,.F.,(2,
2),(-10.,-5.),.UNSPECIFIED.);
#61=B_SPLINE_CURVE_WITH_KNOTS('',1,(#127,#128),.UNSPECIFIED.,.F.,.F.,(2,
2),(-10.,-5.),.UNSPECIFIED.);
#62=B_SPLINE_CURVE_WITH_KNOTS('',1,(#129,#130),.UNSPECIFIED.,.F.,.F.,(2,
2),(-5.,0.),.UNSPECIFIED.);
#63=B_SPLINE_CURVE_WITH_KNOTS('',1,(#131,#132),.UNSPECIFIED.,.F.,.F.,(2,
2),(-5.,0.),.UNSPECIFIED.);
#64=B_SPLINE_CURVE_WITH_KNOTS('',1,(#133,#134),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,5.),.UNSPECIFIED.);
#65=B_SPLINE_CURVE_WITH_KNOTS('',1,(#135,#136),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,5.),.UNSPECIFIED.);
#66=B_SPLINE_CURVE_WITH_KNOTS('',1,(#137,#138),.UNSPECIFIED.,.F.,.F.,(2,
2),(0.,7.07106781186547),.UNSPECIFIED.);
#67=VERTEX_POINT('',#108);
#68=VERTEX_POINT('',#109);
#69=VERTEX_POINT('',#110);
#70=VERTEX_POINT('',#111);
#71=VERTEX_POINT('',#112);
#72=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#100,#101),(#102,#103)),
 .UNSPECIFIED.,.F.,.F.,.F.,(2,2),(2,2),(0.,7.07106781186547),(0.,10.),
 .UNSPECIFIED.);
#73=B_SPLINE_SURFACE_WITH_KNOTS('',1,1,((#104,#105),(#106,#107)),
 .UNSPECIFIED.,.F.,.F.,.F.,(2,2),(2,2),(0.,5.),(0.,5.),.UNSPECIFIED.);
#74=SHAPE_DEFINITION_REPRESENTATION(#75,#92);
#75=PRODUCT_DEFINITION_SHAPE('Document','',#77);
#76=PRODUCT_DEFINITION_CONTEXT('3D␣Mechanical␣Parts',#81,'design');
#77=PRODUCT_DEFINITION('A','First␣version',#78,#76);
#78=PRODUCT_DEFINITION_FORMATION_WITH_SPECIFIED_SOURCE('A',
'First␣version',#83,.MADE.);
#79=PRODUCT_RELATED_PRODUCT_CATEGORY('tool','tool',(#83));
#80=APPLICATION_PROTOCOL_DEFINITION('Draft␣International␣Standard',
'automotive_design',1999,#81);
#81=APPLICATION_CONTEXT(
'data␣for␣automotive␣mechanical␣design␣processes');
#82=PRODUCT_CONTEXT('3D␣Mechanical␣Parts',#81,'mechanical');
#83=PRODUCT('Document','Document','Rhino␣converted␣to␣STEP',(#82));
#84=(
LENGTH_UNIT()
NAMED_UNIT(*)
SI_UNIT(.MILLI.,.METRE.)
);
#85=(
NAMED_UNIT(*)
PLANE_ANGLE_UNIT()
SI_UNIT($,.RADIAN.)
);
#86=DIMENSIONAL_EXPONENTS(0.,0.,0.,0.,0.,0.,0.);
#87=PLANE_ANGLE_MEASURE_WITH_UNIT(PLANE_ANGLE_MEASURE(0.01745329252),#85);
#88=(
CONVERSION_BASED_UNIT('DEGREES',#87)
NAMED_UNIT(#86)
PLANE_ANGLE_UNIT()
);
#89=(
NAMED_UNIT(*)
SI_UNIT($,.STERADIAN.)
SOLID_ANGLE_UNIT()
);
#90=UNCERTAINTY_MEASURE_WITH_UNIT(LENGTH_MEASURE(0.001),#84,
'DISTANCE_ACCURACY_VALUE',
'Maximum␣model␣space␣distance␣between␣geometric␣entities␣at␣asserted␣c
onnectivities');
#91=(
GEOMETRIC_REPRESENTATION_CONTEXT(3)
GLOBAL_UNCERTAINTY_ASSIGNED_CONTEXT((#90))
GLOBAL_UNIT_ASSIGNED_CONTEXT((#89,#88,#84))
REPRESENTATION_CONTEXT('ID1','3D')
);
#92=SHAPE_REPRESENTATION('Document',(#93,#94),#91);
#93=AXIS2_PLACEMENT_3D('',#99,#95,#96);
#94=AXIS2_PLACEMENT_3D('',#139,#97,#98);
#95=DIRECTION('',(0.,0.,1.));
#96=DIRECTION('',(1.,0.,0.));
#97=DIRECTION('',(0.,0.,1.));
#98=DIRECTION('',(1.,0.,0.));
#99=CARTESIAN_POINT('',(0.,0.,0.));
#100=CARTESIAN_POINT('',(5.,0.,-5.));
#101=CARTESIAN_POINT('',(5.,0.,5.));
#102=CARTESIAN_POINT('',(8.88178419700125E-16,5.,-5.));
#103=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#104=CARTESIAN_POINT('',(0.,0.,0.));
#105=CARTESIAN_POINT('',(5.,0.,0.));
#106=CARTESIAN_POINT('',(0.,5.,0.));
#107=CARTESIAN_POINT('',(5.,5.,0.));
#108=CARTESIAN_POINT('',(0.,0.,0.));
#109=CARTESIAN_POINT('',(0.,5.,0.));
#110=CARTESIAN_POINT('',(5.,0.,0.));
#111=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#112=CARTESIAN_POINT('',(5.,0.,5.));
#113=CARTESIAN_POINT('',(5.,0.,0.));
#114=CARTESIAN_POINT('',(0.,5.,0.));
#115=CARTESIAN_POINT('',(0.,5.));
#116=CARTESIAN_POINT('',(7.07106781186547,5.));
#117=CARTESIAN_POINT('',(0.,5.,0.));
#118=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#119=CARTESIAN_POINT('',(7.07106781186547,5.));
#120=CARTESIAN_POINT('',(7.07106781186547,10.));
#121=CARTESIAN_POINT('',(8.88178419700125E-16,5.,5.));
#122=CARTESIAN_POINT('',(5.,0.,5.));
#123=CARTESIAN_POINT('',(7.07106781186547,10.));
#124=CARTESIAN_POINT('',(0.,10.));
#125=CARTESIAN_POINT('',(5.,0.,5.));
#126=CARTESIAN_POINT('',(5.,0.,0.));
#127=CARTESIAN_POINT('',(0.,10.));
#128=CARTESIAN_POINT('',(0.,5.));
#129=CARTESIAN_POINT('',(5.,0.,0.));
#130=CARTESIAN_POINT('',(0.,0.,0.));
#131=CARTESIAN_POINT('',(0.,5.));
#132=CARTESIAN_POINT('',(0.,0.));
#133=CARTESIAN_POINT('',(0.,0.,0.));
#134=CARTESIAN_POINT('',(0.,5.,0.));
#135=CARTESIAN_POINT('',(0.,0.));
#136=CARTESIAN_POINT('',(5.,0.));
#137=CARTESIAN_POINT('',(0.,5.));
#138=CARTESIAN_POINT('',(5.,0.));
#139=CARTESIAN_POINT('',(0.,0.,0.));
#140=(
GEOMETRIC_REPRESENTATION_CONTEXT(2)
PARAMETRIC_REPRESENTATION_CONTEXT()
REPRESENTATION_CONTEXT('pspace','')
);
ENDSEC;
END-ISO-10303-21;
```

References

1. Abi-Ezzi SS, Shirman LA (1991) Tessellation of curved surfaces under highly varying transformations. In: Proceedings of EUROGRAPHICS, vol 91, p 385–397
2. Abi-Ezzi SS, Subramaniam S (1994) Fast dynamic tessellation of trimmed NURBS surfaced. In: Computer Graphics Forum, vol 13, p 107–126
3. Abrams SL, Cho W, Hu CY, Maekawa T, Patrikalakis NM, Sherbrooke EC, Ye X (1998) Efficient and reliable methods for rounded-interval arithmetic. Comput Aided Des 30(8):657–665
4. Agoston MK (2005) Computer graphics and geometric modeling. Springer, London
5. Alavala CR (2013) CAD/CAM: concepts and applications. PHI, New Delhi
6. Aomura S, Uehara T (1990) Self-intersection of an offset surface. Comput Aided Des 22(7):417–421
7. Apostolatos A, Schmidt R, Wüchner R, Bletzinger KU (2014) A Nitsche-type formulation and comparison of the most common domain decomposition methods in isogeometric analysis. Int J Numer Methods Eng 97(7):473–504
8. Appel A (1968) Some techniques for shading machine renderings of solids. In: Proceedings of the spring joint computer conference. ACM, New York, p 37–45
9. Applegarth I, Catley D, Bradley I (1989) Clipping of B-spline patches at surface curves. In: Handscomb DC (ed) The mathematics of surfaces III. Clarendon Press, Oxford, pp 229–242
10. Aurenhammer F (1991) Voronoi diagrams—a survey of a fundamental geometric data structure. ACM Comput Surv 23(3):345–405
11. Auricchio F, Beirão Da Veiga L, Hughes TJR, Reali A, Sangalli G (2010) Isogeometric collocation methods. Math Models Methods Appl Sci 20(11):2075–2107
12. Babuška I (1973) The finite element method with Lagrangian multipliers. Numer Math 20(3):179–192
13. Babuška I (1973) Finite-element method with penalty. Math Comput 27(122):221–228
14. Bajaj CL, Hoffmann CM, Lynch RE, Hopcroft JEH (1988) Tracing surface intersections. Comput Aided Geom Des 5(4):285–307
15. Ballard DH (1981) Strip trees: a hierarchical representation for curves. Commun ACM 24(5):310–321
16. Barber CB, Dobkin DP, Huhdanpaa H (1996) The quickhull algorithm for convex hulls. ACM Trans Math Softw 22(4):469–483
17. Barbosa HJC, Hughes TJR (1991) The finite element method with Lagrange multipliers on the boundary: circumventing the Babuška–Brezzi condition. Comput Methods Appl Mech Eng 85(1):109–128
18. Barbosa HJC, Hughes TJR (1992) Boundary Lagrange multipliers in finite element methods: error analysis in natural norms. Numer Math 62:1–15
19. Barnhill RE, Farin G, Jordan M, Piper BR (1987) Surface/surface intersection. Comput Aided Geom Des 4(1):3–16
20. Barnhill RE, Kersey S (1990) A marching method for parametric surface/surface intersection. Comput Aided Geom Des 7(1–4):257–280
21. Baumgart BG (1972) Winged edge polyhedron representation. Technical report, DTIC Document
22. Bazilevs Y, Calo VM, Cottrell JA, Evans JA, Hughes TJR, Lipton S, Scott MA, Sederberg TW (2010) Isogeometric analysis using T-splines. Comput Methods Appl Mech Eng 199(5–8):229–263
23. Bazilevs Y, Hughes TJR (2007) Weak imposition of Dirichlet boundary conditions in fluid mechanics. Comput Fluids 36(1):12–26
24. Bazilevs Y, Michler C, Calo VM, Hughes TJR (2007) Weak Dirichlet boundary conditions for wall-bounded turbulent flows. Comput Methods Appl Mech Eng 196(49–52):4853–4862
25. Beer G, Marussig B, Zechner J (2015) A simple approach to the numerical simulation with trimmed CAD surfaces. Comput Methods Appl Mech Eng 285:776–790
26. Benouamer MO, Michelucci D, Peroche B (1994) Error-free boundary evaluation based on a lazy rational arithmetic—a detailed implementation. Comput Aided Des 26(6):403–416
27. Benthin C, Wald I, Slusallek P (2004) Interactive ray tracing of free-form surfaces. In: Proceedings of the 3rd international conference on computer graphics, virtual reality, visualisation and interaction in Africa. ACM, p 99–106
28. Bern M, Eppstein D, Gilbert J (1994) Provably good mesh generation. J Comput Syst Sci 48(3):384–409
29. Bézier P (1974) Mathematical and practical possibilities of UNISURF. In: Barnhill RE, Riesenfeld RF (eds) Computer aided geometric design. Academic, New York, pp 127–152
30. Biermann H, Kristjansson D, Zorin D (2001) Approximate Boolean operations on free-form solids. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques, SIGGRAPH '01. ACM, p 185–194
31. Biswas A, Fenves SJ, Shapiro V, Sriram R (2008) Representation of heterogeneous material properties in the Core Product Model. Eng Comput 24(1):43–58
32. Boehm W (1980) Inserting new knots into B-spline curves. Comput Aided Des 12(4):199–201
33. de Boor C (1972) On calculating with B-splines. J Approx Theory 6(1):50–62
34. de Boor C (2001) A practical guide to splines. In: Applied mathematical sciences, vol 27. Springer, New York
35. de Boor C, Fix GJ (1973) Spline approximation by quasiinterpolants. J Approx Theory 8(1):19–45
36. Braid IC (1974) Designing with volumes, 2nd edn. Cantab Press, Cambridge University, Cambridge
37. Breitenberger M (2016) CAD-integrated design and analysis of shell structures. PhD Thesis, Technische Universität München
38. Breitenberger M, Apostolatos A, Philipp B, Wüchner R, Bletzinger KU (2015) Analysis in computer aided design: nonlinear isogeometric B-Rep analysis of shell structures. Comput Methods Appl Mech Eng 284:401–457
39. Brivadis E, Buffa A, Wohlmuth B, Wunderlich L (2015) Isogeometric mortar methods. Comput Methods Appl Mech Eng 284:292–319
40. Brown CM (1982) PADL-2: a technical summary. IEEE Comput Graph Appl 2(2):69–84
41. Brunnett G (1995) Geometric design with trimmed surfaces. In: Hagen H, Farin G, Noltemeierm H (eds) Geometric modelling: Dagstuhl 1993, computing Supplement 10. Springer, Berlin, pp 101–115
42. Burman E, Hansbo P (2012) Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. Appl Numer Math 62(4):328–341
43. C3D Labs. C3D kernel documentation. http://c3d.ascon.net/doc/math/class_mb_surface_intersection_curve.html#details. Accessed 19 Aug 2016
44. Campagna S, Slusallek P, Seidel HP (1997) Ray tracing of spline surfaces: Bézier clipping, Chebyshev boxing, and bounding volume hierarchy—a critical comparison with new results. Vis Comput 13(6):265–282

45. Campbell RJ, Flynn PJ (2001) A survey of free-form object representation and recognition techniques. Comput Vis Image Underst 81(2):166–210
46. Carlson WE (1982) An algorithm and data structure for 3D object synthesis using surface patch intersections. SIGGRAPH Comput Graph 16(3):255–263
47. Casale MS (1987) Free-form solid modeling with trimmed surface patches. IEEE Comput Graph Appl 7(1):33–43
48. Casale MS, Bobrow JE (1989) The analysis of solids without mesh generation using trimmed patch boundary elements. Eng Comput 5(3–4):249–257
49. Casale MS, Bobrow JE (1989) A set operation algorithm for sculptured solids modeled with trimmed patches. Comput Aided Geom Des 6(3):235–247
50. Casale MS, Bobrow JE, Underwood R (1992) Trimmed-patch boundary elements: bridging the gap between solid modeling and engineering analysis. Comput Aided Des 24(4):193–199
51. Cashman TJ, Augsdörfer UH, Dodgson NA, Sabin MA (2009) NURBS with extraordinary points: high-degree, non-uniform, rational subdivision schemes. ACM Trans Graph 28(3):46:1–46:9
52. Catmull E (1974) A subdivision algorithm for computer display of curved surfaces. Technical report. Computer Science Department, University of Utah
53. Catmull E, Clark J (1978) Recursively generated B-spline surfaces on arbitrary topological meshes. Comput Aided Des 10(6):350–355
54. Chaikin GM (1974) An algorithm for high-speed curve generation. Comput Graph Image Process 3(4):346–349
55. Chew LP (1993) Guaranteed-quality mesh generation for curved surfaces. In: Proceedings of the ninth annual symposium on computational geometry. ACM, New York, p 274–280
56. Chiyokura H, Kimura F (1983) Design of solids with free-form surfaces. SIGGRAPH Comput Graph 17(3):289–298
57. Cho W, Maekawa T, Patrikalakis NM, Peraire J (1999) Topologically reliable approximation of trimmed polynomial surface patches. Graph Models Image Process 61(2):84–109
58. Cho W, Patrikalakis NM, Peraire J (1998) Approximate development of trimmed patches for surface tessellation. Comput Aided Des 30(14):1077–1087
59. Cirak F, Long Q (2011) Subdivision shells with exact boundary control and non-manifold geometry. Int J Numer Methods Eng 88(9):897–923
60. Cohen E, Lyche T, Riesenfeld R (1980) Discrete B-splines and subdivision techniques in computer-aided geometric design and computer graphics. Comput Graph Image Process 14(2):87–111
61. Cohen E, Martin T, Kirby RM, Lyche T, Riesenfeld RF (2010) Analysis-aware modeling: understanding quality considerations in modeling for isogeometric analysis. Comput Methods Appl Mech Eng 199(5–8):334–356
62. Cohen E, Riesenfeld RF, Elber G (2001) Geometric modeling with splines: an introduction. A K Peters, Natick
63. Collin A, Sangalli G, Takacs T (2016) Analysis-suitable G^1 multi-patch parametrizations for C^1 isogeometric spaces. Comput Aided Geom Des 47:93–113 (SI: New Developments Geometry)
64. Cook RL, Porter T, Carpenter L (1984) Distributed ray tracing. SIGGRAPH Comput Graph 18(3):137–145
65. Corney J, Lim T (2001) 3D modeling with ACIS. Saxe-Coburg, Stirling
66. Cottrell JA, Hughes TJR, Bazilevs Y (2009) Isogeometric analysis: toward integration of CAD and FEA. Wiley, Chichester
67. Cottrell JA, Hughes TJR, Reali A (2007) Studies of refinement and continuity in isogeometric structural analysis. Comput Methods Appl Mech Eng 196(41–44):4160–4183
68. Cottrell JA, Reali A, Bazilevs Y, Hughes TJR (2006) Isogeometric analysis of structural vibrations. Comput Methods Appl Mech Eng 195(41–43):5257–5296
69. Dehaemer MJ Jr, Zyda MJ (1991) Simplification of objects rendered by polygonal approximations. Comput Graph 15(2):175–184
70. DeRose T, Kass M, Truong T (1998) Subdivision surfaces in character animation. In: Proceedings of the 25th annual conference on computer graphics and interactive techniques. ACM, p 85–94
71. DeRose TD, Goldman RN, Hagen H, Mann S (1993) Functional composition algorithms via blossoming. ACM Trans Graph 12(2):113–135
72. Dijkstra EW (1959) A note on two problems m connexion with graphs. Numer Math 1:269–271
73. Dobkin DP, Laszlo MJ (1987) Primitives for the manipulation of three-dimensional subdivisions. In: Proceedings of the third annual symposium on computational geometry. ACM, New York, p 86–99
74. Dolenc A, Mäkelä I (1994) Optimized triangulation of parametric surfaces. In: Bowyer A (ed) Computer-aided surface geometry and design: the mathematics of surfaces IV, vol 48. Oxford University Press, Institute of Mathematics and Its Applications, p 169–183
75. Doo D, Sabin M (1978) Behaviour of recursive division surfaces near extraordinary points. Comput Aided Des 10(6):356–360
76. Duff T (1992) Interval arithmetic recursive subdivision for implicit functions and constructive solid geometry. SIGGRAPH Comput Graph 26(2):131–138
77. Edelsbrunner H, Mücke EP (1990) Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms. ACM Trans Graph 9(1):66–104
78. Efremov A, Havran V, Seidel HP (2005) Robust and numerically stable Bézier clipping method for ray tracing NURBS surfaces. In: Proceedings of the 21st spring conference on computer graphics. ACM, New York, p 127–135
79. Elber G (1996) Error bounded piecewise linear approximation of freeform surfaces. Comput Aided Des 28(1):51–57
80. Embar A, Dolbow J, Harari I (2010) Imposing Dirichlet boundary conditions with Nitsche's method and spline-based finite elements. Int J Numer Methods Eng 83(7):877–898
81. Evans JA, Hughes TJR (2013) Isogeometric divergence-conforming B-splines for the steady Navier–Stokes equations. Math Models Methods Appl Sci 23(8):1421–1478
82. Fang SF, Bruderlin B, Zhu XH (1993) Robustness in solid modeling: a tolerance-based intuitionistic approach. Comput Aided Des 25(9):567–576
83. Farin G (2002) Curves and surfaces for CAGD: a practical guide, 5th edn. Morgan Kaufmann, San Francisco
84. Farouki RT (1986) The characterization of parametric surface sections. Comput Vis Graph Image Process 33(2):209–236
85. Farouki RT (1987) Trimmed-surface algorithms for the evaluation and interrogation of solid boundary representations. IBM J Res Dev 31(3):314–334
86. Farouki RT (1999) Closing the gap between CAD model and downstream application. SIAM News 32(5):303–319
87. Farouki RT, Han CY, Hass J, Sederberg TW (2004) Topologically consistent trimmed surface approximations based on triangular patches. Comput Aided Geom Des 21(5):459–478
88. Farouki RT, Hinds JK (1985) A hierarchy of geometric forms. IEEE Comput Graph Appl 5(5):51–78
89. Filip D, Magedson R, Markot R (1986) Surface algorithms using bounds on derivatives. Comput Aided Geom Des 3(4):295–311
90. Flöry S, Hofer M (2008) Constrained curve fitting on manifolds. Comput Aided Des 40(1):25–34

91. Foley T, Sugerman J (2005) KD-tree acceleration structures for a GPU raytracer. In: Proceedings of the conference on graphics hardware. ACM, New York, p 15–22
92. Forsythe GE (1970) Pitfalls in computation, or why a math book isn't enough. Am Math Mon 77(9):931–956
93. Fortune S (1995) Polyhedral modelling with exact arithmetic. In: Proceedings of the third ACM symposium on solid modeling and applications. ACM, p 225–234
94. Frey PJ, George PL (2010) Mesh generation: application to finite elements, 2nd edn. Wiley, New York
95. Fries TP, Omerović S (2016) Higher-order accurate integration of implicit geometries. Int J Numer Methods Eng 106(5):323–371
96. Gaul L, Kögl M, Wagner M (2003) Boundary element methods for engineers and scientists: an introductory course with advanced topics. Springer, Berlin
97. Geimer M, Abert O (2005) Interactive ray tracing of trimmed bicubic Bézier surfaces without triangulation. In: Proceedings of WSCG, p 71–78
98. George PL, Borouchaki H, Frey PJ, Laug P, Saltel E (2004) Chapter 17: mesh generation and mesh adaptivity: theory and techniques. In: Stein E, de Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics. Fundamentals, vol 1. Wiley, Chichester, pp 502–532
99. Glassner AS (1989) An introduction to ray tracing. Academic, London
100. Goldman RN, Filip DJ (1987) Conversion from Bézier rectangles to Bézier triangles. Comput Aided Des 19(1):25–27
101. Goldstein BLM, Kemmerer SJ, Parks CH (1998) A brief history of early product data exchange standards. NISTIR 6221. US Department of Commerce, Technology Administration, Electronics and Electrical Engineering Laboratory, National Institute of Standards and Technology
102. Golovanov N (2014) Geometric modeling. Academia Publishing House, Praha
103. Gordon WJ, Hall CA (1973) Transfinite element methods—blending-function interpolation over arbitrary curved element domains. Numer Math 21(2):109–129
104. Gossard DC, Zuffante RP, Sakurai H (1988) Representing dimensions, tolerances, and features in MCAE systems. IEEE Comput Graph Appl 8(2):51–59
105. Grandine TA, Klein FW (1997) A new approach to the surface intersection problem. Comput Aided Geom Des 14(2):111–134
106. Guibas L, Stolfi J (1985) Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. ACM Trans Graph 4(2):74–123
107. Guo Y, Ruess M, Schillinger D (2016) A parameter-free variational coupling approach for trimmed isogeometric thin shells. Comput Mech 59(4):693–715
108. Guthe M, Balázs A, Klein R (2005) GPU-based trimming and tessellation of NURBS and T-spline surfaces. ACM Trans Graph 24(3):1016–1023
109. Hales TC (2007) The Jordan curve theorem, formally and informally. Am Math Mon 114(10):882–894
110. Hamann B, Tsai PY (1996) A tessellation algorithm for the representation of trimmed NURBS surfaces with arbitrary trimming curves. Comput Aided Des 28(6–7):461–472
111. Hanna SL, Abel JF, Greenberg DP (1983) Intersection of parametric surfaces by means of look-up tables. IEEE Comput Graph Appl 3(7):39–48
112. Hansbo P (2005) Nitsche's method for interface problems in computational mechanics. GAMM-Mitt 28(2):183–206
113. Harbrecht H, Randrianarivony M (2010) From computer aided design to wavelet BEM. Comput Vis Sci 13(2):69–82
114. Hardwick MF, Clay RL, Boggs PT, Walsh EJ, Larzelere AR, Altshuler A (2005) DART system analysis. Technical report SAND2005-4647. Sandia National Laboratories
115. Hass J, Farouki RT, Han CY, Song X, Sederberg TW (2007) Guaranteed consistency of surface intersections and trimmed surfaces using a coupled topology resolution and domain decomposition scheme. Adv Comput Math 27(1):1–26
116. Havran V (2000) Heuristic ray shooting algorithms. PhD Thesis, Czech Technical University, Prague
117. Hickey T, Ju Q, Van Emden MH (2001) Interval arithmetic: from principles to implementation. J ACM 48(5):1038–1068
118. Hiemstra RR, Calabrò F, Schillinger D, Hughes TJR (2016) Optimal and reduced quadrature rules for tensor product and hierarchically refined splines in isogeometric analysis. Comput Methods Appl Mech Eng. doi:10.1016/j.cma.2016.10.049
119. Hofer M, Pottmann H (2004) Energy-minimizing splines in manifolds. ACM Trans Graph 23(3):284–293
120. Hoffmann CM (1989) Geometric and solid modeling. Morgan Kaufmann, San Mateo
121. Hoffmann CM (1989) The problems of accuracy and robustness in geometric computation. Computer 22(3):31–39
122. Hoffmann CM, Hopcroft JE, Karasick MS (1988) Towards implementing robust geometric computations. In: Proceedings of the symposium on computational geometry. ACM, p 106–117
123. Hoffmann CM, Hopcroft JE, Karasick MS (1989) Robust set operations on polyhedral solids. IEEE Comput Graph Appl 9(6):50–59
124. Höllig K (2003) Finite element methods with B-splines. In: Frontiers in applied mathematics, vol 26. SIAM, Philadelphia
125. Höllig K, Reif U (2003) Nonuniform web-splines. Comput Aided Geom Des 20(5):277–294
126. Höllig K, Reif U, Wipper J (2001) B-spline approximation of Neumann problems. Mathematisches Institut A, University of Stuttgart
127. Höllig K, Reif U, Wipper J (2002) Weighted extended B-spline approximation of Dirichlet problems. SIAM J Numer Anal 39(2):442–462
128. Hong YS, Chang TC (2002) A comprehensive review of tolerancing research. Int J Prod Res 40(11):2425–2459
129. Hoschek J (1987) Approximate conversion of spline curves. Comput Aided Geom Des 4(1–2):59–66
130. Hoschek J, Lasser D (1992) Grundlagen der geometrischen Datenverarbeitung. Vieweg+Teubner. English version "Fundamentals of Computer Aided Geometric Design" translated by Schumaker LL
131. Hoschek J, Schneider FJ (1990) Spline conversion for trimmed rational Bézier- and B-spline surfaces. Comput Aided Des 22(9):580–590
132. Hoschek J, Schneider FJ, Wassum P (1989) Optimal approximate conversion of spline surfaces. Comput Aided Geom Des 6(4):293–306
133. Houghton EG, Emnett RF, Factor JD, Sabharwal CL (1985) Implementation of a divide-and-conquer method for intersection of parametric surfaces. Comput Aided Geom Des 2(1):173–183
134. Hu CY, Maekawa T, Patrikalakis NM, Ye X (1997) Robust interval algorithm for surface intersections. Comput Aided Des 29(9):617–627
135. Hu CY, Patrikalakis NM, Ye X (1996) Robust interval solid modelling part I: representations. Comput Aided Des 28(10):807–817
136. Hu CY, Patrikalakis NM, Ye X (1996) Robust interval solid modelling part II: boundary evaluation. Comput Aided Des 28(10):819–830

137. Hu YP, Sun TC (1997) Moving a B-spline surface to a curve—a trimmed surface matching algorithm. Comput Aided Des 29(6):449–455
138. Huerta A, Belytschko T, Fernández-Méndez S, Rabczuk T (2004) Chapter 10: meshfree methods. In: Stein E, de Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics. Fundamentals, vol 1. Wiley, Chichester, pp 279–309
139. Hughes TJR (2000) The finite element method: linear static and dynamic finite element analysis. Courier Corporation, Chicago
140. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Eng 194(39–41):4135–4195
141. Hughes TJR, Sangalli G (2016) Mathematics of isogeometric analysis: a conspectus. In: Stein E, de Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics. Volume set, 2nd edn, vol 3. Wiley, Chichester
142. Hui KC, Wu YB (2005) Feature-based decomposition of trimmed surface. Comput Aided Des 37(8):859–867
143. Jackson DJ (1995) Boundary representation modelling with local tolerances. In: Proceedings of the symposium on solid modeling and applications. ACM, p 247–254
144. Joy KI, Bhetanabhotla MN (1986) Ray tracing parametric surface patches utilizing numerical techniques and ray coherence. SIGGRAPH Comput Graph 20(4):279–285
145. Jüttler B, Mantzaflaris A, Perl R, Rumpf M (2016) On numerical integration in isogeometric subdivision methods for PDEs on surfaces. Comput Methods Appl Mech Eng 302:131–146
146. Kajiya JT (1982) Ray tracing parametric patches. SIGGRAPH Comput Graph 16(3):245–254
147. Kang P, Youn SK (2015) Isogeometric analysis of topologically complex shell structures. Finite Elem Anal Des 99:68–81
148. Kang P, Youn SK (2016) Isogeometric shape optimization of trimmed shell structures. Struct Multidiscip Optim 53(4):825–845
149. Kapl M, Buchegger F, Bercovier M, Jüttler B (2016) Isogeometric analysis with geometrically continuous functions on planar multi-patch geometries. Comput Methods Appl Mech Eng. doi:10.1016/j.cma.2016.06.002
150. Kapl M, Vitrih V, Jüttler B, Birner K (2015) Isogeometric analysis with geometrically continuous functions on two-patch geometries. Comput Math Appl 70(7):1518–1538
151. Kasik DJ, Buxton W, Ferguson DR (2005) Ten CAD challenges. IEEE Comput Graph Appl 25(2):81–92
152. Katz S, Sederberg TW (1988) Genus of the intersection curve of two rational surface patches. Comput Aided Geom Des 5(3):253–258
153. Kaufman A, Cohen D, Yagel R (1993) Volume graphics. Computer 26(7):51–64
154. Kay TL, Kajiya JT (1986) Ray tracing complex scenes. SIGGRAPH Comput Graph 20(4):269–278
155. Kennicott PR, Morea G, Reid E, Parks C, Rinaudot G, Harrod Jr DA, Gruttke WB (1996) Initial graphics exchange specification IGES 5.3. ANS US PRO/IPO-100-1996. US Product Data Association
156. Keyser J, Culver T, Manocha D, Krishnan S (1999) MAPC: a library for efficient and exact manipulation of algebraic points and curves. In: Proceedings of the fifteenth annual symposium on computational geometry. ACM, p 360–369
157. Keyser J, Culver T, Manocha D, Krishnan S (2000) Efficient and exact manipulation of algebraic points and curves. Comput Aided Des 32(11):649–662
158. Keyser J, Krishnan S, Manocha D (1999) Efficient and accurate B-rep generation of low degree sculptured solids using exact arithmetic: I. Representations. Comput Aided Geom Des 16(9):841–859
159. Keyser J, Krishnan S, Manocha D (1999) Efficient and accurate B-rep generation of low degree sculptured solids using exact arithmetic: II. Computation. Comput Aided Geom Des 16(9):861–882
160. Kim HJ, Seo YD, Youn SK (2009) Isogeometric analysis for trimmed CAD surfaces. Comput Methods Appl Mech Eng 198(37–40):2982–2995
161. Kim HJ, Seo YD, Youn SK (2010) Isogeometric analysis with trimming technique for problems of arbitrary complex topology. Comput Methods Appl Mech Eng 199(45–48):2796–2812
162. Kim J, Pratt MJ, Iyer RG, Sriram RD (2008) Standardized data exchange of CAD models with design intent. Comput Aided Des 40(7):760–777
163. Kleiss SK, Pechstein C, Jüttler B, Tomar S (2012) IETI—isogeometric tearing and interconnecting. Comput Methods Appl Mech Eng 247–248:201–215
164. Kollmannsberger S, Özcan A, Baiges J, Ruess M, Rank E, Reali A (2015) Parameter-free, weak imposition of Dirichlet boundary conditions and coupling of trimmed and non-conforming patches. Int J Numer Methods Eng 101(9):670–699
165. Koparkar PA, Mudur SP (1983) A new class of algorithms for the processing of parametric curves. Comput Aided Des 15(1):41–45
166. Korneev VG, Langer U (2004) Chapter 22: domain decomposition methods and preconditioning. In: Stein E, de Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics. Fundamentals, vol 1. Wiley, Chichester, pp 617–647
167. Kosinka J, Cashman TJ (2015) Watertight conversion of trimmed CAD surfaces to Clough–Tocher splines. Comput Aided Geom 37:25–41
168. Kriezis GA, Patrikalakis NM, Wolter FE (1992) Topological and differential-equation methods for surface intersections. Comput Aided Des 24(1):41–55
169. Kriezis GA, Prakash PV, Patrikalakis NM (1990) Method for intersecting algebraic surfaces with rational polynomial patches. Comput Aided Des 22(10):645–654
170. Krishnan S, Manocha D (1997) An efficient surface intersection algorithm based on lower-dimensional formulation. ACM Trans Graph 16(1):74–106
171. Krishnan S, Manocha D, Gopi M, Culver T, Keyser J (2001) BOOLE: a boundary evaluation system for Boolean combinations of sculptured solids. Int J Comput Geom Appl 11(1):105–144
172. Krüger J, Westermann R (2003) Acceleration techniques for GPU-based volume rendering. In: Proceedings of the IEEE visualization. IEEE Computer Society, p 287–292
173. Kudela L (2013) Highly accurate subcell integration in the context of the finite cell method. Master's Thesis, Technical University Munich
174. Kudela L, Zander N, Bog T, Kollmannsberger S, Rank E (2015) Efficient and accurate numerical quadrature for immersed boundary methods. Adv Model Simul Eng Sci 2(1):1–22
175. Kudela L, Zander N, Kollmannsberger S, Rank E (2016) Smart octrees: accurately integrating discontinuous functions in 3D. Comput Methods Appl Mech Eng 306:406–426
176. Kumar S, Manocha D (1995) Efficient rendering of trimmed NURBS surfaces. Comput Aided Des 27(7):509–521
177. LaCourse DE (1995) Handbook of solid modeling. McGraw-Hill, Inc., New York
178. Lane JM, Carpenter L (1979) A generalized scan line algorithm for the computer display of parametrically defined surfaces. Comput Graph Image Process 11(3):290–297
179. Lane JM, Carpenter LC, Whitted T, Blinn JF (1980) Scan line methods for displaying parametrically defined surfaces. Commun ACM 23(1):23–34

180. Lane JM, Riesenfeld RF (1980) A theoretical development for the computer generation and display of piecewise polynomial surfaces. IEEE Trans Pattern Anal Mach Intell PAMI–2(1):35–46
181. Lasser D (1986) Intersection of parametric surfaces in the Bernstein–Bézier representation. Comput Aided Des 18(4):186–192
182. Lasser D (2008) Triangular subpatches of rectangular Bézier surfaces. Comput Math Appl 55(8):1706–1719
183. Lasserre J (1998) Integration on a convex polytope. Proc Am Math Soc 126(8):2433–2441
184. Li K, Qian X (2011) Isogeometric analysis and shape optimization via boundary integral. Comput Aided Des 43(11):1427–1437
185. Lien SL, Shantz M, Pratt V (1987) Adaptive forward differencing for rendering curves and surfaces. SIGGRAPH Comput Graph 21(4):111–118
186. Limaiem A, Trochu F (1995) Geometric algorithms for the intersection of curves and surfaces. Comput Graph 19(3):391–403
187. Lipton S, Evans JA, Bazilevs Y, Elguedj T, Hughes TJR (2010) Robustness of isogeometric structural discretizations under severe mesh distortion. Comput Methods Appl Mech Eng 199(5–8):357–373
188. Litke N, Levin A, Schröder P (2001) Trimming for subdivision surfaces. Comput Aided Geom Des 18(5):463–481
189. Liu W, Mann S (1997) An optimal algorithm for expanding the composition of polynomials. ACM Trans Graph 16(2):155–178
190. Loop C (1987) Smooth subdivision surfaces based on triangles. Master's Thesis, University of Utah
191. Luken WL (1996) Tessellation of trimmed NURBS surfaces. Comput Aided Geom Des 13(2):163–177
192. Ma YL, Hewitt WT (2003) Point inversion and projection for NURBS curve and surface: control polygon approach. Comput Aided Geom Des 20(2):79–99
193. Maekawa T, Patrikalakis NM (1993) Computation of singularities and intersections of offsets of planar curves. Comput Aided Geom Des 10(5):407–429
194. Mäntylä M (1988) An introduction to solid modeling. Computer Science Press, Rockville, Md
195. Markot RP, Magedson RL (1989) Solutions of tangential surface and curve intersections. Comput Aided Des 21(7):421–427
196. Martin RC (2009) Clean code: a handbook of Agile software craftsmanship, 1st edn. Pearson Education, Inc., Upper Saddle River
197. Martin T, Cohen E, Kirby M (2008) Volumetric parameterization and trivariate B-spline fitting using harmonic functions. In: ACM symposium on solid and physical modeling, New York, NY, USA, p 269–280
198. Martin W, Cohen E, Fish R, Shirley P (2000) Practical ray tracing of trimmed NURBS surfaces. J Graph Tools 5(1):27–52
199. Marussig B (2016) Seamless integration of design and analysis through boundary integral equations. Monographic Series TU Graz: structural analysis. Verlag der Technischen Universität Graz
200. Marussig B, Zechner J, Beer G, Fries T (2016) Integration of design and analysis through boundary integral equations. In: VII European congress on computational methods in applied sciences and engineering, ECCOMAS. https://eccomas2016.org/proceedings/pdf/5812.pdf
201. Marussig B, Zechner J, Beer G, Fries TP (2015) Fast isogeometric boundary element method based on independent field approximation. Comput Methods Appl Mech Eng 284:458–488
202. Marussig B, Zechner J, Beer G, Fries TP (2016) Stable isogeometric analysis of trimmed geometries. Comput Methods Appl Mech Eng. doi:10.1016/j.cma.2016.07.040
203. Massarwi F, Elber G (2016) A B-spline based framework for volumetric object modeling. Comput Aided Des 78:36–47
204. Milenkovic VJ (1988) Verifiable implementations of geometric algorithms using finite precision arithmetic. Artif Intell 37(1):377–401
205. Miller JR (1986) Sculptured surfaces in solid models: issues and alternative approaches. IEEE Comput Graph Appl 6(12):37–48
206. Moreton H (2001) Watertight tessellation using forward differencing. In: Proceedings of the workshop on graphics hardware. ACM, p 25–32
207. Mortenson ME (1997) Geometric modeling, 2nd edn. Wiley, New York
208. Mudur SP, Koparkar PA (1984) Interval methods for processing geometric objects. IEEE Comput Graph Appl 4(2):7–17
209. Nagel RN, Braithwaite WW, Kennicott PR (1980) Initial graphics exchange specification (IGES) version 1.0. NBSIR 80-1978 (R). National Bureau of Standards
210. Nagy AP, Abdalla MM, Gurdal Z (2010) On the variational formulation of stress constraints in isogeometric design. Comput Methods Appl Mech Eng 199(41–44):2687–2696
211. Nagy AP, Benson DJ (2015) On the numerical integration of trimmed isogeometric elements. Comput Methods Appl Mech Eng 284:165–185
212. Nguyen T, Karčiauskas K, Peters J (2014) A comparative study of several classical, discrete differential and isogeometric methods for solving Poisson's equation on the disk. Axioms 3(2):280–299
213. Nguyen VP, Kerfriden P, Brino M, Bordas SPA, Bonisoli E (2014) Nitsche's method for two and three dimensional NURBS patch coupling. Comput Mech 53(6):1163–1182
214. Nishita T, Sederberg TW, Kakimoto M (1990) Ray tracing trimmed rational surface patches. SIGGRAPH Comput Graph 24(4):337–345
215. Nitsche J (1971) Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind. Abh Math Semin Univ Hambg 36(1):9–15
216. Omerović S, Fries TP (2016) Conformal higher-order remeshing schemes for implicitly defined interface problems. Int J Numer Methods Eng. doi:10.1002/nme.5301
217. Pabst HF, Springer JP, Schollmeyer A, Lenhardt R, Lessig C, Froehlich B (2006) Ray casting of trimmed NURBS surfaces on the GPU. In: IEEE symposium on interactive ray tracing, p 151–160
218. Parreira P (1988) On the accuracy of continuous and discontinuous boundary elements. Eng Anal 5(4):205–211
219. Patrikalakis NM (1993) Surface-to-surface intersections. IEEE Comput Graph Appl 13(1):89–95
220. Patrikalakis NM, Maekawa T (2002) Chapter 25: intersection problems. In: Handbook of computer aided geometric design. Elsevier, Amsterdam, p 623–650
221. Patrikalakis NM, Maekawa T (2009) Shape interrogation for computer aided design and manufacturing. Springer, Berlin
222. Patrikalakis NM, Prakash PV (1990) Surface intersections for geometric modeling. J Mech Des 112(1):100–107
223. Patterson C, Sheikh M (1984) Interelement continuity in the boundary element method. In: Brebbia C (ed) Topics in boundary element research. Springer, New York, pp 123–141
224. Peng QS (1984) An algorithm for finding the intersection lines between two B-spline surfaces. Comput Aided Des 16(4):191–196
225. Peterson AF, Bibby MM (2009) An introduction to the locally-corrected Nyström method. Synth Lect Comput Electromagn 4(1):1–115
226. Pharr M, Kolb C, Gershbein R, Hanrahan P (1997) Rendering complex scenes with memory-coherent ray tracing. In:

Proceedings of the conference on computer graphics and interactive techniques. ACM, p 101–108
227. Philipp B, Breitenberger M, D'Auria I, Wüchner R, Bletzinger KU (2016) Integrated design and analysis of structural membranes using the isogeometric B-Rep analysis. Comput Methods Appl Mech Eng 303:312–340
228. Piegl LA (2005) Ten challenges in computer-aided design. Comput Aided Des 37(4):461–470
229. Piegl LA, Richard AM (1995) Tessellating trimmed NURBS surfaces. Comput Aided Des 27(1):16–26
230. Piegl LA, Tiller W (1997) The NURBS book, 2nd edn. Springer, New York
231. Piegl LA, Tiller W (1998) Geometry-based triangulation of trimmed NURBS surfaces. Comput Aided Des 30(1):11–18
232. Pratt MJ (2001) Introduction to ISO 10303-the STEP standard for product data exchange. Technical report. National Institute of Standards and Technology, Manufacturing Systems Integration Division, Gaithersburg
233. Pratt MJ, Anderson BD, Ranger T (2005) Towards the standardized exchange of parameterized feature-based CAD models. Comput Aided Des 37(12):1251–1265
234. Pratt MJ, Kim J (2006) Experience in the exchange of procedural shape models using ISO 10303 (STEP). In: Symposium on solid and physical modeling. ACM, p 229–238
235. Purcell TJ, Buck I, Mark WR, Hanrahan P (2002) Ray tracing on programmable graphics hardware. ACM Trans Graph 21(3):703–712
236. Ramshaw L (1987) Blossoming: a connect-the-dots approach to splines. Digital Equipment Corporation, Palo Alto
237. Randrianarivony M (2006) Geometric processing of CAD data and meshes as input of integral equation solvers. PhD Thesis, Computer Science Faculty Technische Universität Chemnitz
238. Randrianarivony M (2009) On global continuity of Coons mappings in patching CAD surfaces. Comput Aided Des 41(11):782–791
239. Rank E, Ruess M, Kollmannsberger S, Schillinger D, Düster A (2012) Geometric modeling, isogeometric analysis and the finite cell method. Comput Methods Appl Mech Eng 249–252:104–115
240. Renner G, Weiß V (2004) Exact and approximate computation of B-spline curves on surfaces. Comput Aided Des 36(4):351–362
241. Requicha AA, Rossignac JR (1992) Solid modeling and beyond. IEEE Comput Graph Appl 12(5):31–44
242. Requicha AAG (1980) Representations for rigid solids: theory, methods, and systems. ACM Comput Surv 12(4):437–464
243. Requicha AAG, Voelcker HB (1982) Solid modeling: a historical summary and contemporary assessment. IEEE Comput Graph Appl 2(2):9–24
244. Requicha AAG, Voelcker HB (1983) Solid modeling: current status and research directions. IEEE Comput Graph Appl 3(7):25–37
245. Requicha AAG, Voelcker HB (1985) Boolean operations in solid modeling: boundary evaluation and merging algorithms. Proc IEEE 73(1):30–44
246. Riesenfeld RF (1975) On Chaikin's algorithm. Comput Graph Image Process 4(3):304–310
247. Riesenfeld RF, Haimes R, Cohen E (2015) Initiating a CAD renaissance: multidisciplinary analysis driven design: framework for a new generation of advanced computational design, engineering and manufacturing environments. Comput Methods Appl Mech Eng 284:1054–1072
248. Riffnaller-Schiefer A, Augsdörfer UH, Fellner DW (2016) Isogeometric shell analysis with NURBS compatible subdivision surfaces. Appl Math Comput 272, Part 1:139–147
249. Rockwood A, Heaton K, Davis T (1989) Real-time rendering of trimmed surfaces. In: ACM SIGGRAPH computer graphics, vol 23. ACM, p 107–116
250. Rossignac JR, Requicha AAG (1987) Piecewise-circular curves for geometric modeling. IBM J Res Dev 31(3):296–313
251. Rossignac JR, Requicha AAG (1999) Solid modeling. Technical report
252. Rüberg T, Cirak F (2012) Subdivision-stabilised immersed B-spline finite elements for moving boundary flows. Comput Methods Appl Mech Eng 209–212:266–283
253. Ruess M, Schillinger D, Bazilevs Y, Varduhn V, Rank E (2013) Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method. Int J Numer Methods Eng 95(10):811–846
254. Ruess M, Schillinger D, Özcan AI, Rank E (2014) Weak coupling for isogeometric analysis of non-matching and trimmed multi-patch geometries. Comput Methods Appl Mech Eng 269:46–71
255. Salesin D, Stolfi J, Guibas L (1989) Epsilon geometry: building robust algorithms from imprecise computations. In: Proceedings of the symposium on computational geometry. ACM, p 208–217
256. Sarraga RF (1983) Algebraic methods for intersections of quadric surfaces in GMSOLID. Comput Vis Graph Image Process 22(2):222–238
257. Sarraga RF, Waters WC (1984) Free-form surfaces in GMSOLID: goals and issues. In: Pickett MS, Boyse 'JW (eds) Solid modeling by computers. Springer, New York, pp 187–209
258. Schillinger D, Evans JA, Reali A, Scott MA, Hughes TJR (2013) Isogeometric collocation: cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations. Comput Methods Appl Mech Eng 267:170–232
259. Schillinger D, Ruess M (2015) The finite cell method: a review in the context of higher-order structural analysis of CAD and image-based geometric models. Arch Comput Methods Eng 22(3):391–455
260. Schmidt R, Wüchner R, Bletzinger KU (2012) Isogeometric analysis of trimmed NURBS geometries. Comput Methods Appl Mech Eng 241–244:93–111
261. Schollmeyer A, Fröhlich B (2009) Direct trimming of NURBS surfaces on the GPU. ACM Trans Graph 28(3):47:1–47:9
262. Scott MA (2011) T-splines as a design-through-analysis technology. PhD Thesis, University of Texas Computational Science, Engineering, and Mathematics
263. Scott MA, Simpson RN, Evans JA, Lipton S, Bordas SPA, Hughes TJR, Sederberg TW (2013) Isogeometric boundary element analysis using unstructured T-splines. Comput Methods Appl Mech Eng 254:197–221
264. SCRA (2006) STEP application handbook ISO 10303 version 3
265. Sederberg TW (1983) Implicit and parametric curves and surfaces for computer aided geometric design. PhD Thesis, Purdue University
266. Sederberg TW, Anderson DC, Goldman RN (1984) Implicit representation of parametric curves and surfaces. Comput Vis Graph Image Process 28(1):72–84
267. Sederberg TW, Christiansen HN, Katz S (1989) Improved test for closed loops in surface intersections. Comput Aided Des 21(8):505–508
268. Sederberg TW, Li X, Lin HW, Ipson H, Finnigan GT (2008) Watertight trimmed NURBS. ACM Trans Graph 27(3):79:1–79:8
269. Sederberg TW, Nishita T (1990) Curve intersection using Bézier clipping. Comput Aided Des 22(9):538–549
270. Sederberg TW, Zheng J, Bakenov A, Nasri A (2003) T-splines and T-NURCCs. ACM Trans Graph 22(3):477–484

271. Segal M (1990) Using tolerances to guarantee valid polyhedral modeling results. SIGGRAPH Comput Graph 24(4):105–114
272. Seo YD, Kim HJ, Youn SK (2010) Isogeometric topology optimization using trimmed spline surfaces. Comput Methods Appl Mech Eng 199(49–52):3270–3296
273. Seo YD, Kim HJ, Youn SK (2010) Shape optimization and its extension to topological design based on isogeometric analysis. Int J Solids Struct 47(11–12):1618–1640
274. Shah JJ, Mäntylä M (1995) Parametric and feature based CAD/CAM. Wiley, New York
275. Shantz M, Chang SL (1988) Rendering trimmed NURBS with adaptive forward differencing. SIGGRAPH Comput Graph 22(4):189–198
276. Shapiro V (2002) Solid modeling. Handb Comput Aided Geom Des 20:473–518
277. Shen J, Kosinka J, Sabin M, Dodgson N (2016) Converting a CAD model into a non-uniform subdivision surface. Comput Aided Geom Des. doi:10.1016/j.cagd.2016.07.003
278. Shen J, Kosinka J, Sabin MA, Dodgson NA (2014) Conversion of trimmed NURBS surfaces to Catmull–Clark subdivision surfaces. Comput Aided Geom Des 31(7–8):486–498
279. Sheng X, Hirsch BE (1992) Triangulation of trimmed surfaces in parametric space. Comput Aided Des 24(8):437–444
280. Sherbrooke EC, Patrikalakis NM (1993) Computation of the solutions of nonlinear polynomial systems. Comput Aided Geom Des 10(5):379–405
281. Shewchuk JR (2002) Delaunay refinement algorithms for triangular mesh generation. Comput Geom Theory Appl 22(1–3):21–74
282. Siemens Product Lifecycle Management Software, Inc. (2008) Parasolid XT format reference. Siemens Product Lifecycle Management Software, Inc., Cambridge
283. Sinha P, Klassen E, Wang KK (1985) Exploiting topological and geometric properties for selective subdivision. In: Proceedings of the first annual symposium on computational geometry. ACM, p 39–45
284. Skytt V, Haenisch J (2013) Extension of ISO 10303 with isogeometric model capabilities. ISO TC 184/SC 4/WG 12. ISO
285. Song XW, Sederberg TW, Zheng JM, Farouki RT, Hass J (2004) Linear perturbation methods for topologically consistent representations of free-form surface intersections. Comput Aided Geom Des 21(3):303–319
286. Steidl JW (2013) Trimmed NURBS: implementation of an element type discrimination algorithm. Master Project, Institute for Structural Analysis at Graz University of Technology
287. Steidl JW, Fries TP (2016) Automatic conformal decomposition of elements cut by NURBS. In: Papadrakakis M, Papadopoulos V, Stefanou G, Plevris V (eds) ECCOMAS congress
288. Stenberg R (1995) On some techniques for approximating boundary conditions in the finite element method. J Comput Appl Math 63(1):139–148
289. Sticko S, Kreiss G (2016) A stabilized Nitsche cut element method for the wave equation. Comput Methods Appl Mech Eng 309:364–387
290. Stroud I (2006) Boundary representation modelling techniques. Springer, London
291. Sugihara K, Iri M (1989) A solid modelling system free from topological inconsistency. J Inf Process 12(4):380–393
292. Sutherland IE, Sproull RF, Schumacker RA (1974) A characterization of ten hidden-surface algorithms. ACM Comput Surv 6(1):1–55
293. Sweeney MAJ, Bartels RH (1986) Ray tracing free-form B-spline surfaces. IEEE Comput Graph Appl 6(2):41–49
294. Tassey G, Brunnermeier SB, Martin SA (1999) Interoperability cost analysis of the U.S. automotive supply chain. Technical report. Research Triangle Institute
295. Temizer I, Wriggers P, Hughes TJR (2011) Contact treatment in isogeometric analysis with NURBS. Comput Methods Appl Mech Eng 200(9–12):1100–1112
296. Toshniwal D, Speleers H, Hughes TJR (2017) Smooth cubic spline spaces on unstructured quadrilateral meshes with particular emphasis on extraordinary points: design and analysis considerations. Technical report. ICES Reports
297. Toth DL (1985) On ray tracing parametric surfaces. SIGGRAPH Comput Graph 19(3):171–179
298. Urick B (2016) Reconstruction of tensor product spline surfaces to integrate surface–surface intersection geometry and topology while maintaining inter-surface continuity. PhD Thesis, The University of Texas at Austin
299. da Veiga LB, Buffa A, Sangalli G, Vázquez R (2014) Mathematical analysis of variational isogeometric methods. Acta Numer 23:157–287
300. Vigo M, Brunet P (1995) Piecewise linear approximation of trimmed surfaces. In: Hagen H, Farin G, Noltemeierm H (eds) Geometric modelling: Dagstuhl 1993, computing supplement 10. Springer, New York, pp 341–356
301. Vries-Baayens AE, Seebregts CH (1992) Chapter 7: exact conversion of a trimmed nonrational Bézier surface into composite or basic nonrational Bézier surfaces. In: Topics in surface modeling. SIAM, Philadelphia, p 115–144
302. Wald I, Slusallek P, Benthin C, Wagner M (2001) Interactive rendering with coherent ray tracing. Comput Graph Forum 20(3):153–165
303. Wang SW, Shih ZC, Chang RC (2000) An improved rendering technique for ray tracing Bézier and B-spline surfaces. J Vis Comput Animat 11(4):209–219
304. Wang X (2001) Geometric trimming and curvature continuous surface blending for aircraft fuselage and wing shapes. Master's Thesis, Virginia Polytechnic Institute and State University
305. Wang Y, Benson DJ (2016) Geometrically constrained isogeometric parameterized level-set based topology optimization via trimmed elements. Front Mech Eng 1–16
306. Wang Y, Benson DJ, Nagy AP (2015) A multi-patch nonsingular isogeometric boundary element method using trimmed elements. Comput Mech 56(1):173–191
307. Wang YW, Huang ZD, Zheng Y, Zhang SG (2013) Isogeometric analysis for compound B-spline surfaces. Comput Methods Appl Mech Eng 261–262:1–15
308. Warren J, Weimer H (2001) Subdivision methods for geometric design: a constructive approach, 1st edn. Morgan Kaufmann Publishers, Inc., San Francisco
309. Wei X, Zhang Y, Hughes TJR, Scott MA (2015) Truncated hierarchical Catmull–Clark subdivision with local refinement. Comput Methods Appl Mech Eng 291:1–20
310. Weiler KJ (1986) Topological structures for geometric modeling. PhD Thesis, Rensselaer Polytechnic Institute
311. Whitted T (1980) An improved illumination model for shaded display. Commun ACM 23(6):343–349
312. Wriggers P, Zavarise G (2004) Chapter 6: computational contact mechanics. In: Stein E, de Borst R, Hughes TJR (eds) Encyclopedia of computational mechanics. Fundamentals, vol 2. Wiley, p 195–226
313. Wu R, Peters J (2015) Correct resolution rendering of trimmed spline surfaces. Comput Aided Des 58:123–131
314. Xia S, Qian X (2016) Isogeometric analysis with Bézier tetrahedra. Comput Methods Appl Mech Eng. doi:10.1016/j.cma.2016.09.045
315. Xia S, Wang X, Qian X (2015) Continuity and convergence in rational triangular Bézier spline based isogeometric analysis. Comput Methods Appl Mech Eng 297:292–324
316. Yang CG (1987) On speeding up ray tracing of B-spline surfaces. Comput Aided Des 19(3):122–130

317. Yang YJ, Cao S, Yong JH, Zhang H, Paul JC, Sun JG, Gu Hj (2008) Approximate computation of curves on B-spline surfaces. Comput Aided Des 40(2):223–234
318. Yap CK (1997) Towards exact geometric computation. Comput Geom 7(1):3–23
319. Zechner J, Marussig B, Beer G, Fries TP (2015) The isogeometric Nyström method. Comput Methods Appl Mech Eng 306:212–237
320. Zhang X (2005) Optimal geometric trimming of B-spline surfaces for aircraft design. PhD Thesis, Virginia Polytechnic Institute and State University
321. Zhang YJ (2016) Geometric modeling and mesh generation from scanned images. CRC Press
322. Zienkiewicz OC, Taylor RL, Zhu JZ (2013) Chapter 17: the finite element method: its basis and fundamentals. In: Automatic mesh generation, 7th edn. Butterworth-Heinemann, Oxford, p 573–640