A survey on global continuity in geometric processing of CAD objects for the Wavelet-Galerkin scheme

Maharavo Randrianarivony

Institute of Computer Science
Christian-Albrecht University of Kiel, Germany

September 6, 2007

Abstract: We need to decompose the boundary of a solid into four-sided patches F_i such that there is a regular mapping γ_i from the unit square to each F_i. In this paper, we focus on the analysis of the global continuity of the mappings γ_i over the whole surface. Since we use Coons functions to generate the mappings γ_i, we demonstrate theoretically that if all curves are parametrized in arc length then the functions γ_i match well at surface joints. That result is valid for any blending functions of the Coons patches. We will describe a reparametrization technique whose goal is to keep the shape of the initial curves while achieving arc length parametrization. The reparametrization process is done by using cubic Bézier spline approximation whose accuracy is estimated in L^∞ norm. For a rational Bézier curve with bounded weights, we develop an algorithm for length computation with an accuracy of $\mathcal{O}(2^{-n})$. The generalization of that result for other types of curves will be discussed. Numerical results are provided to support the theoretical studies. Furthermore, the decomposition techniques are applied to real CAD data which come from IGES files.

1 Introduction

The Wavelet-Galerkin method [21, 3, 25] is a very efficient numerical approach for solving engineering problems. That is shown by its ability to produce arbitrary accuracy with low computational cost [7] by means of adaptivity [5, 23]. The rate between cost and accuracy has been demonstrated to be optimal [6] as specified by N-term approximation. While its theoretical advantages have been completely proved [6, 7, 31], its implementation has not been successfully applied to real CAD data in comparison to

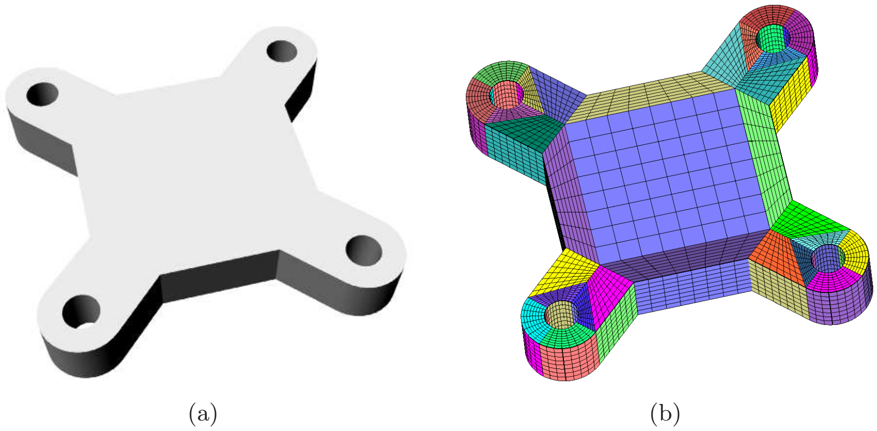


Figure 1: (a)Input geometric model (b)Decomposition and mapping.

mesh-based approaches like FEM so far. That is caused by the complexity of surface structure that the Wavelet Galerkin scheme requires as specified in [31]. Some wavelet experts [4] refer to such geometric imperfections as *curse of geometry*. If we see yet this situation from another perspective, geometric specification in Wavelet-Galerkin scheme is an advantage because it is possible to treat mesh-free problems. In particular, it is obvious that a lot of geometric accuracies are lost by using a mesh or a piecewise linear approximation of the initial geometry. This paper addresses the problem of preparing geometric data so that they can be efficiently used as input in Wavelet-Galerkin methods. In fact, this is a continuation of the geometric splitting method in [26] where nothing about global continuity has been treated at all. Our main contribution in this paper is about global continuity of the mappings in the four-sided patches. In particular, we will comment about the correlation between the Coons patch which resides in an individual patch and the global continuity. On the other hand, we need to show the independence of the global continuity on the chosen blending function. Note that no knowledge of Wavelet-Galerkin scheme is necessary to understand this paper because we only develop geometric algorithms. Although we deal with Wavelet-Galerkin scheme, we believe that the geometric structure here can be used for other schemes like panel-clustering [17, 18] because we have mappings starting from the 2D unit square which can serve as generation of hierarchical structure from 2D to 3D.

In the next section, we will describe more accurately the representation of the geometry that is required in Wavelet-Galerkin scheme. Afterwards, we will summarize the method [26] of splitting a given closed surface which bounds a solid into four-sided patches. The main results of this paper can be found in section 4 where theoretical background and practical realization of global continuity are examined. Since it is not practical to reparametrize the curves exactly, we have to use approximated reparametrizations. We will rigorously analyze the accuracy of that approximation which makes use of cubic Bézier splines. We show how to manage the error of the length approximation in order not to deteriorate the accuracy of $\mathcal{O}(h^4)$ of the cubic spline interpolation. Since we have to compute lengths repeatedly, we show in section 5 how to efficiently evaluate the length values. We will concentrate first on rational Bézier curves which we subdivide recursively. An accuracy of $\mathcal{O}(2^{-n})$ can be achieved after n subdivisions. The results for rational Bézier curves can be generalized for other types of curves like B-spline or NURBS. At the end of the paper, we will supplement our theoretical description by numerical results. Since we have implemented our method with real CAD data, we will also report on our results about decomposition of surfaces coming from IGES files.

2 Problem setting

Let us consider a closed surface $S \subset \mathbf{R}^3$ given as a collection of M trimmed [2] parametric surfaces $S_1, \cdots, S_M$ defined on the domains $\mathcal{D}_1, \cdots, \mathcal{D}_M$ which are multiply connected regions in $\mathbf{R}^2$. The external and internal (when relevant) boundary curves of each domain $\mathcal{D}_i$ are supposed to be composite curves. That is, there are univariate smooth functions κ_i^j defined on $[e_i^j, f_i^j]$ such that

$$\partial\mathcal{D}_i = \bigcup_j \mathrm{Im}(\kappa_i^j). \tag{1}$$

We suppose further that the parametric functions defining S_i

$$\psi_i : \mathcal{D}_i \longrightarrow S_i \tag{2}$$

are regular in the sense that they are bijective, smooth and that their Jacobian matrices have maximal rank. A graphical illustration of this formulation can be found in Fig. 2. Furthermore, we need that the bounding curves are regular, smooth and without zero angles. More precisely, we must have the following conditions for each $\mathcal{D}_i$:

(B1) For all j, we have $\dot{\kappa}_i^j(\tau) \neq \mathbf{0}$.

(B2) Each κ_i^j is $\mathcal{C}^r$continuously differentiable for sufficiently large r.

(B3) Suppose that the terminating point of κ_i^j and the starting point of κ_i^k coincide. We must have

$$\lim_{t \to (f_i^j)^-} \dot{\kappa}_i^{j_1}(t) \neq -\lambda \lim_{t \to (e_i^k)^+} \dot{\kappa}_i^{j_2}(t) \qquad \forall\, \lambda > 0. \tag{3}$$

The objective of this paper is to prepare CAD objects so that they can be used in Wavelet-Galerkin scheme. That means, we will tessellate the surface S into m four-sided domains F_i

$$S = \bigcup_{i=1}^{m} F_i, \tag{4}$$

where the splitting is conforming. That is to say, every two different nondisjoint patches F_i and F_j share either a complete edge or a single corner. We need also some regular functions γ_i such that

$$F_i = \gamma_i([0,1]^2). \tag{5}$$

Additionally, we require global continuity which means that for two adjacent patches F_i and F_j, there is a bijective mapping A such that

$$\gamma_i(s) = \gamma_j(A(s)) \qquad \forall s \in \partial[0,1]^2. \tag{6}$$

The whole geometric operation can be graphically summarized in Fig. 1 where the grids represent the images by γ_i of a uniform grid on the unit square. For real CAD data, we are not able to achieve the exact global continuity (6). As a consequence, we will have only matching condition with certain accuracy $\varepsilon > 0$ such that

$$\mathrm{dist}[\gamma_i(s), \gamma_j(A(s))] < \varepsilon \qquad \forall s \in \partial[0,1]^2. \tag{7}$$

3 Summary of the decomposition procedure

It is beyond the scope of this paper to provide a detailed description of decomposing a 3D-model. We are summarizing only the main steps and point out the principal difficulties [26] which have to be confronted in practice.

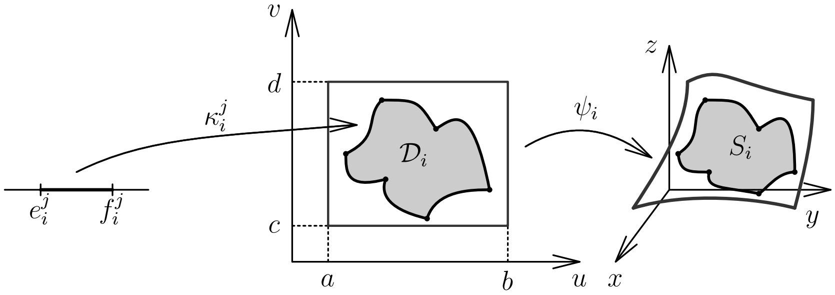
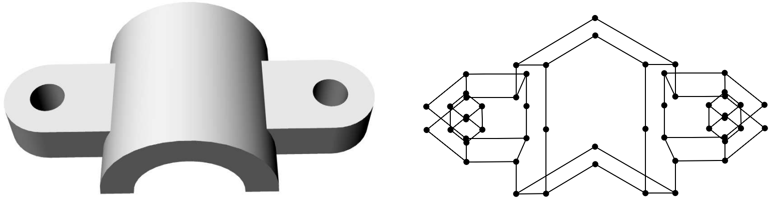


Figure 2: The boundary of $\mathcal{D}_i \subset \mathbf{R}^2$ is the image of several curves κ_i^j. Apply ψ_i to have the trimmed surface S_i.



(a) (b)

Figure 3: (a)Given 3D model (b)Polygonal approximation.

3.1 Four-sided splitting

As a starting step of the decomposition, we approximate the curved boundaries of $\{S_i\}$ by straight line segments separated by nodes $\{X_k\} \subset \mathbf{R}^3$ as in Fig. 3. In order to achieve that approximation while having conforming splitting in mind, we create planar polygonalizations of $\{\mathcal{D}_i\}_{i\in\Lambda}$ which amount to doing the following. For each trimmed surface S_i, we generate a polygon $P^{(i)}$ whose nodes $\mathbf{x}_k^{(i)}$ are taken from the curved boundary of the 2D domain $\mathcal{D}_i$. We have to make sure that for two adjacent different surfaces S_i and S_j sharing a curve $\mathcal{C}$, if $\psi_i(\mathbf{x}_k^{(i)}) \in \mathcal{C}$, then there must exist a vertex $\mathbf{x}_l^{(j)} \in P^{(j)}$ such that

$$\psi_i(\mathbf{x}_k^{(i)}) = \psi_j(\mathbf{x}_l^{(j)}). \tag{8}$$

Let us note that if we take too few vertices, the resulting polygon $P^{(i)}$ may have artifacts such that its edges do not form an admissible polygon as illustrated in Fig. 4(a). But if the polygonal approximation is too fine, then it results in overly many four-sided surfaces. As a consequence, one has to split the curved edges adaptively while trying to maintain relation (8) which involves some preimage computations. Let us emphasize that only polygons having an *even* number of boundary vertices can be decomposed into quadrilaterals. It is not straightforward to convert odd faces into even ones inside a closed surface with arbitrary genus. One should assemble the adjacency graph which is used in the Dijkstra algorithm to search for the shortest path joining two odd polygons in order that the number of additional nodes to be inserted are not too many. We could theoretically prove that the number of odd faces must be even for a closed model and that the odd faces can be converted to even ones pairwise.

Our main approach to achieve (4) consists in splitting the 2D regions $\mathcal{D}_i$ into four-sided regions $Q_{k,i}$ such that $\mathcal{D}_i = \bigcup_k Q_{k,i}$. The foursided patches F_k are therefore the images by ψ_i of the 2D domains $Q_{k,i}$

$$F_k = \psi_i(Q_{k,i}). \tag{9}$$

As for the decomposition into $Q_{k,i}$, we consider the polygon $P^{(i)}$ which we

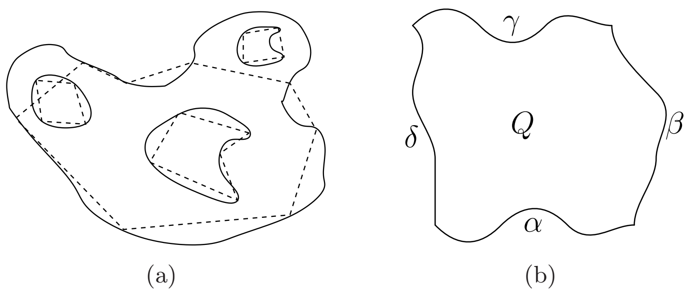


Figure 4: (a)Artifacts in polygonal approximation (b)Four-sided region.

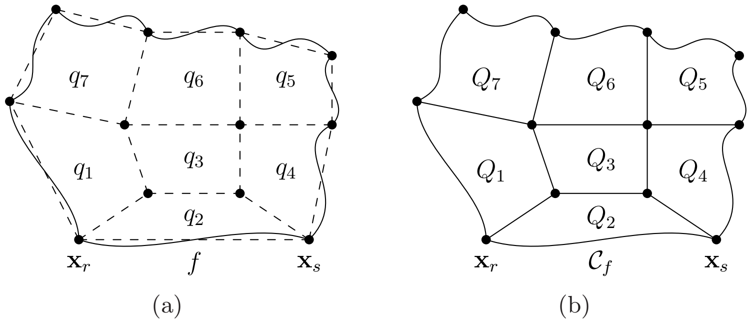


Figure 5: Convertion of a quadrangulation $\{q_k\}$ into subregions having curved sides $\{Q_k\}$.

decompose into a set of convex quadrilaterals $q_{k,i}$. The four-sided domains $Q_{k,i}$ are obtained from $q_{k,i}$ by replacing the straight boundary edges of $q_{k,i}$ by the corresponding curve portion of $\mathcal{D}_i$ as illustrated in Fig. 5. In the decomposition of a polygon $P^{(i)}$ into quadrilaterals $q_{k,i}$, we use only the preimages $\psi_i^{-1}(X_k)$ of the nodes $\{X_k\}$ as boundary vertices. That is, we do not use any additional boundary nodes in the course of the quadrangulation process. We have developed in [26] an approach that decomposes a polygon with n boundary vertices into $\mathcal{O}(n)$ convex quadrilaterals. There are two main difficulties in that process. First, converting nonconvex quadrilaterals into convex ones requires many cases to be handled individually. Second, finding cuts connecting an internal boundary and the exterior boundary of a multiply connected polygon is complicated.

The process of replacing a boundary edge with the corresponding curve can generate three serious problems. First, it is possible that the curve intersects an internal edge causing a boundary interference. The second problem is that some corners in a four-sided region $Q_{k,i}$ might be smoothened out. Third, it is possible that the corresponding Coons patch is not a diffeomorphism. In those cases, we have to make a polygonal refinement. We have developed a method for making only a small local repairment while keeping the large part of the quadrangulation. The things about which one has to be careful is that it is not easy to detect those three problems and we have to guarantee relation (8) when we insert new nodes.

3.2 Transfinite interpolation using Coons maps

This section will be occupied by the description of the mapping from the unit square to each of the four-sided domains $Q := Q_{k,i} \subset \mathbf{R}^2$ from relation (9). Since our method of generating that map is based on transfinite interpolation, we briefly recall some basic facts about this technique [14, 15, 32]. Suppose that Q is delineated by four curves α, β, γ, $\delta : [0,1] \longrightarrow \mathbf{R}^2$ that fulfill the compatibility conditions at the corners:

$$\alpha(0) = \delta(0)\,, \quad \alpha(1) = \beta(0)\,, \quad \gamma(0) = \delta(1)\,, \quad \gamma(1) = \beta(1). \tag{10}$$

We assume that besides the common points in (10), there are no further intersection points as in Fig. 4(b). We are interested in generating a parametric surface $\mathbf{x}(u,v)$ defined on the unit square $[0,1]^2$ such that the boundary of the image of $\mathbf{x}$ coincides with the given four curves:

$$\begin{array}{lll} \mathbf{x}(u,0) = \alpha(u) & \mathbf{x}(u,1) = \gamma(u) & \forall\, u \in [0,1] \\ \mathbf{x}(0,v) = \delta(v) & \mathbf{x}(1,v) = \beta(v) & \forall\, v \in [0,1]\ . \end{array} \tag{11}$$

This transfinite interpolation problem can be solved by a first order Coons patch $\mathbf{x}$ which can be defined in matrix form as

$$\mathbf{x}(u,v) = -\begin{bmatrix} -1 \\ F_0(u) \\ F_1(u) \end{bmatrix}^T \begin{bmatrix} \mathbf{0} & \mathbf{x}(u,0) & \mathbf{x}(u,1) \\ \mathbf{x}(0,v) & \mathbf{x}(0,0) & \mathbf{x}(0,1) \\ \mathbf{x}(1,v) & \mathbf{x}(1,0) & \mathbf{x}(1,1) \end{bmatrix} \begin{bmatrix} -1 \\ F_0(v) \\ F_1(v) \end{bmatrix}. \tag{12}$$

The blending functions F_0 and F_1 denote two arbitrary smooth functions satisfying [16, 33]:

$$F_i(j) = \delta_{ij} \quad i,j = 0,1 \quad \text{and} \qquad F_0(t) + F_1(t) = 1 \quad \forall\, t \in [0,1]. \tag{13}$$

In practical cases, the blending functions are chosen as in the next table.

Type	$F_0(t)$	$F_1(t)$
Linear	$1-t$	t
Cubic	$B_0^3(t) + B_1^3(t)$	$B_2^3(t) + B_3^3(t)$
Trigonometric	$\cos^2(0.5\pi t)$	$\sin^2(0.5\pi t)$

In order to check whether a planar Coons map is a diffeomorphism, we have developed in [27] some efficient criteria which are easy to verify in practice.

The mapping γ_i of relation (5) from the unit square to the foursided patch $F_i \subset \mathbf{R}^3$ is the composition of the function ψ_i and the Coons map. On account of the decomposition algorithm of the former section, we note that the boundaries α, β, γ, δ of the Coons maps are either straight edges or restrictions of the images of the curves κ_i^j.

4 Global continuity

If the boundary curves κ_i^j from (1) are parametrized arbitrarily, there is no guarantee that the global continuity in (6) is fulfilled. On the other hand, we cannot modify the base surfaces ψ_i because they are given as input in the initial CAD storage. Therefore, our objective is to replace the 2D curves κ_i^j by $\tilde{\kappa}_i^j$ so that they have the same shapes ($\mathrm{Im}(\kappa_i^j) = \mathrm{Im}(\tilde{\kappa}_i^j)$) but they have different parametrizations. Let us denote by ρ_i^j the composition $\psi_i \circ \kappa_i^j$ and let us introduce the length function

$$\chi_i^j(t) := \int_{e_i^j}^{t} \left\| \frac{d\rho_i^j}{dt}(\theta) \right\| d\theta. \tag{14}$$

This function is defined from $[e_i^j, f_i^j]$ to $[0, L]$ where L is the total length of the curve ρ_i^j. On account of the properties of κ_i^j and ψ_i that we met in section 2, let us observe that

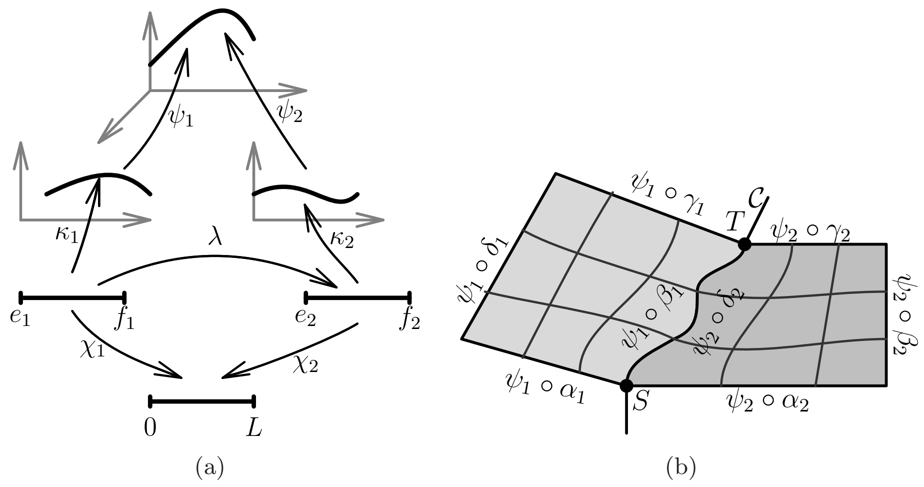
$$\frac{d\chi_i^j}{dt}(t) = \left\| \frac{d\rho_i^j}{dt}(t) \right\| \neq 0 \qquad \forall t. \tag{15}$$

Hence, there is an inverse function $\phi_i^j := (\chi_i^j)^{-1}$ and our objective is to replace the function κ_i^j by $\tilde{\kappa}_i^j := \kappa_i^j \circ \phi_i^j$.

4.1 Chord length vs. Coons patch

In this section we want to demonstrate that if we use the chord length parametrization ϕ_i^j then two adjacent Coons patches verify matching conditions at the interface curve. Additionally, we will show that that result holds irrespective of the chosen blending functions F_0 and F_1. To that end, let us consider two adjacent trimmed surfaces S_i and S_j. In order to facilitate the presentation, we may suppose that they are S_1 and S_2 and we omit the superscripts. Further, we denote by $[e_1, f_1]$ and $[e_2, f_2]$ the intervals of definition of $\psi_1 \circ \kappa_1$ and $\psi_2 \circ \kappa_2$ which have coinciding images:

$$(\psi_1 \circ \kappa_1)([e_1, f_1]) = (\psi_2 \circ \kappa_2)([e_2, f_2]) =: \mathcal{C}. \tag{16}$$



(a) (b)

Figure 6: (a)Involved functions for adjacent Coons maps (b)Two Coons maps match well at interface curve $\mathcal{C}$ for chord length parametrization.

In the following, we denote two Coons maps by $\mathbf{x}_1$ and $\mathbf{x}_2$ which are respectively incident upon κ_1 and κ_2. In addition, we will assume that we have coincidence of the images but we do not necessarily have *pointwise* agreement. That is, one of the following relations hold for some $u_1, u_2, v_1, v_2 \in \{0, 1\}$

$$\begin{aligned}
\mathrm{Im}(\psi_1[\mathbf{x}_1(u_1, \cdot)]) &= \mathrm{Im}(\psi_2[\mathbf{x}_2(u_2, \cdot)]) \\
\mathrm{Im}(\psi_1[\mathbf{x}_1(u_1, \cdot)]) &= \mathrm{Im}(\psi_2[\mathbf{x}_2(\cdot, v_2)]) \\
\mathrm{Im}(\psi_1[\mathbf{x}_1(\cdot, v_1)]) &= \mathrm{Im}(\psi_2[\mathbf{x}_2(u_2, \cdot)]) \\
\mathrm{Im}(\psi_1[\mathbf{x}_1(\cdot, v_1)]) &= \mathrm{Im}(\psi_2[\mathbf{x}_2(\cdot, v_2)]).
\end{aligned} \tag{17}$$

Our claim in the next theorem is that if we reparametrize into chord length parametrization, then we have pointwise coincidence. While reading its proof, we recommend that the reader compare them with Fig. 6.

Theorem 1 Suppose that we use the chord length parametrization $\tilde{\kappa}_1 = \kappa_1 \circ \phi_1$ and $\tilde{\kappa}_2 = \kappa_2 \circ \phi_2$ and that one of the relations in (17) is fulfilled. Then, the images of the Coons maps have sides which agree *pointwise* irrespective of the blending functions F_0 and F_1. That is, one of the following relations must hold:

$$\begin{aligned}
\psi_1[\tilde{\mathbf{x}}_1(u_1, t)] &= \psi_2[\tilde{\mathbf{x}}_2(u_2, t)] & \forall\, t \in [0, 1] \\
\psi_1[\tilde{\mathbf{x}}_1(u_1, t)] &= \psi_2[\tilde{\mathbf{x}}_2(t, v_2)] & \forall\, t \in [0, 1] \\
\psi_1[\tilde{\mathbf{x}}_1(t, v_1)] &= \psi_2[\tilde{\mathbf{x}}_2(u_2, t)] & \forall\, t \in [0, 1] \\
\psi_1[\tilde{\mathbf{x}}_1(t, v_1)] &= \psi_2[\tilde{\mathbf{x}}_2(t, v_2)] & \forall\, t \in [0, 1],
\end{aligned} \tag{18}$$

where $u_1, u_2, v_1, v_2 \in \{0, 1\}$.

Proof

Since ρ_2 is invertible, we may define $\lambda := \rho_2^{-1} \circ \rho_1$. By definition of chord length parametrization, we have

$$\chi_1(t) = \int_{e_1}^{t} \left\| \frac{d\rho_1}{dt}(\theta) \right\| d\theta \qquad \chi_2(t) = \int_{e_2}^{t} \left\| \frac{d\rho_2}{dt}(\theta) \right\| d\theta. \tag{19}$$

From the definition of λ we obtain by chain rule

$$\frac{d\rho_1}{dt}(t) = \frac{d\rho_2}{dt}[\lambda(t)]\lambda'(t). \tag{20}$$

As a consequence, we deduce

$$\chi_1(t) = \int_{e_1}^{t} \left\| \frac{d\rho_1}{dt}(\theta) \right\| d\theta = \int_{e_1}^{t} \left\| \frac{d\rho_2}{dt}[\lambda(\theta)] \right\| \lambda'(\theta) d\theta. \tag{21}$$

After making the change of variable $\sigma := \lambda(\theta)$, we obtain:

$$\chi_1(t) = \int_{\lambda(e_1)}^{\lambda(t)} \left\| \frac{d\rho_2}{dt}(\sigma) \right\| d\sigma = \int_{e_2}^{\lambda(t)} \left\| \frac{d\rho_2}{dt}(\sigma) \right\| d\sigma = \chi_2(\lambda(t)). \tag{22}$$

Hence

$$\chi_2^{-1} \circ \chi_1 = \lambda = \rho_2^{-1} \circ \rho_1. \tag{23}$$

Therefore, if we denote the total length by $L := \chi_1(f_1) = \chi_2(f_2)$, we have

$$(\rho_2 \circ \phi_2)(t) = (\rho_1 \circ \phi_1)(t) \qquad \forall\, t \in [0, L]. \tag{24}$$

Let $S, T \in \mathcal{C}$ be the starting and terminating points of the common side. That is, there are $s_1, \theta_1 \in [e_1, f_1]$ and $s_2, \theta_2 \in [e_2, f_2]$ such that

$$S = (\psi_1 \circ \kappa_1)(s_1) = (\psi_2 \circ \kappa_2)(s_2) \tag{25}$$

$$T = (\psi_1 \circ \kappa_1)(\theta_1) = (\psi_2 \circ \kappa_2)(\theta_2). \tag{26}$$

From (24), we deduce

$$\chi_1(s_1) = \chi_2(s_2) =: s \in [0, L] \tag{27}$$

$$\chi_1(\theta_1) = \chi_2(\theta_2) =: \theta \in [0, L]. \tag{28}$$

We assume only the first relation of (17) with $u_1 = 1$ and $u_2 = 0$ and we are proving the first equality of (18) while the other 15 cases can be treated in a similar manner. That means, let us assume that the sides of the Coons maps $\mathbf{x}_1$ and $\mathbf{x}_2$ are β_1 and δ_2. That is, β_1 is the restriction of κ_1 on $[s_1, \theta_1]$ and δ_2 is the restriction of κ_2 on $[s_2, \theta_2]$:

$$\beta_1(t) = \kappa_1[t\theta_1 + (1-t)s_1] \qquad t \in [0, 1] \tag{29}$$

$$\delta_2(t) = \kappa_2[t\theta_2 + (1-t)s_2] \qquad t \in [0, 1]. \tag{30}$$

Hence,

$$\psi_1 \circ \beta_1 \circ \phi_1(t) = (\rho_1 \circ \phi_1)[t\theta + (1-t)s] \qquad t \in [0,1]. \tag{31}$$

Similarly we have for δ_2:

$$\psi_2 \circ \delta_2 \circ \phi_2(t) = (\rho_2 \circ \phi_2)[t\theta + (1-t)s] \qquad t \in [0,1]. \tag{32}$$

From relation (24), we obtain:

$$\psi_1 \circ \beta_1 \circ \phi_1(t) = \psi_2 \circ \delta_2 \circ \phi_2(t) \qquad \forall\, t \in [0,1]. \tag{33}$$

We denote $\tilde{\beta}_1 := \beta_1 \circ \phi_1$ and $\tilde{\beta}_2 := \beta_2 \circ \phi_2$. Since we have $\tilde{\mathbf{x}}_1(1,v) = \tilde{\beta}_1(v)$ and $\tilde{\mathbf{x}}_2(0,v) = \tilde{\delta}_2(v)$ independently of the chosen blending functions F_0, F_1, we can deduce from (33) that the Coons patches match well as specified in (18).

□

4.2 Practical realization and error analysis

In order that the reader is not confused, let us note that the reparametrization steps take place before computing Coons patches. In fact, when we load the initial geometric information from an IGES input, the first thing that we should do is to perform reparametrization of the bounding curves κ_i^j.

Unfortunately we are not able to solve the problem of global continuity exactly and we cannot find in the literature any method which can do that. As a consequence, we will only show how to solve that problem numerically without too much computational cost. What we should avoid in practice is to pick 3D samples in the image curve $\mathcal{C}$ and to compute the preimages numerically (say with Newton iteration method). That approach has three major drawbacks. First, that is computationally expensive because we need many inverse operations. Second, the involved functions are generally nonlinear. Furthermore, that is not very stable when there is CAD flaw which is a large gap between two input surfaces.

In the next discussion, we show how to approximate the reparametrization function $\phi = \chi^{-1}$ which we met in (15). Additionally, we will examine the accuracy of the approximation in terms of the size of the samples. In the description of the method, we suppose that we have a fast method of evaluating the arc length $\mathcal{L}(\mathbf{x},\mathbf{y})$ between two points $\mathbf{x}$, $\mathbf{y}$ belonging to the curve $\mathcal{C}$.

Let $\{s_i\}_{i=0,\dots,n} \subset [e,f]$ be some samples such that $s_i < s_{i+1}$ and $s_0 = e$ and $s_n = f$. Let $\{\mathbf{x}_i\}_{i=0,\dots,n}$ be their images by $\psi \circ \kappa$ and define $L_i := \mathcal{L}(\mathbf{x_i}, \mathbf{x_{i+1}})$ for $i = 0,...,n-1$. We introduce now $t_i := \sum_{k=0}^{i} L_k \in [0,L]$ where L is the length of the whole curve. Afterwards, we consider the samples $\{(t_i, s_i),\ i = 0, ..., n\}$ and we interpolate them by a composite cubic Bézier curve ϕ_h:

$$\begin{aligned} \phi_h(t_i) &= s_i \qquad \forall\, i = 0, ..., n, && (34) \\ \phi_h(t) &= \sum_{j=0}^{3} b_{3i+j} B_j^3(t) \qquad \forall t \in [t_i, t_{i+1}]. && (35) \end{aligned}$$

Since the use of cubic spline interpolation is very well understood [1], we use that standard interpolation method. In particular, we have the following well known accuracy result.

Lemma 1 *[13, 8, 20]* Let f_h be the cubic spline interpolating the samples $[t_i, f(t_i)]$ which are obtained from a function f. By defining $h := \max |t_i - t_{i+1}|$, we have the following accuracy of the approximation

$$\|f - f_h\|_\infty \leq Ch^4. \tag{36}$$

In practice, it is generally infeasible to compute the lengths exactly. Therefore, we want to examine the influence of the length error to the function approximation. More precisely, on account of length evaluation inexactness, let us suppose that we have evaluated $\{\tau_i\}$ instead of $\{t_i\}$. Let $\tilde{\phi}_h$ be the cubic spline which is computed with the help of $\{\tau_i\}$. The accuracy result for $\|\phi - \tilde{\phi}_h\|_\infty$ still holds true on condition that the length computation has certain accuracy.

Theorem 2 Suppose that the error in length computation is

$$\max_{i=0,...,n} |t_i - \tau_i| = \mathcal{O}(h^\alpha), \tag{37}$$

where $\alpha \geq 1$ is some fixed integer where $h := \max |t_i - t_{i+1}|$. Then, we have the following approximation accuracy

$$\|\phi - \tilde{\phi}_h\|_\infty = \mathcal{O}(h^{\min\{4,\alpha\}}). \tag{38}$$

Proof

On the one hand, we have the following accuracy of the cubic spline approximation ϕ_h by using exact lengths due to the previous lemma:

$$\|\phi - \phi_h\|_\infty = \mathcal{O}(h^4). \tag{39}$$

We introduce now the piecewise polynomial interpolant σ of degree seven satisfying the following boundary conditions (use for example higher order Hermite interpolants [9, 22]):

$$\sigma(\tau_i) = t_i, \quad \sigma'(\tau_i) = 1, \quad \sigma^{(k)}(\tau_i) = 0 \quad k = 2, 3. \tag{40}$$

A Taylor development reveals that for $\tau \in [\tau_i, \tau_{i+1}]$, we have:

$$\sigma(\tau) - \tau = \sigma(\tau_i) - \tau_i + \sum_{k=1}^{3} \frac{(\tau - \tau_i)^k}{k!} (\sigma^{(k)}(\tau_i) - \delta_{1,k}) + \mathcal{O}(|\tau_i - \tau_{i+1}|^4). \quad (41)$$

On account of the relations (37) and (40), we obtain

$$\sigma(\tau) - \tau = \mathcal{O}(h^\alpha) + \mathcal{O}(|\tau_i - \tau_{i+1}|^4). \quad (42)$$

Additionally, since $\alpha \geq 1$, we note that

$$|\tau_{i+1} - \tau_i| \leq |\tau_{i+1} - \sigma(\tau_{i+1})| + |\tau_i - \sigma(\tau_i)| + |t_i - t_{i+1}| = \mathcal{O}(h). \quad (43)$$

As a consequence,

$$\sigma(\tau) - \tau = \mathcal{O}(h^{\min\{4,\alpha\}}). \quad (44)$$

On the other hand, by applying the Taylor development for the second time, we have

$$\phi(t) = \phi(\sigma(t)) + (t - \sigma(t))\phi'(\sigma(t)) + \mathcal{O}((\sigma(t) - t)^2). \quad (45)$$

Since ϕ' is uniformly bounded, we deduce from the last two relations:

$$|(\phi \circ \sigma)(t) - \phi(t)| = \mathcal{O}(h^{\min\{4,\alpha\}}). \quad (46)$$

Since $\tilde{\phi}_h$ is the cubic interpolant which transform τ_i into s_i, we have the following estimation due to the former lemma

$$\|\phi_h \circ \sigma - \tilde{\phi}_h\|_\infty = \mathcal{O}(h^4). \quad (47)$$

By combining the two last estimations, we obtain

$$\|\phi \circ \sigma - \tilde{\phi}_h\|_\infty = \mathcal{O}(h^{\min\{4,\alpha\}}). \quad (48)$$

Finally, from (39) and (48), we deduce $\|\phi - \tilde{\phi}_h\|_\infty = \mathcal{O}(h^{\min\{4,\alpha\}})$.

□

In practice, we convert the piecewise Bézier $\tilde{\phi}$ into a B-spline curve $\tilde{\phi}(t) = \sum_{i=0}^{n} d_i N_0^n(t)$. It is known [22] that the problem of sample interpolation by cubic Bézier and convertion into a B-spline is equivalent to solving a sparse linear system of the following form

$$\begin{bmatrix} 1 & & & & \\ \alpha_1 & \beta_1 & \gamma_1 & & \\ & \dots & \dots & \dots & \\ & & \alpha_{n-1} & \beta_{n-1} & \gamma_{n-1} \\ & & & & 1 \end{bmatrix} \begin{bmatrix} d_0 \\ d_1 \\ \dots \\ d_{n-1} \\ d_n \end{bmatrix} = \begin{bmatrix} b_0 \\ (\Delta_0 + \Delta_1)\mathbf{x}_1 \\ \dots \\ (\Delta_{n-2} + \Delta_{n-1})\mathbf{x}_1 \\ b_{3n-1} \end{bmatrix} \quad (49)$$

where $\Delta_i := \tau_i - \tau_{i-1}$ and the coefficients α_i, β_i, γ_i depend exclusively on Δ_i. Note that the above proposed result has three main advantages. First, all samples $\{\mathbf{x}_i\}$ are treated simultaneously. Second, we have a linear problem instead of a nonlinear one. Finally, it is easy to compute the entries of the matrix which is banded and diagonal dominant.

5 Length estimation

Since the formerly developed method invokes many evaluations of the length function, our next objective is to design an algorithm for estimating the length of a curve $\mathbf{x}$ inside an interval $[a,b]$

$$\mathcal{L} := \int_a^b \|\mathbf{x}'(t)\| dt. \tag{50}$$

We would like also to investigate the rate of convergence of the approximation. Without loss of generality, we suppose that the curve is defined on $[0,1]$ and we compute the whole length i.e. $a=0$, $b=1$. The general case where $[a,b] \neq [0,1]$ can be treated in a very similar way. We will not need to use quadrature rules to estimate the integral in (50) because the structure of the function $\mathbf{x}$ is known [30, 36]. Our preferred method is to apply subdivision recursively while using some flatness criterion [19, 12] in order to know if the curve is close to be linear. As a starting step, we suppose that the curve is a rational Bézier curve

$$\mathbf{x}(t) := \frac{\sum_{i=0}^{m} \omega_i \mathbf{b}_i B_i^m(t)}{\sum_{i=0}^{m} \omega_i B_i^m(t)}, \tag{51}$$

where we have in mind that m is small (say $m \leq 4$). Additionally, we assume that the weights ω_i are uniformly bounded:

$$\exists\, R_1 > 0, R_2 > 0 \quad \text{such that} \quad R_1 < \left| \sum_{i=0}^{m} \omega_i B_i^m(t) \right| < R_2 \qquad \forall\, t \in [0,1]. \tag{52}$$

Although we mainly deal with rational Bézier curves, treatments of other types of curves will be discussed at the end of this section.

5.1 Rational Bézier subdivision

Let us first introduce some notations related to the successive subdivision of an arbitrary Bézier function

$$S(t) = \sum_{i=0}^{m} \mathbf{s}_i B_i^m(t). \tag{53}$$

Let $\mathbf{s}_i^{(j)}$ be the points which are found by using the de Casteljau [9] algorithm at $t = 0.5$, i.e. $\mathbf{s}_i^{(j+1)} := 0.5(\mathbf{s}_i^{(j)} + \mathbf{s}_{i+1}^{(j)})$ and $\mathbf{s}_i^{(0)} := \mathbf{s}_i$. The function $S^{[0,0]} := S$ can be split into two Bézier functions $S^{[1,1]}$ and $S^{[1,2]}$ which have respectively the control points $\mathbf{s}_i^{[1,1]} := \mathbf{s}_0^{(i)}$ and $\mathbf{s}_i^{[1,2]} := \mathbf{s}_i^{(m-i)}$ and which are defined on $[0,0.5]$ and $[0.5,1]$. We can apply that process successively in order to obtain from each Bézier function $S^{[p-1,i]}$ two Bézier functions $S^{[p,2i-1]}$ and $S^{[p,2i]}$. That is, after applying subdivisions n times we have 2^n Bézier curves $S^{[n,k]}$ whose control points are denoted by $\mathbf{s}_i^{[n,k]}$ for $k = 1, ..., 2^n$ and $i = 0, ..., m$. Each function $S^{[n,k]}$ is defined on the interval $[p_{k-1}, p_k]$ where $p_k := k/2^n$.

Now, we want to apply the above subdivision technique to the numerator $\tilde{\mathbf{x}}(\cdot)$ and the denominator $\omega(\cdot)$ of the function in (51). Let us denote the control points of $\tilde{\mathbf{x}}$ by $\tilde{\mathbf{b}}_i := \mathbf{b}_i\omega_i$. The functions $\tilde{\mathbf{x}}(\cdot)$ and $\omega(\cdot)$ will be subdivided into functions $\tilde{\mathbf{x}}^{[n,k]}$ and $\omega^{[n,k]}$ having the control points $\tilde{\mathbf{b}}_i^{[n,k]}$ and $\omega_i^{[n,k]}$. On each subinterval $[p_k, p_{k+1}]$ we introduce the rational Bézier $\mathbf{x}^{[n,k]} := \tilde{\mathbf{x}}^{[n,k]}/\omega^{[n,k]}$. Thus, by defining $\mathbf{b}_i^{[n,k]} := \tilde{\mathbf{b}}_i^{[n,k]}/\omega_i^{[n,k]}$, we have forall $\tau \in [p_k, p_{k+1}]$:

$$\mathbf{x}^{[n,k]}(\tau) = \frac{\sum_{i=0}^m \omega_i^{[n,k]}\mathbf{b}_i^{[n,k]}B_i^m(s)}{\sum_{i=0}^m \omega_i^{[n,k]}B_i^m(s)} \quad \text{where} \quad s = \frac{\tau - p_k}{p_{k+1} - p_k}. \tag{54}$$

Furthermore, we have the restriction property:

$$\mathbf{x}^{[n,k]} = \mathbf{x}_{|_{[p_{k-1},p_k]}}. \tag{55}$$

By considering the interval $[p_{k-1}, p_k]$, we can introduce $\theta_{i,k} := (i/m)p_k + (1 - i/m)p_{k-1}$ for $i = 0, ..., m$. We have the following convergence result for the subdivision of a rational Bézier curve.

Theorem 3 Suppose that the rational Bézier in (51) has been subdivided n times. Then, we have the following accuracy order for all $k = 1, ..., 2^n$ and $i = 0, ..., m$:

$$\|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{b}_i^{[n,k]}\| = \mathcal{O}(2^{-2n}). \tag{56}$$

Proof

Since the function ω is uniformly bounded as specified in (52), there exist constants K_1, K_2 such that

$$\left\|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \frac{\omega_i^{[n,k]}\mathbf{b}_i^{[n,k]}}{\omega^{[n,k]}(\theta_{i,k})}\right\| \leq K_1 \left\|\tilde{\mathbf{x}}^{[n,k]}(\theta_{i,k}) - \omega_i^{[n,k]}\mathbf{b}_i^{[n,k]}\right\|, \tag{57}$$

$$\left\|\mathbf{b}_i^{[n,k]} - \frac{\omega_i^{[n,k]}\mathbf{b}_i^{[n,k]}}{\omega^{[n,k]}(\theta_{i,k})}\right\| \leq K_2 \left|\omega^{[n,k]}(\theta_{i,k}) - \omega_i^{[n,k]}\right|. \tag{58}$$

As a consequence, we obtain

$$\left\|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{b}_i^{[n,k]}\right\| \leq K_1 \left\|\tilde{\mathbf{x}}^{[n,k]}(\theta_{i,k}) - \omega_i^{[n,k]}\mathbf{b}_i^{[n,k]}\right\| + K_2 \left|\omega^{[n,k]}(\theta_{i,k}) - \omega_i^{[n,k]}\right|. \tag{59}$$

On the other hand, let us consider the blossom function $\mathcal{P}$ of the polynomial $\tilde{\mathbf{x}}^{[n,k]}$. We have the relation with the control points [34]:

$$\tilde{\mathbf{b}}_i^{[n,k]} = \mathcal{P}(\underbrace{p_{k-1}, ..., p_{k-1}}_{m-i}, \underbrace{p_k, ..., p_k}_{i}). \tag{60}$$

Thus, we have the following Taylor development:

$$\begin{aligned} \tilde{\mathbf{b}}_i^{[n,k]} &= \mathcal{P}(\theta_{i,k}, ..., \theta_{i,k}) + \sum_{p=1}^{m-i} (p_{k-1} - \theta_{i,k}) \frac{\partial}{\partial x_p} \mathcal{P}(\theta_{i,k}, ..., \theta_{i,k}) + \\ & \sum_{p=m-i+1}^{m} (p_k - \theta_{i,k}) \frac{\partial}{\partial x_p} \mathcal{P}(\theta_{i,k}, ..., \theta_{i,k}) + \mathcal{O}(|p_k - p_{k-1}|^2). \end{aligned}$$

Since $\mathcal{P}$ is symmetric, all partial derivatives in the above relation are the same. Due to the fact that $(m-i)(p_k - \theta_{i,k}) + i(p_{k-1} - \theta_{i,k}) = 0$, we obtain $\tilde{\mathbf{b}}_i^{[n,k]} = \mathcal{P}(\theta_{i,k}, ..., \theta_{i,k}) + \mathcal{O}(|p_k - p_{k-1}|^2)$. As a consequence, we deduce $\tilde{\mathbf{b}}_i^{[n,k]} = \tilde{\mathbf{x}}^{[n,k]}(\theta_{i,k}) + \mathcal{O}(2^{-2n})$. The same analysis can be repeated to the blossom of the polynomial ω in order to obtain $\omega_i^{[n,k]} = \omega(\theta_{i,k}) + \mathcal{O}(2^{-2n})$. Therefore, we can deduce from (59) that $\|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{b}_i^{[n,k]}\| = \mathcal{O}(2^{-2n})$.

□

We can now use that approximation result in order to deduce the accuracy in length computation.

Corollary 1 Let $\mathcal{L}$ be the length of $\mathbf{x}$. Define

$$A_k := \sum_{i=0}^{m-1} \|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{x}^{[n,k]}(\theta_{i+1,k})\|, \quad \text{and} \tag{61}$$

$$B_k := \sum_{i=0}^{m-1} \|\mathbf{b}_i^{[n,k]} - \mathbf{b}_{i+1}^{[n,k]}\| \qquad \forall\, k = 0, ..., 2^n - 1. \tag{62}$$

We claim that $\mathcal{L}_n := \sum_k (\lambda A_k + (1-\lambda) B_k)$ converges to the true length $\mathcal{L}$ in dyadic order:

$$|\mathcal{L} - \mathcal{L}_n| = \mathcal{O}(2^{-n}). \tag{63}$$

Proof

Let l_k be the length of the curve $\mathbf{x}^{[n,k]}(\tau)$. Due to the convex hull property we have

$$A_k \le l_k \le B_k. \tag{64}$$

On the other hand, the difference of the bounds can be estimated as follows

$$\begin{aligned}|B_k - A_k| &= \sum_{i=0}^{m-1} \|\mathbf{b}_i^{[n,k]} - \mathbf{b}_{i+1}^{[n,k]}\| - \|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{x}^{[n,k]}(\theta_{i+1,k})\| \\ &\leq \sum_{i=0}^{m-1} \|[\mathbf{b}_i^{[n,k]} - \mathbf{x}^{[n,k]}(\theta_{i,k})] - [\mathbf{b}_{i+1}^{[n,k]} - \mathbf{x}^{[n,k]}(\theta_{i+1,k})] + \\ &\quad [\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{x}^{[n,k]}(\theta_{i+1,k})]\| - \|\mathbf{x}^{[n,k]}(\theta_{i,k}) - \mathbf{x}^{[n,k]}(\theta_{i+1,k})\| \\ &\leq \sum_{i=0}^{m-1} \|[\mathbf{b}_i^{[n,k]} - \mathbf{x}^{[n,k]}(\theta_{i,k})]\| + \|[\mathbf{b}_{i+1}^{[n,k]} - \mathbf{x}^{[n,k]}(\theta_{i+1,k})]\|.\end{aligned}$$

By using the previous theorem with the last inequality, we deduce $|B_k - A_k| = \mathcal{O}(2^{-2n})$. As a consequence, we obtain $|B_k - l_k| = \mathcal{O}(2^{-2n})$ and $|A_k - l_k| = \mathcal{O}(2^{-2n})$. Hence, the accuracy of the length estimation is given as

$$\begin{aligned}|\mathcal{L} - \mathcal{L}_n| &= \left|\sum_{k=0}^{2^n} l_k - [\lambda A_k + (1-\lambda)B_k]\right| \\ &\leq \sum_{k=0}^{2^n} |\lambda(l_k - A_k) + (1-\lambda)(l_k - B_k)| = 2^n \mathcal{O}(2^{-2n}) = \mathcal{O}(2^{-n}).\end{aligned}$$

□

5.2 Adaptivity and more general curves

We have developed a method which always subdivides each rational Bézier curve into two in each iteration. In this section, we would like to discuss about two possible generalization of that approach. First, we will show how to develop adaptive strategy in order to only apply subdivisions at positions where they are necessary. Second, we will discuss about extensions of results for rational Bézier to curves of other types.

In practice, when the rational Bézier curve is almost linear, we do not need to subdivided it any more. The quantities A_k and B_k of relation (64) are very good values for evaluating the flatness of the curve. We have proven that the difference between A_k and B_k tends to zero. That is, we should only apply subdivision at positions where $\varepsilon_k := |A_k - B_k|$ is large. One can even devise an adaptive strategy where we refine the rational Bézier curves corresponding to the s largest values of ε_k (say $s = 3$). By doing that, we need only to use subdivisions at subintervals where the accuracy is bad.

Note that the requirement in (52) is not a restrictive condition because it can be replaced by specifying that all weights (which are of finite number) are

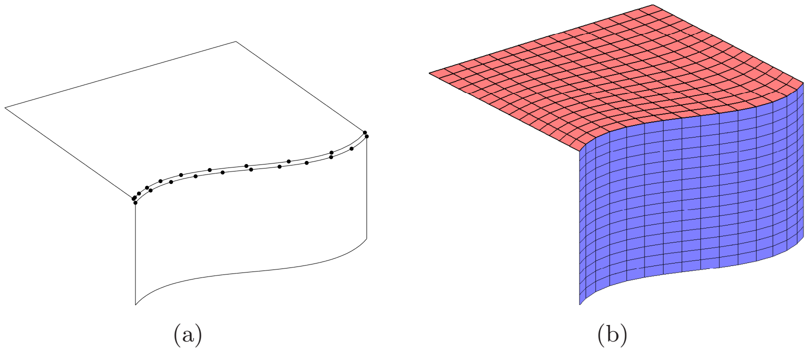


Figure 7: (a)Before reparametrization (b)After reparametrization: Coons patches match well with blending functions $F_0(t) = \sin^2(0.5\pi t)$ and $F_1(t) = \cos^2(0.5\pi t)$.

strictly positive. Of course, the former results hold for Bézier curves because the corresponding weights are unity. As a result, the bounds $R_1 = R_2 = 1$ in relation (52) exist. In order to be able to apply the former results to a B-spline, we have to convert the B-spline first to a piecewise Bézier. That convertion step can be done very fast as described in [9]. Similarly, a NURBS curve can be converted into composite rational Bézier. We have specified in (51) that m is small because we have B-spline or NURBS curves with smoothness $\mathcal{C}^p$ with $p = 2$ at most in practice.

6 Numerical results

In this section, we want to numerically examine the effect of the parametrization to the boundary matchings of adjacent Coons maps. As a test example, let us consider the patches in Fig. 7(a) where we see the images of samples having uniform stepsize. It can be observed that the samples do not at all match at the interface. For the description of the boundary curve at the common side, we use the following five curves κ_i which have the same image but which have different parametrizations

$$\kappa_j(t) = \sum_{i=0}^{3} \mathbf{b}_i B_i^3(p_j(t)) \qquad j = 1, ..., 5, \tag{65}$$

where p_j are distortion functions given by $p_1(t) := t^2$, $p_2(t) := \sqrt{t}$, $p_3(t) := t^3$, $p_4(t) := \sin(\frac{1}{2}\pi t)$, $p_5(t) := t^6$. In the above description, the values of the control points are $\mathbf{b}_0 := (3.5, 0.0)$, $\mathbf{b}_1 := (2.5, 1.0)$, $\mathbf{b}_2 := (4.5, 2.0)$, $\mathbf{b}_3 := (3.5, 3.0)$.

In order to evaluate the nonuniformity of the functions κ_i, we introduce the gauge:

$$\mu := \sum_{j=0}^{n-1} \left| \delta_j - \frac{L}{n} \right| , \tag{66}$$

where δ_j represents the length of the curve κ_i between $\frac{j}{n}$ and $\frac{j+1}{n}$ while L is the whole length. Note that when the curve has arc length parametrization, μ has zero value. A large value of the uniformity gauge μ indicates that the deviation from being arc-length parametrization is very large. We want to investigate the accuracy of matching between the neighboring patches after reparametrization. In Table 1, we collect the maximum errors and the average errors between the two adjacent Coons patches for the curves κ_i. It can be observed that the accuracies after reparametrization are almost the same irrespective of the uniformity of the curves κ_i. That confirms our theoretical predictions that the Coons maps are globally continuous if we use chord length reparametrizations. As graphical illustration, we see in Fig. 7(b) that the Coons patches match well after reparametrizations.

Function	Non-uniformity	Maximum error	Average error
κ_1	1.615948	1.812050e-006	2.939276e-007
κ_2	1.636991	1.394744e-006	2.046115e-007
κ_3	2.469409	1.288736e-006	3.108232e-007
κ_4	1.347950	1.560663e-006	1.819363e-007
κ_5	3.726542	1.289617e-006	2.936744e-007

Table 1: Errors between the adjacent images of two Coons patches.

7 Results for CAD objects

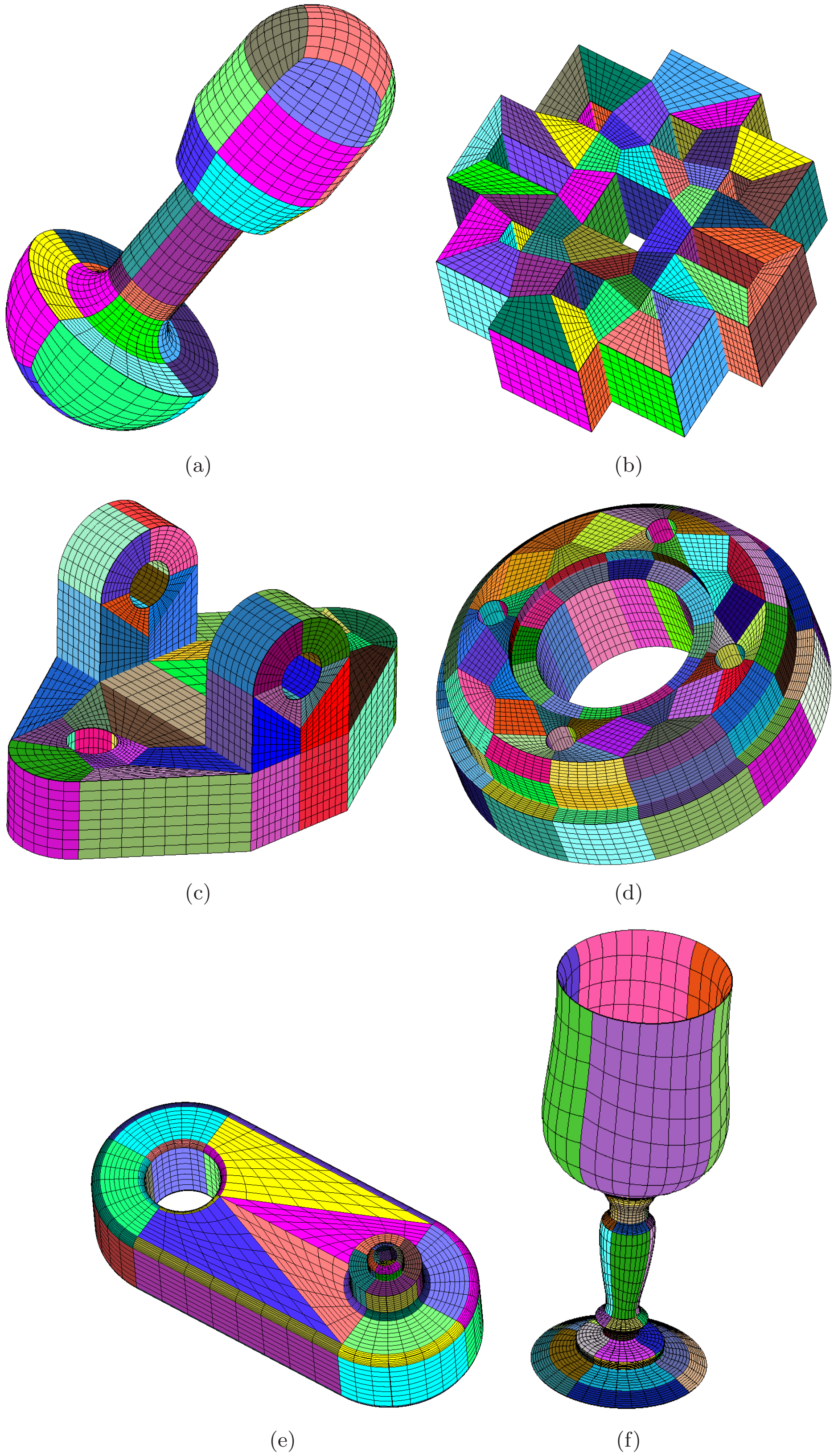
Apart from simulated data, we have equally applied the formerly described algorithms to real CAD objects [28]. Since we choose IGES to serve as our exchange format [35], it was necessary to implement routines that read IGES files. First, we need to assemble routines which can find information about the components of the stored geometry. Special functions have to be implemented in order to locate the positions of separators, IGES sections and IGES records which can be used to identify the values pertaining to IGES entities. On the other hand, we have to implement a large number of extraction routines for the different IGES entities. As a consequence, we must have efficient data structures to organize the components of the stored geometry. Finally, we have to implement an evaluation routine for each data structure to provide access to the needed information in the geometric algorithms.

The initial CAD models corresponding to the mechanical parts which are found in Fig.8(a) through Fig.9(b) have been designed with a CAD system. The resulting IGES files have been used in our algorithms and we have used the approach in section 3 to decompose the mechanical parts into surface patches. We need to reparametrize the boundary curves so that they possess possess arc length parametrizations because the mappings are obtained by using Coons maps. As for the outputs, we have implemented two methods. The first one stores the exact mappings γ_i inside a file as specified in [29] where we developed some functions for easily accessing and evaluating the data without knowledge of CAGD. The second method stores the gridpoints which are the 3D images of dyadic points $(i/2^L, j/2^L) \in [0,1]^2$ where $i, j = 0, ..., 2^L$. In Table 2, we gather the number of entities corresponding to the mechanical parts.

Mechanical parts	Nb surfaces	Nb four-sided patches	Nb gridpoints
Fig.7 (a)	16	38	41,382
Fig.7 (b)	58	136	148,104
Fig.7 (c)	26	90	98,010
Fig.7 (d)	28	212	230,868
Fig.7 (e)	40	95	103,455
Fig.7 (f)	32	70	76,230
Fig.8 (a)	243	727	791,703
Fig.8 (b)	593	2081	2,266,209

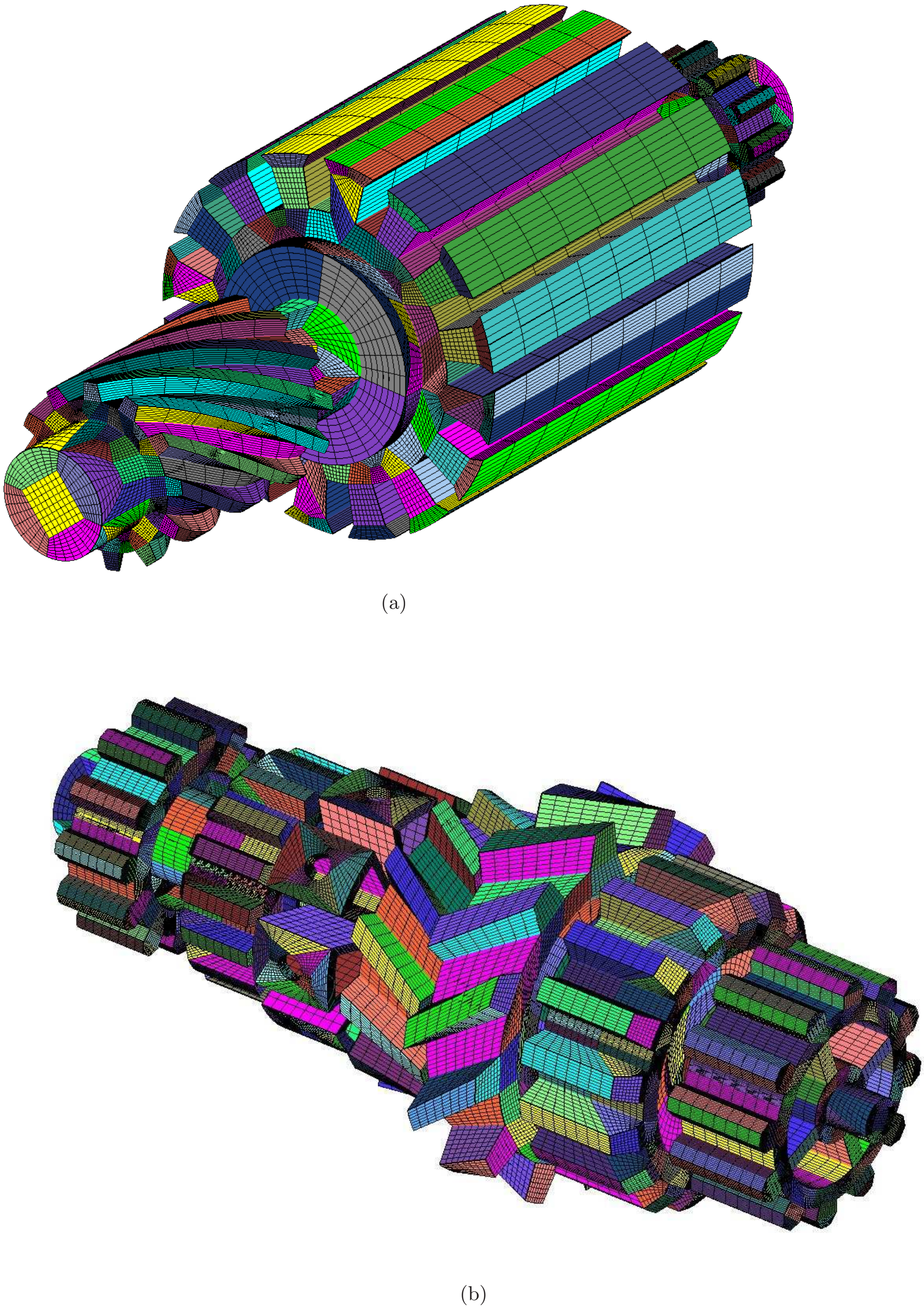
Table 2: Number of initial surfaces, number of foursided patches and number of points at level $L = 5$.

Acknowledgment: The author would like to thank Prof. Reinhold Schneider for helping him to pursue his research work at the Scientific Computing Group of the Christian-Albrecht University of Kiel.



(a) (b) (c) (d) (e) (f)

Figure 8: First set of results.



(a)

(b)

Figure 9: Second set of results.

References

[1] J. Ahlberg, E. Nilson, J. Walsh, The theory of splines and their applications, Academic Press, New York, 1967.

[2] G. Brunnett, Geometric design with trimmed surfaces, Computing Supplementum **10** (1995) 101–115.

[3] A. Cohen, W. Dahmen, R. DeVore, Adaptive wavelet methods for elliptic operator equations-convergence rates, Math. Comp. **70** (2000) 27-75.

[4] A. Cohen, Numerical analysis of wavelet methods, Elsevier Science & Technology, 2003.

[5] S. Dahlke, I. Weinreich, Wavelet-Galerkin-methods: an adapted biorthogonal wavelet basis, Constr. Approx. **9** (1993) 237-262.

[6] W. Dahmen, A. Kunoth, Adaptive wavelet methods for linear–quadratic elliptic control problems: convergence rates, SIAM J. Contr. Optim. **43**(5) (2005) 1640-1675.

[7] W. Dahmen, A. Kunoth, J. Vorlöper, Convergence of adaptive wavelet methods for goal-oriented error estimation, in: Proc. Enumath 2006, pp. 39–60.

[8] C. de Boor, A practical guide to splines, Springer-Verlag, New York, 1978.

[9] G. Farin, Curves and surfaces for computer aided geometric design: a practical guide, Academic Press, Boston, 1997.

[10] G. Farin, Discrete Coons patches, Comput. Aided Geom. Des. **16**, No. 7 (1999) 691–700.

[11] G. Farin, S. Hahmann, G. Brunnett, in: Geometric Modelling, Dagstuhl 2002, Computing **72**, Wien, 2004, pp. 1–246.

[12] O. Figueiredo, J. Reveilles, R. Hersch, Digitization of Bézier curves and patches using discrete geometry, in: Proc. 8th international conference Discrete Geometry for Computer Imagery, Marne-la-Vallée, France 1999, pp. 388-398.

[13] M. Floater, Chordal cubic spline interpolation is fourth order accurate, IMA J. Numer. Anal. **26** (2006) 25–33.

[14] A. Forrest, On Coons and other methods for the representation of curved surfaces, Comput. Graph. Img. Process. **1** (1972) 341–359.

[15] W. Gordon, C. Hall, Construction of curvilinear co-ordinate systems and applications to mesh generation, Int. J. Numer. Methods Eng. **7** (1973) 461–477.

[16] W. Gordon, C. Hall, Transfinite element methods: blending-function interpolation over arbitrary curved element domains, Numer. Math. **21** (1973) 109–129.

[17] W. Hackbusch, Integral equations: theory and numerical treatment, International series of numerical mathematics 120, Basel: Birkhaüser, 1995.

[18] W. Hackbusch, C. Lage, S. Sauter, On the efficient realization of sparse matrix techniques for integral equations with focus on panel clustering, cubature and software design aspects, in: Proc. final conf. of priority research programme boundary element methods, Stuttgart, 1997, pp. 51–75.

[19] T. Hain, Rapid termination evaluation for recursive subdivision of Bézier curves, in: Proc. international conference on Image Science, Systems, and Technology, Las Vegas 2002, pp. 323–328.

[20] C. Hall, W. Meyer, Optimal error bounds for cubic spline interpolation, J. Approx. Theory **16** (1976) 105–122.

[21] H. Harbrecht, R. Schneider, Biorthogonal wavelet bases for the boundary element method, Math. Nachr. **269-270** (2004) 167–188.

[22] J. Hoschek, D. Lasser, Grundlagen der geometrischen Datenverarbeitung, Teubner, Stuttgart, 1989.

[23] A. Kunoth, Adaptive wavelet schemes for an elliptic control problem with Dirichlet boundary control, Numer. Algor. **39** (1-3) (2005) 199-220.

[24] H. Prautzsch, W. Boehm, M. Paluszny, Bézier and B-Spline techniques, Springer, Berlin, 2002.

[25] A. Rathsfeld, A wavelet algorithm for the solution of the double layer potential equation over polygonal boundaries, J. Int. Equations Appl. **7** (1994) 47-98.

[26] M. Randrianarivony, Geometric processing of CAD data and meshes as input of integral equation solvers, PhD thesis, Technische Universität Chemnitz, 2006.

[27] M. Randrianarivony, G. Brunnett, Necessary and sufficient conditions for the regularity of a planar Coons map, Preprint SFB393/04-07, 2004.

[28] M. Randrianarivony, Software pertaining to the preparation of CAD data from IGES interface for mesh-free and mesh-based numerical solvers (IntegralCAD version 0.9.1), Monarch preprint 0026-2007, TU Chemnitz, 2007 (http://archiv.tu-chemnitz.de/pub/2007/0026).

[29] M. Randrianarivony, The ITL programming interface toolkit (API functions Version 0.9.1), Monarch preprint 0025-2007, TU Chemnitz, 2007 (http://archiv.tu-chemnitz.de/pub/2007/0025).

[30] J. Roulier, Specifying the arc length of Bézier curves, Computer Aided Geometric Design **10** (1993) 25–56.

[31] R. Schneider, Multiskalen- und Wavelet-Matrixkompression: Analysisbasierte Methoden zur Lösung grosser vollbesetzter Gleichungssysteme, Teubner, Stuttgart, 1998.

[32] G. Schulze, Blending-Function-Methoden im CAGD, Diplomarbeit, Universität Dortmund, 1986.

[33] G. Schulze, Segmentation operators on Coons' patches, in: Mathematical methods in computer aided geometric design, edts. T. Lyche, L. Schumaker, Academic Press, Boston, 1989, pp. 561–572.

[34] H. Seidel, Polar forms for geometrically continuous spline curves of arbitrary degree. ACM Trans. Graph. **12** (1993) 1-34.

[35] U. S. Product Data Association, Initial Graphics Exchange Specification. IGES 5.3, Trident Research Center, SC, 1996.

[36] M. Walter, A. Fournier, Approximate arc length parametrization, in: Proc. 9th Brazilian symposium on computer graphics and image processing, Brazil, 1996, pp. 143–150.