Algorithm ⋆: Bi-cubic splines for polyhedral control nets

JÖRG PETERS*, University of Florida, USA
KYLE SHIH-HUANG LO*, University of Florida, USA
KĘSTUTIS KARČIAUSKAS, Vilnius University, Lithuania

For control nets outlining a large class of topological polyhedra, not just tensor-product grids, bi-cubic polyhedral splines form a piecewise polynomial, first-order differentiable space that associates one function with each vertex. Akin to tensor-product splines, the resulting smooth surface approximates the polyhedron. Admissible polyhedral control nets consist of quadrilateral faces in a grid-like layout, star-configuration where $n \neq 4$ quadrilateral faces join around an interior vertex, n-gon configurations, where $2n$ quadrilaterals surround an n-gon, polar configurations where a cone of n triangles meeting at a vertex is surrounded by a ribbon of n quadrilaterals, and three types of T-junctions where two quad-strips merge into one.

The bi-cubic pieces of a polyhedral spline have matching derivatives along their break lines, possibly after a known change of variables. The pieces are represented in Bernstein-Bézier form with coefficients depending linearly on the polyhedral control net, so that evaluation, differentiation, integration, moments, etc. are no more costly than for standard tensor-product splines. Bi-cubic polyhedral splines can be used both to model geometry and for computing functions on the geometry. Although polyhedral splines do not offer nested refinement by refinement of the control net, polyhedral splines support engineering analysis of curved smooth objects. Coarse nets typically suffice since the splines efficiently model curved features. Algorithm ⋆ is a C++ library with input-output example pairs and an iges output choice.

CCS Concepts: • **Computing methodologies** → **Parametric curve and surface models**; • **Mathematics of computing** → **Continuous functions**.

Additional Key Words and Phrases: polyhedral spline, free-form surface, n-sided face, C1 spline, T-junction, extraordinary point, polar layout, functions on manifolds, isogeometric analysis

ACM Reference Format:
Jörg Peters, Kyle Shih-Huang Lo, and Kęstutis Karčiauskas. 202x. Algorithm ⋆: Bi-cubic splines for polyhedral control nets. In *Proceedings of* . ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/1122445.1122456

1 MOTIVATION: SPLINES AND POLYHEDRAL CONTROL NETS

The widely-used tensor-product spline [11] efficiently represents differentiable polynomial function spaces, e.g. as finite elements. Under the acronym NURBS, tensor-product splines are widely used to construct curved smooth geometry. Tensor-products lift many uni-variate (single-variable) properties effortlessly to surfaces, volumes and beyond. Tensor-product splines consist of a grid-like arrangement of smoothly-joined polynomial (or rational) pieces that are a linear combination of finite support basis functions, called B-splines, with multipliers, called control points. The support overlap of B-splines can be encoded by joining control points to form a grid, the tensor-product *control net*. In particular when control points have three coordinates, the control net outlines the shape of the spline surface, making it the prevailing paradigm for representing curved free-form

,
,

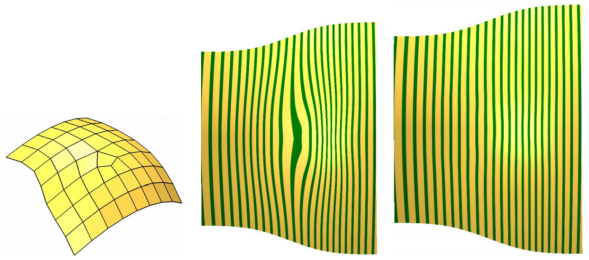
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

geometry: the geometric shape of spline curves and surfaces can be intuitively manipulated via their control net. The control net is therefore a powerful bridge between the discrete computational world the continuous real world.

However, at *irregularities* – where the tensor-structure breaks down (see the 3-valent and the 5-valent point in Fig. 1) – the B-spline and its control net are not well-defined. Commonly used computational function spaces for polyhedral unstructured layout either sacrifice differentiability near irregularities – as in the (dis)continuous Galerkin approach [9]; or they restrict the computation to individual, internally smooth spline regions linked by penalty functions. Spline functions on triangulations [30] and radial basis functions [6] provide built-in smoothness but are restricted to fixed, typically planar domains. That is, they can represent general free-form surfaces only as level sets of a tri-variate function. For modeling complex geometric shapes, tensor-product spline (NURBS) domains are carved up into complex regions to which the spline is restricted. This restriction of the domain is known as *trimming*. Since trimming leads to a plethora of downstream challenges, the animation industry has instead adopted a class of singular splines, called subdivision surfaces [14]. Subdivision surfaces consist of an infinite sequence of nested surface rings and are usually approximated by a fine faceted model [38].

Fig. 1. A polyhedral control net.

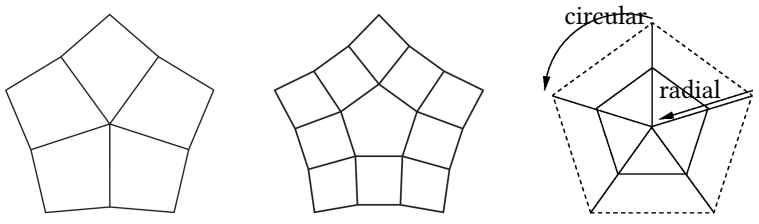
Reduction of smoothness, trimming and subdivision are workarounds that come at a cost. Computational spaces that give up on differentiability when the variational problem calls for differentiability work in too large a computational space and may therefore yield outcomes that are not solutions to the original problem, e.g. return non-physical discontinuous flow lines. Similarly, penalty methods require careful calibration to converge to the proper solution. Geometric models that are based on trimming complicate downstream processing due to heterogeneity in size, parameter orientation, continuity and polynomial degree: blends of blends and revisions often result in minuscule gaps in the geometry sometimes filled in with patches that are little more than slivers with poor aspect ratios. Trimmed many-sided domains and pinched slivers can cause computational instability that require special integration rules for engineering analysis. Subdivision surfaces, too, require special treatment of differentiation and integration near their singularly-parameterized central limit points – and their shape often suffers from flat spots (widening highlight lines) as well as pinched highlight lines, both illustrated in Fig. 2b. See [29] for a detailed analysis. Irregularities are therefore often viewed as an obstacle when devising mathematical software.



(a) polyhedral net (b) subdivision (c) polyhedral spline

Fig. 2. The highlight line distribution [2] (uniform=better) of a typical subdivision surface [7, 14] vs polyhedral spline.

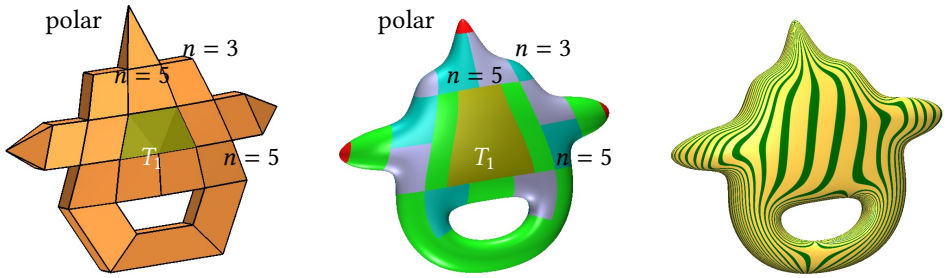
The contribution of the provided software, Algorithm ⋆, is to extend bi-quadratic (bi-2) tensor-product splines to more general polyhedral control net. The polyhedral control net may include several useful types of non-tensor product sub-nets. The cost of allowing non-regular control sub-nets is that the polynomial degree of the pieces is increased to bi-cubic (bi-3).



(a) star-net $n = 5$ (b) n-gon-net, $n = 5$ (c) polar-net $m = 5$

(d) T_0-net (e) T_1-net (f) T_2-net

Fig. 3. Six supported non-tensor-product polyhedral control net patterns. Other n and high m are permissible.



(a) polyhedral control net (b) spline patch layout (c) highlight lines

Fig. 4. Example of a polyhedral spline combining non-tensor product patterns in close proximity. (a) Control net of the polyhedral spline with a tight layout: the top pole node is a direct neighbor of four nodes of valence $n = 5$. 5-valent nodes are direct neighbors (b) Surface layout including bi-quadratic bi-2 splines corresponding to the grid-like parts of the control net, n-sided star-confgurations (blue or gray), bi-3 polar caps, and surface pieces covering T_1-junctions. (c) The highlight lines [2] of the resulting surface flow well-distributed and without kinks confirming good shape and smoothness of the polyhedral spline.

(Extending bi-3 tensor-product splines to polyhedral control nets and surfaces of good shape requires polynomial degree bi-4 or higher, see e.g. [21, 24].) Moreover, some designers prefer least degree to better control the deviation of the surface from the outlining control net, see Fig. 5. Also, when computational use only requires C^1 continuity then bi-2 splines are more efficient than bi-3

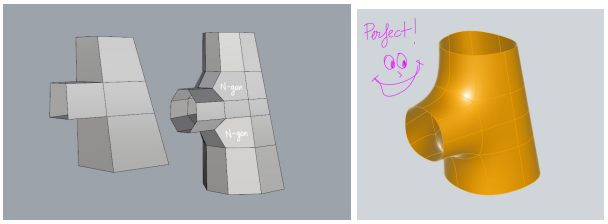


Fig. 5. Branching design without bulging (polyhedron by designer Tony Black)

splines. The polyhedral patterns listed in Fig. 3 can be in close proximity, enabling complex layouts such as Fig. 4. A polyhedral spline is well-defined across irregularities and automatically co-joins the bi-cubic pieces with smooth transitions. Just like the tensor-product control net, the polyhedral control net expresses the neighbor relations of the polyhedral spline generating functions, outlines shape and provides handles for manipulating shape: the polyhedral control net vertices can be used as computational degrees of freedom, e.g. for least squares fitting, computing moments or for solving partial differential equations.

We summarize input and output of Algorithm ⋆, with more detail to follow:

Input: A polyhedral (control) mesh.
Output: A smooth polyhedral spline (consisting of pieces) of degree bi-3.

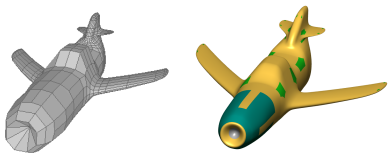
The output polyhedral spline is by default a smooth piecewise polynomial function or surface of degree bi-3 (bi-cubic) expressed piecemeal in Bernstein-Bézier form [13]. Or, by choice, as a collection of uniform B-splines of degree 3 [11] with, near an irregular pattern, a separate net for each polynomial piece. The polynomial pieces join with matching derivatives, possibly after a known reparameterization. There are two ways to introduce *creases or discontinuities* if wanted: by creating facets of zero area, e.g. placing opposing sides of a quad onto each other, or by manipulating individual polynomial coefficients of the output. By default, Algorithm ⋆ smoothly joins polynomial pieces of degree bi-3 arising from

- tensor-product sub-nets (splines of degree bi-2 [13], re-expressed in degree bi-3);
- star-nets with central vertices of valence $n \in \{3, 5, .., 8\}$ [23], see Fig. 3a;
- nets with central n-gons, $n \in \{3, 5, .., 8\}$, [23], see Fig. 3b;
- polar nets [26], $n \in \{3, 4, 5, .., 8\}$, see Fig. 3c;
- T-nets [25] see Fig. 3d, 3e, 3f.

Due to the sharp degree bound proven in [27], there exists no polyhedral control net refinement for bi-3 polyhedral splines with geometric continuity. However, the spline space can be refined by de Casteljau's algorithm [15] applied per piece. The algorithm collects any of the itemized sub-nets of control points and applies a linear transformation to the sub-net to generate the polynomial coefficients of the output pieces. The transformation automatically enforces geometric continuity, i.e. smoothness after change of variables, between the pieces.

1.1 Literature

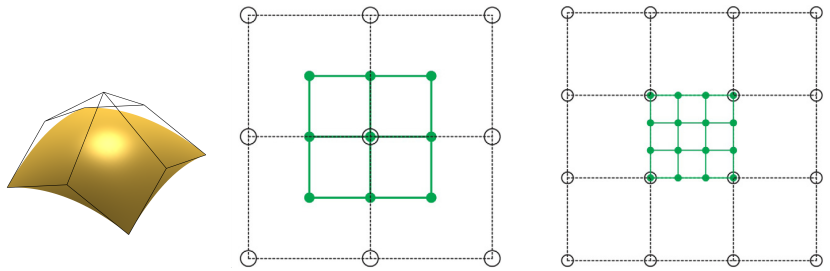
The interested reader may find a survey on spline trimming in the context of isogeometric design in [31] and a survey of splines for meshes with irregularities in [37]. The latter survey characterizes the splines as smooth after a change of variables, referred to as geometric continuity, or singular. Singular construction include those with singularities at corners [34, 35, 40, 46], singular edges [32, 44] and contracting faces, a.k.a. subdivision algorithms [38]. Splines leveraging geometric continuity include [4, 10, 33, 36] and for data fitting and simulation specific to planar domains [3, 18–20].



(a) input (b) output

Fig. 6. Algorithm ⋆. (a) input: polyhedral control net. (b) output: polyhedral spline rendered in Bézierview [39].

Overview. Section 2 describes the input mesh patterns supported by the code. Section 3 explains the (structure of the) output polynomial pieces generated by Algorithm ⋆: how many, how arranged, what degree, and their inter-patch smoothness (see e.g. Fig. 6). Section 4 defines and illustrates



(a) BB-net and bi-2 patch (b) bi-2 spline nets (c) bi-3 spline nets

Fig. 7. B-spline control points ○ [11] form a control net and BB-coefficients ● [13] form a BB-net of $(d+1)^2$ nodes for a polynomial piece of degree bi-d.

the usage of the algorithm; and what to do in the case when the control net is not designed for polyhedral splines in that patterns overlap to form configurations that are not supported. Note that Eqs. (1) – (4) below provide the complete mathematical machinery for understanding polyhedral splines: tensor-product splines and their control net (1), (2), the output surface pieces in BB-form (3) and the smoothness relations (4) of abutting pieces by reparameterization (change of variables).

2 INPUT: POLYHEDRAL CONTROL NET

According to [11] a tensor-product spline in two variables, (u, v) is a piecewise polynomial function

$$\mathbf{p} : (u, v) \rightarrow \mathbf{p}(u, v) := \sum_{i=0}^{k_1} \sum_{j=0}^{k_2} \mathbf{c}_{ij} \mathrm{N}_i^{d_1}(u) \mathrm{N}_j^{d_2}(v). \tag{1}$$

The coefficients $\mathbf{c}_{ij}$ scale the B-splines $\mathrm{N}_i^{d_\ell}$ of degree d_ℓ and act as degrees of freedom, say for finite element computations or to outline shape. The $\mathbf{c}_{ij}$ are therefore often called *control points*. Connecting $\mathbf{c}_{ij}$ to $\mathbf{c}_{i+1,j}$ and $\mathbf{c}_{i,j+1}$ wherever possible yields the *control-net*. Due to variation diminishing property and the convex hull property, the control-net outlines the graph of the spline function, respectively the geometric shape of the spline surface. One of the main attractions of the tensor-product spline is this elegant bridging of the continuous and the discrete representations via the quadrilateral grid of the control net. Computer-aided design (CAD) designers edit the control points to fine-tune a 3D surface while automatically maintaining the desired smoothness.

Tensor-product spline control nets form grids of $k_1 \times k_2$ quadrilateral faces. Specifically for our setup, any three by three sub-net can be interpreted as the control net of the tensor-product of B-splines $\mathrm{N}^2(t-k)$ of degree 2 with a *uniform knot sequence* [13] that define one bi-quadratic (bi-2) polynomial piece:

$$\mathbf{p} : (u, v) \rightarrow \mathbf{p}(u, v) := \sum_{i=0}^{2} \sum_{j=0}^{2} \mathbf{c}_{ij} \mathrm{N}^2(u-i) \mathrm{N}^2(v-j). \tag{2}$$

Meshes consisting of quadrilaterals are popular in polyhedral 3D modeling where quad-strips follow the principal directions and delineate features. When n such directions merge, the control net pattern forms either a star-net with an extraordinary point of valence n at the hub, see Fig. 3a, or an n-gon, see Fig. 3b. Often valences 3 and 5 suffice for modeling since vertices or faces with valencies 6, 7, 8 can be split and the effect distributed by local re-meshing. However, it is convenient to also have the freedom to include $n = 6, 7, 8$ to avoid re-meshing where multiple surface regions meet.

High valence occurs naturally when modeling finger tips and airplane nose cones. A polar configuration is typically used to cap such cylindrical structures. The triangles joining at the pole are interpreted as quadrilaterals with one edge collapsed hence modeled by smoothly joining spline pieces with a (removable) singularity at the pole.

Where two finer quadrilaterals meet a coarser quadrilateral face a T-joint results. This T gives the name to the T_1-gon (formally a pentagonal face). A T_2-gon (formally a hexagon) combines the ends two quad-strips at two T-junctions. The T_0-gon similarly merges neighboring quad-strips but has no T-junction.

Fig. 4 demonstrates that irregularities can be placed in close proximity. The resulting model has few control points and is lightweight in the sense that few higher-order curved elements create often elegant shape. Note though that for the code to work, the overall control net cannot be completely unstructured. All sub-nets have to form one of the following *polyhedral layout configurations* (the restriction to $n < 9$ is only in the distributed code; the underlying theory allows for higher n):

(1) *tensor-configuration*: a control point surrounded by $n = 4$ quads, (Fig. 7b).
(2) *star configuration*: a control point of valence $n \in \{3, 5..8\}$, surrounded by n quads, (Fig. 3a).
(3) *n-gon configuration*: an n-gon, $n \in \{3, 5..8\}$, surrounded by $2n$ quads, one per edge and one additional per vertex, (Fig. 3b).
(4) *T_0-configuration*: a triangle surrounded by quads with two vertices of valence 4 and one of valence 5, (Fig. 3d).
(5) *T_1-configuration*: a pentagon surrounded by quads with four vertices of valence 4 and one of valence 3, (Fig. 3e).
(6) *T_2 configuration*: a hexagon surrounded by quads with three consecutive vertices of valence 4 and two of valence 3 separated by one vertex of valence 4, (Fig. 3f).
(7) *polar configuration*: a control point surrounded by n triangles, $n \in \{3..8\}$, (Fig. 3c).

Note that overlapping star configurations are admissible: a quadrilateral face may have multiple non-4-valent vertices. For configurations other than those listed, the code does not return a corresponding spline piece. [28] provides recipes for minimal localized remeshing to fix impermissible configurations, and a localized refinement that guarantees permissibility at the cost of more pieces.

Since designers spend many years to excel at creating elegant control nets and surfaces, it is not possible to provide a manual of best polyhedral design practices – but three recommendations specific to the increased flexibility of modeling with polyhedral splines can be given. T-gons should appear only isolated by a frame of quadrilaterals to avoid too rapid a transition from coarse to fine and the non-contracting T-gon direction should be aligned with creases to avoid unwanted dips. Localized refinement should be isotropic for configurations (2), (3), (7), but preserve the two dominant directions of a regular surface layout for (4), (5), (6). That is for (2), (3), (7) use of Augmented Refinement from [28] is recommended while for (4), (5), (6) T-refinement from [28] is recommended.

3 OUTPUT: SMOOTHLY-JOINED POLYNOMIAL PIECES OF DEGREE BI-3

Algorithm ⋆ outputs a polyhedral spline as a collection of smoothly-joined polynomial pieces in Bernstein-Bézier form (BB-form) [13, 15]):

$$\mathbf{p}(u,v) := \sum_{i=0}^{d_1}\sum_{j=0}^{d_2} \mathbf{b}_{ij}\mathbf{B}_i^{d_1}(u)\mathbf{B}_j^{d_2}(v), \quad (u,v) \in [0..1]^2 \tag{3}$$

where $\mathbf{B}_k^d(t) := \binom{d}{k}(1-t)^{d-k}t^k \in \mathbb{R}$ are the Bernstein polynomials of degree d and $\mathbf{b}_{ij}$ are the BB-coefficients. For surfaces in 3-space, $\mathbf{b}_{ij} \in \mathbb{R}^3$ and $\mathbf{p}$ is a piece of the surface, called a *patch*. For example, a bi-3 patch has 4×4 BB-coefficients as in Fig. 7c. Connecting $\mathbf{b}_{ij}$ to $\mathbf{b}_{i+1,j}$ and $\mathbf{b}_{i,j+1}$ wherever possible yields the *BB-net*. Note that the BB-net is usually finer than the B-spline control net and represents a single polynomial piece rather than a piecewise polynomial function. (According to [12, 13] any B-spline can be expressed as multiple pieces of polynomials in BB-form and any basis function of the BB-form can be expressed in B-spline form with suitably repeated knots.) The BB-form is evaluated via de Casteljau's algorithm [45], differentiated exactly by forming differences of the BB-coefficients and exactly integrated by forming sums [15].

The output *polyhedral spline pieces* are (see Fig. 4, Fig. 7, Fig. 8 and examples in the user manual):

(1) (tensor-configuration) one bi-2 patch with $3 \times 3 = 9$ BB-coefficients,
(2) (star configuration) $n = 3, 5$ patches of degree bi-3,
respectively $4n$ patches of degree bi-3 for $n > 5$,
(3) (n-gon configuration) $n = 3, 5$ patches of degree bi-3,
respectively $4n$ patches of degree bi-3 for $n > 5$,
(4) (T_0-configuration) 2×2 bi-3 patches,
(5) (T_1-configuration) 4×2 bi-3 patches,
(6) (T_2 configuration) 4×4 bi-3 patches,
(7) (polar configuration) $n \in 3..8$ patches of degree $(3, 2)$ (radial, circular) with $\mathbf{b}_{0j}$, $j = 0, .., 2$ collapsed into the pole; the singular patches are surrounded by a ring of n patches of degree $(2, 2)$.

At global boundaries the outermost layer recedes – matching the behavior of uniform B-splines of degree 2, see Fig. 1. For configurations other than those listed, the code returns no patch. For these impermissible configurations a localized re-meshing is often possible [28]. Alternatively, a single Catmull-Clark refinement step guarantees a permissible quad mesh with star configurations only, i.e. a net that controls a polyhedral spline surface without holes. Caps formed by n patches enclose the central point without gap or overlap. To obtain a uniform degree bi-3 in all cases, any patch of degree lower than bi-3 can be expressed as a patch of degree bi-3 by a process called degree-raising [13, 15]. Uniform degree bi-3 is the default output option.

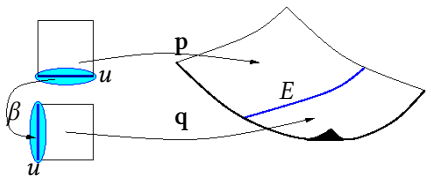


Fig. 9. Geometric smoothness between patches $\mathbf{p}$ and $\mathbf{q}$ to form an atlas.

The polyhedral spline pieces join by default without gaps or overlap and with matching first derivatives – possibly after a reparameterization (a change of variables) as is appropriate for manifolds. Specifically, consider the shared boundary E between two abutting patches $\mathbf{p}$ and $\mathbf{q}$, see Fig. 9. Let $\beta : \mathbb{R}^2 \to \mathbb{R}$, $\beta(u) := (u + b(u)v, a(u)v)$ be a (local) reparameterization and $\mathbf{p}(u, 0) = E = \mathbf{q}(u, 0)$. We say that $\mathbf{q}$ and $\mathbf{p}$ join G^1 if the partial derivatives lie in the same plane and the transversal v-derivatives of the two patches lie on opposite sides with respect to the u-derivative

Fig. 8. From left to right: Input control net (.obj) , output (.bv) polyhedral spline and highlight line distribution of models Tee, Hand, Airplane. Patch types: gold=regular, green=n-valent, silver=polar (finger tips),Cyan=T1.

along the shared boundary:

$$\partial_v \mathbf{q}(u,0) + a(u)\partial_v \mathbf{p}(u,0) = b(u)\partial_u \mathbf{p}(u,0), \quad a(u) \neq 0,\ u \in [0..1]. \tag{4}$$

When $b(u) := 0$ and $a(u) := const$, then we say the spline pieces join parameterically C^1, short C^1. Tensor-product and polar configurations join internally C^1 (the latter with a removable singularity at the pole, see [26]). Otherwise the spline pieces join with geometric smoothness, short G^1. For example, [23] uses the for $n = 3, 5$ the quadratic change of variables

$$a(u) := \frac{1 - \mathrm{c}(1-u)}{1-\mathrm{c}}, \quad b(u) := \frac{\mathrm{c}(1-u)u}{1-\mathrm{c}} \quad \mathrm{c} := \cos\frac{2\pi}{n},$$

to transition from the surrounding surface to one of the n bi-3 patches of the cap and $a(u) := 1$, $b(u) := 2\mathrm{c}(1-u)$ between adjacent bi-3 patches of the cap. The reparameterizations of the T-configurations are listed and displayed in [25] (Table 1 and column (c) of Fig.4 of [25]). Once the reparameterization β is chosen, the polynomial equation (4) becomes a linear system of equations in the coefficients of $\mathbf{p}$ and $\mathbf{q}$. For correctly selected β and degree of $\mathbf{p}$ and $\mathbf{q}$ these systems have been symbolically constructed and inverted , and the residual degrees of freedom chosen to yield good highlight lines for all input nets of an extensive obstacle course [22]. Outcomes for a part of the obstacle course are shown in the user manual. The choice of residual degrees of freedom in the

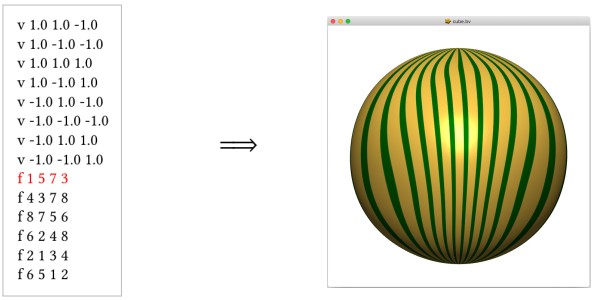
formulas is both heuristic and mathematically grounded. Linear functionals that minimize integrals of derivatives in symbolic form reduce the degrees of freedom to a one or two scalar weights. These remaining weights are set by a line search near an algebraically natural default value and judged by outcomes of highlight line distributions on the obstacle course. This mathematically-constrained heuristic derivation of the formulas is necessary since there is no accepted mathematical theory of what constitutes a well-designed surface – other than the absence of flaws. The absence of flaws is judged by highlight lines flowing as uniformly as possible. Unwanted flat spots (widening highlight lines) and creases (pinched highlight lines) are flaws.

The result are explicit formulas that relate the input polyhedral net to the output BB-coefficients of the polyhedral spline [23], [25], [26]. Algorithm ★ is the implementation of these formulas.

Each quadrilateral bi-3 patch in BB-form can internally be partitioned and replaced by a C^1-connected spline complex that provides a local refinement hierarchy. This simple 'T-spline' space of 'elements with hanging nodes' is not implemented here, but is described for example in [17].

When used to model free-form surfaces, the highlight line distribution is typically best within any of the bi-3 caps for the six irregular layout configurations (2–7) and worst across the regular bi-2 spline parts of the surface. A similar family of formulas, for patches of degree greater than bi-3, generalizes smoother, but also more feature-erasing bi-3 tensor-product splines to polyhedral splines suitable for high-end surface constructions (see e.g. [21, 24]).

4 USAGE



(a) Input: cube.obj

(b) Output surface

Fig. 10. Input of Algorithm ★ and output (b) visualized with highlight lines in Bview [39].

Algorithm ★ is accompanied by a user manual. This section gives a quick overview. The input format is .obj (Wavefront) [43], shown for a cube in Fig. 10a. The input represents a faceted free-form shape, a topological polyhedron, possibly with global boundary and can include any combination of the seven polyhedral layout configurations of Section 2. The output is by default a collection of the BB-coefficients of the pieces of a smooth spline in the BView (.bv) format. We added the option to output rudimentary IGES 128 format. The Bview format is specified at [39]. An online webGL viewer at [39] provides surface inspection tools, such as highlight line rendering in Fig. 10b. Each x, y , z coordinate is a polyhedral spline – in Algorithm ★ all three are packed together to define the x, y, z coordinates of a free-form surface in 3-space. Note that the PatchConsumer (see the code diagram in the user manual) also provides a commented code snippet illustrating differentiation of the polyhedral spline patches).

Unsupported configurations result in holes in the surface. If patch count is not an issue, a global Catmull-Clark refinement [7] of the mesh separates the configurations sufficiently so that

the resulting application of polyhedral splines has no holes due to missing configurations. If the patch count is critical, localized re-meshing is recommended [28].

Applications. Bi-cubic polyhedral splines have many applications: design, visualization, animation, moment computation, re-approximation, reconstruction, computation of partial differential equations on manifolds, etc. . For example, generalizing the higher-order iso-parametric (isogeometric) approach of [1, 5, 8, 16, 41, 42] to polyhedral control nets without additional meshing, [33] uses a sub-class of polyhedral spline functions to solve fourth order partial differential equations, and to compute geodesics on a free-form polyhedral spline surface via the heat equation.

REFERENCES

[1] F.T.K. Au and Y.K. Cheung. 1993. Isoparametric Spline Finite Strip for Plane Structures. *Computers & Structures* 48 (1993), 22–32.
[2] Klaus-Peter Beier and Yifan Chen. 1994. Highlight-line algorithm for realtime surface-quality assessment. *Computer-Aided Design* 26, 4 (1994), 268–277.
[3] Michel Bercovier and Tanya Matskewich. 2017. Smooth Bézier Surfaces over Unstructured Quadrilateral Meshes. *Lecture Notes of the Unione Matematica Italiana* (2017).
[4] Ahmed Blidia, Bernard Mourrain, and Gang Xu. 2020. Geometrically smooth spline bases for data fitting and simulation. *Computer Aided Geometric Design* 78 (March 2020), 101814.
[5] V. Braibant and C. Fleury. 1984. Shape Optimal Design using B-splines. *Computer Methods in Applied Mechanics and Engineering* 44 (1984), 247–267.
[6] Martin D. Buhmann. 2009. *Radial Basis Functions - Theory and Implementations*. Cambridge monographs on applied and computational mathematics, Vol. 12. Cambridge University Press. I–X, 1–259 pages. http://www.cambridge.org/de/academic/subjects/mathematics/numerical-analysis/radial-basis-functions-theory-and-implementations
[7] E. Catmull and J. Clark. 1978. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design* 10 (Sept. 1978), 350–355.
[8] F. Cirak, M. Ortiz, and P. Schröder. 2000. Subdivision surfaces: a new paradigm for thin-shell finite-element analysis. *Internat. J. Numer. Methods Engrg.* 47 (April 2000).
[9] B. (Bernardo) Cockburn, George Karniadakis, and Chi-Wang Shu. 2000. *Discontinuous Galerkin methods: theory, computation, and applications*. Vol. 11. Springer-Verlag Inc., pub-SV:adr. xi + 470 pages.
[10] Annabelle Collin, Giancarlo Sangalli, and Thomas Takacs. 2016. Analysis-suitable G1 multi-patch parametrizations for C1 isogeometric spaces. *Computer Aided Geometric Design* 47 (2016), 93–113.
[11] C. de Boor. 1978. *A Practical Guide to Splines*. Springer.
[12] Carl de Boor. 1986. *B (asic)-Spline Basics*. Technical Report. U of Wisconsin, Mathematics Research Center.
[13] C. de Boor. 1987. B-form basics. In *Geometric Modeling: Algorithms and New Trends*, G. Farin (Ed.). SIAM, 131–148.
[14] Tony DeRose, Michael Kass, and Tien Truong. 1998. Subdivision Surfaces in Character Animation. ACM Press, New York, 85–94.
[15] Gerald Farin. 1988. *Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide*. Academic Press.
[16] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. 2005. Isogeometric Analysis: CAD, Finite Elements, NURBS, Exact Geometry and Mesh Refinement. *Computer Methods in Applied Mechanics and Engineering* 194 (2005), 4135–4195.
[17] Hongmei Kang, Jinlan Xu, Falai Chen, and Jiansong Deng. 2015. A new basis for PHT-splines. *Graphical Models* 82 (2015), 149–159.
[18] Mario Kapl, Giancarlo Sangalli, and Thomas Takacs. 2018. Construction of analysis-suitable G1 planar multi-patch parameterizations. *Computer-Aided Design* 97 (2018), 41–55.
[19] Mario Kapl, Giancarlo Sangalli, and Thomas Takacs. 2019. Isogeometric analysis with C^1 functions on planar, unstructured quadrilateral meshes. *The SMAI journal of computational mathematics* (2019), 67–86.
[20] Mario Kapl, Giancarlo Sangalli, and Thomas Takacs. 2019. An isogeometric C ^{1} subspace on unstructured multi-patch planar domains. *Computer Aided Geometric Design* 69 (2019), 55–75.
[21] Kęstutis Karčiauskas, Thien Nguyen, and Jörg Peters. 2016. Generalizing bicubic splines for modeling and IGA with irregular layout. *Computer-Aided Design* 70 (2016), 23–35.
[22] Kęstutis Karčiauskas and Jörg Peters. [n.d.]. Quad-net obstacle course. http://www.cise.ufl.edu/research/SurfLab/shape_gallery.shtml. Accessed: June 2020.
[23] Kęstutis Karčiauskas and Jörg Peters. 2015. Smooth multi-sided blending of biquadratic splines. *Computers & Graphics* 46 (2015), 172–185.
[24] Kęstutis Karčiauskas and Jörg Peters. 2019. High quality refinable G-splines for locally quad-dominant meshes with T-gons. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 151–161.

[25] Kęstutis Karčiauskas and Jörg Peters. 2020. Low degree splines for locally quad-dominant meshes. *Computer Aided Geometric Design* 83 (2020), 1–12.
[26] Kęstutis Karčiauskas and Jörg Peters. 2020. Smooth polar caps for locally quad-dominant meshes. *Computer Aided Geometric Design* 81 (06 2020), 1–12. https://doi.org/10.1016/j.cagd.2020.101908 PMC7343232.
[27] Kęstutis Karčiauskas and Jörg Peters. 2021. Least Degree G^1-refinable Multi-sided Surfaces Suitable for Inclusion into C^1 bi-2 Splines. *Computer-Aided Design* 130 (2021), 1–12.
[28] Kęstutis Karčiauskas and Jörg Peters. 2022. Localized remeshing for polyhedral splines. *Computers & Graphics* 106 (2022), 58–65.
[29] Kęstutis Karčiauskas, Jörg Peters, and Ulrich Reif. 2004. Shape characterization of subdivision surfaces–case studies. *Computer Aided Geometric Design* 21 (2004).
[30] Ming-Jun Lai and Larry L. Schumaker. 2007. *Spline Functions on Triangulations.* Cambridge University Press. https://doi.org/10.1017/CBO9780511721588
[31] B Marussig and TJR Hughes. [n.d.]. A Review of Trimming in Isogeometric Analysis: Challenges, Data Exchange and Simulation Aspects. *Arch Comput Methods Eng.* 25, 4 ([n. d.]), 1059–1127. Erratum in: Arch Comput Methods Eng. 2018;25(4):1131.
[32] Ashish Myles and Jörg Peters. 2011. C^2 splines covering polar configurations. *Computer-Aided Design* 43, 11 (2011).
[33] Thien Nguyen, Kęstutis Karčiauskas, and Jörg Peters. 2016. C^1 finite elements on non-tensor-product 2d and 3d manifolds. *Appl. Math. Comput.* 272, 1 (2016), 148–158.
[34] Thien Nguyen and Jörg Peters. 2016. Refinable C^1 spline elements for irregular quad layout. *Computer Aided Geometric Design* 43 (March 29 2016), 123–130.
[35] J. Peters. 1991. Smooth interpolation of a mesh of curves. *Constructive Approximation* 7 (1991), 221–247.
[36] J. Peters. 1995. C^1-surface splines. *SIAM J. Numer. Anal.* 32, 2 (1995), 645–666.
[37] J. Peters. 2019. Splines for Meshes with Irregularities. *The SMAI journal of computational mathematics* S5 (2019), 161–183.
[38] J. Peters and U. Reif. 2008. *Subdivision Surfaces.* Geometry and Computing, Vol. 3. Springer-Verlag, New York. i–204 pages.
[39] J. Peters and X. Wu. 2021. BezierView (short bview): a light weight viewer that renders Bezier patches. https://www.cise.ufl.edu/research/SurfLab/bview
[40] Ulrich Reif. 1998. TURBS—Topologically Unrestricted Rational B-Splines. *Constructive Approximation* 14 (1998), 57–77.
[41] Uwe Schramm and Walter D. Pilkey. 1993. The coupling of geometric descriptions and finite elements using NURBS - A study in shape optimization. *Finite elements in Analysis and Design* 340 (1993), 11–34.
[42] Y.K. Shyy, C. Fleury, and K. Izadpanah. 1988. Shape Optimal Design using higher-order elements. *Computer Methods in Applied Mechanics and Engineering* 71 (1988), 99–116.
[43] Wavefront Technologies. 2021. Wavefront .obj file. https://en.wikipedia.org/wiki/Wavefront_.obj_file
[44] D Toshniwal, H Speleers, R R Hiemstra, and TJR Hughes. 2017. Multi-degree smooth polar splines: A framework for geometric modeling and isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering* 316 (2017), 1005–1061.
[45] Wikipedia contributors. 2021. De Casteljau's algorithm — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=De_Casteljau%27s_algorithm&oldid=1043959654 [Online; accessed 1-December-2021].
[46] Meng Wu, Bernard Mourrain, André Galligo, and Boniface Nkonga. 2017. Hermite Type Spline Spaces over Rectangular Meshes with Complex Topological Structures. *Communications in Computational Physics* 21, 3 (2017), 835–866.