Crossing Knot Lines in Composition of Freeform B-spline Geometry

Boris van Sosin[1], Gershon Elber[1]

[1] Computer Science Department, Technion - Israel Institute of Technology, Haifa, Israel

Abstract

Since the first publications on composition of splines [5, 6], it was clear that the composition, $T(S)$, of a tensor product B-spline function S with another B-spline function T is no longer a tensor product B-spline if S crosses a knot line in the domain of T. The reason can be easily explained for the fact that the new function $T(S)$ has a finite continuity that is governed by the knot line of T that is not, in general, an iso-parametric direction of the resulting composition $T(S)$.

In this work, we propose two approaches to circumvent this long standing difficulty and reconstruct precise representations for $T(S)$ that are either tensor products or trimmed geometry. We focus our discussion on surface-trivariate compositions, $T(S)$, while we present examples and results, for both reconstruction approaches, for microstructure surfaces and trivariates embedded in surface and trivariate deformation functions.

1 Introduction and Previous Work

Function composition is a powerful operation. In geometric modeling, the composition of splines functions, in the Bézier and B-spline bases, was introduced more than two decades ago [5, 6]. DeRose et al. [5] reduced the problem of function composition to Blossoming evaluations. Elber [6] reduced the problem to basic symbolic operations, such as sums and products of splines, and extended the composition operator to both the polynomial Bézier and piecewise polynomial B-spline domains. Lasser [15] provided formulas for computing the control meshes of the composition of Bézier curves into Bézier surfaces and trivariates, and of Bézier surfaces into Bézier trivariates. Since then, efforts have been made to improve the efficiency of the spline composition computation, e.g. [13, 17].

Beyond the obvious use of compositions in spline reparametrizations, in the last couple of decades, new algorithms were also developed for various specific applications, with the aid of the composition operator. For example, using the surface-surface composition for bilinear patches, Feng and Peng [13] showed how to transform a rectangular (tensor product) patch into two triangular patches and how to convert a triangular patch into three rectangular ones, a problem that was also examined by [5]. In [8, 16, 19], compositions were used to convert trimmed surfaces to tensor product surfaces, by dividing the valid domains into quads, and reparametrizing the quads as new tensor product surfaces, only to be composed into the original surface.

The original freeform deformations work [23] was introduced several years prior to [5, 6] and did not consider precise function composition. Instead, mapping of (control) points was simply suggested over trivariate Bézier volumes. Since then, precision in deformations was sought with the aid of compositions. One example of this approach is [11], in which Feng and Peng proposed an efficient way of deforming polygonal models by using B-spline trivariates as mapping functions. The composition method used in [11] works by polynomial interpolation. The process of deforming polygonal models by composition was further improved upon in [10], where polygons which crossed knot lines where subdivided at the knot lines.

Surazhsky and Elber [24] is another example of precise deformation using the composition operator. They employed the curve-surface composition for a precise text deformation of piecewise Bézier outline fonts, where the underlying deformation functions were represented as bivariate B-splines.

Elber [7] used the curve-curve composition to normalize vector fields in general, and to approximate piecewise polynomial arc-length curves in specific. Cohen et al. [4] employed the curve-curve composition for reparametrization towards the elimination of self-intersections in planar ruled surfaces and in metamorphosis between two curves. Kim and Elber [14] developed a precise G^1 surface blending scheme that exploits the curve-surface composition to precisely locate the rail curves of the blending surface over the given input surfaces. In [8], composition of splines were employed to construct blends and establish bounds on (Hausdorff) distances between surfaces.

Since the first publications on composition of splines [5, 6], it was clear that the composition of a tensor product B-spline function S with another B-spline function T, $T(S)$, is no longer a tensor product B-spline, if S crosses a knot line in the domain of T. The reason can be easily explained for the fact that the new function $T(S)$ has a finite continuity that is governed by the knot line [1] of T and is not an iso-parametric direction of the resulting composition $T(S)$, in general.

In this work, we address the question of general composition of B-spline function $T(S)$, i.e. when S is crossing a knot line in the domain of T. The rest of this work is organized as follows. Section 2 considers two approaches to the problem and its solution, by subdividing S along its non-isoparametric directions that are the knot lines of T. Section 3 presents some results, while in Section 4, we conclude and discuss future work.

2 Proposed solution

The following discussion holds for B-spline compositions in any dimensions. However, and as an example, we focus our discussion on surface-trivariate composition. Consider a B-spline surface

$$S(r,t) = (s_x(r,t), s_y(r,t), s_z(r,t)),$$

and B-spline trivariate $T(u,v,w)$. Now consider the composition

$$T(S) = T(s_x, s_y, s_z), \tag{1}$$

and assume S crosses a knot line in the domain of T, D^T, at u_0 (See Figure 1). For now, and for the sake of simplicity, assume u_0 is the only interior knot in T. Let P_{u_0} be the plane induced by knot u_0, in D^T. Clearly $T(S)$ is a B-spline function no longer, as the intersection of S with P_{u_0} introduces a non isoparametric finite continuity (from T) in $T(S)$.

[1]in this work, *knot lines* will denote isoparametric hyperplanes in the domain of T, induced by the knots of T, regardless of the dimension of T.

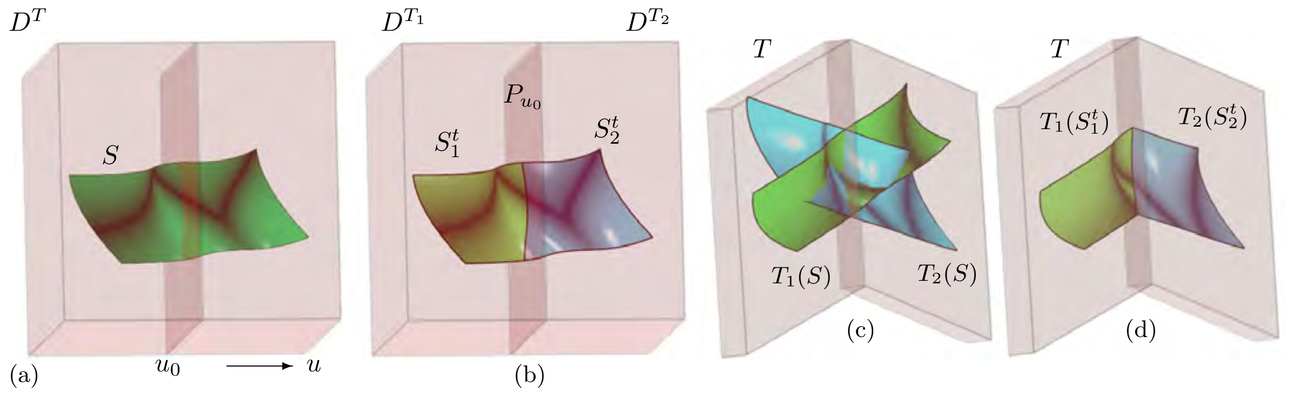


Figure 1: A tensor product surface S is embedded in the domain of tensor product trivariate T, D^T (the transparent box), in (a). S crosses the knot value u_0, in T (represented by the (hyper-)plane $u = u_0$ in the transparent boxes), and hence the composition $T(S)$ is not a tensor product surface. Here, S is subdivided into two trimmed surfaces, S_i^t, $i = 1, 2$, in (b), along the knot plane, P_{u_0}. The full tensor product surface S is composed with the two polynomial trivariates, T_i, $i = 1, 2$, in (c) (the transparent volume T, partitioned into T_1, T_2 at the knot value u_0), only to be trimmed using the trimming information of S_i^t, resulting in (d). See also Figure 2.

We propose two approach to recreate $T(S)$. One by subdividing S to trimmed surfaces that cross no knot lines in D^T, in Section 2.1, and one by subdividing S into tensor product surfaces (that, again, cross no knot lines), in Section 2.2.

2.1 Representing $T(S)$ using trimmed surfaces

Recall the surface-trivariate composition in Equation (1), and denote the two polynomial trivariates of T that are delineated by P_{u_0}, as T_i, $i = 1, 2$, as u_0 is assumed to be the only interior knot in T. Similarly, we subdivide S at P_{u_0} into two trimmed surfaces, $S_i^t \subset T_i$ (see Figure 1). Then, the following observation is now crucial:

Observation 1. *T_i, being a polynomial, has all of $\mathbb{R}^3$ as its domain. However, since T_i is defined as a B-spline patch with no interior knots, its domain is initially limited to a finite box, D^{T_i}. However, T_i can also be considered a Bézier patch, up to an affine transformation of the domain, with all of $\mathbb{R}^3$ as its domain, and with the same control points as the B-spline patch. Then, S, the original input surface, can be composed with the Bézier representation of T_i. Further, for every point $p \in D^{T_i}$ and specifically $p \in S_i^t \subset D^{T_i}$, $T(p) = T_i(p)$. Hence, we can compute $T_i(S)$ and restrict it with the trimming curves of S_i^t, to all points $p \in S_i^t$.*

Algorithm 1 Composition algorithm resulting in trimmed surfaces

```
Input:
$S$: a B-spline surface;
$T$: a multivariate B-spline, such that $S \subset D^T$;

Output:
A set of trimmed surfaces, the union of which forms the composition $T(S)$;

Algorithm:
1: $\boldsymbol{\mathcal{T}} = \{T_i\}$ := subdivide $T$ at all its interior knot values, in all directions, into polynomial multivariate patches;
2: $\boldsymbol{\mathcal{S}} = \{S_i^t\}$ := trim $S$ against all hyperplanes in $D^T$, induced by all interior knot values of $T$, as trimmed surfaces;
3: $\boldsymbol{\mathcal{R}} := \emptyset$; // Initialize the set of solutions.
4: for all $S_i^t \in \boldsymbol{\mathcal{S}}$ do
5:     $T_j$ := the polynomial patch in $\boldsymbol{\mathcal{T}}$ that contains $S_i^t$;
6:     $T_j^e$ := expand the domain of $T_j$ to fully contain the original surface $S$;
7:     $R_i := T_j^e(S)$;
8:     $R_i^t$ := a trimmed surface from the surface $R_i$, and the trimming curves of $S_i^t$;
9:     $\boldsymbol{\mathcal{R}} := \boldsymbol{\mathcal{R}} \cup \{R_i^t\}$.
10: end for
11: return $\boldsymbol{\mathcal{R}}$;
```

The process we perform is described by Algorithm 1. Even though we described the reasoning behind the algorithm for surface-trivariate composition, it is also applicable for composing B-spline surfaces (and even higher dimension multivariates) into any B-spline multivariate that contains it in its domain.

Algorithm 1 has the following properties:

1. In Step 2, if a partitioning of S at a knot hyperplane in D^T is isoparametric in S, the partitioning is done by simple B-spline subdivision, rather than trimming. In the general case, this is a highly improbable event, but in man-made models it can happen quite often. This has the effect of reducing the complexity of the trimmed surfaces produced by Step 2 of the algorithm.
2. The partitioning of S into trimmed surface patches S_i^t (Step 2) is disjoint, and their union covers all of S. Consequently, the set of resulting composed (trimmed) surfaces, $\boldsymbol{\mathcal{R}}$ (see Algorithm 1, Step 3), is also disjoint, and covers the (non tensor product) surface, $T(S)$.

3. Each trimmed surface patch S_i^t is *fully* contained inside the parametric domain of a single polynomial patch D^{T_j}; specifically in Step 5.
4. The resulting set of trimmed surface patches, $\boldsymbol{\mathcal{R}}$, forms a data structure that consists of multiple trimmed surfaces, sharing the same parametric domain, D^T. This data structure can be evaluated at any parameter values u, v of S by finding and evaluating the trimmed surface that contains u, v.
5. The expansion of T_j in Step 6 is only for the purpose of matching the domains. Since T_j is polynomial, it is defined for all of $\mathbb{R}^3$.

Finally, we define an additional property that the representation, which results from Algorithm 1, possesses:

Definition 1. *Let T be a parametric multivariate, and S be another parametric multivariate embedded in T, $S \subset D^T$. Let the composition of S into T be $T(S)$. Then, the composition operation is said to have the property of* preservation of parametrization *if for any values in the parametric domain of S, evaluating $T(S)$ at the parameter values yields the same result as evaluating S and then evaluating T at S.*

The preservation of parametrization can be important for applications in which the composition of a surface into another B-spline function is treated as functional composition.

2.2 Representing $T(S)$ using tensor product surfaces

The algorithm presented in Section 2.1 has a significant drawback that the output contains trimmed surfaces. As an alternative, one can apply an algorithm that converts trimmed surfaces to tensor products (e.g. [16, 19]), only to compose the new tensor product surfaces with T. If the trimmed surfaces, S_i^t, by construction, never cross a knot line of T, the tensor product surfaces, that cover only valid regions in the trimmed surface, can be composed with T with ease, as they will always be contained in a single polynomial patch, T_i.

Algorithms for untrimming, or converting trimmed surfaces into piecewise tensor-product surfaces, typically work on the level of the trimming loops in the parametric space of the trimmed surfaces. Untrimming algorithms of this type first need to tile the interior of the trimming loops with tensor-product patches, and then map the resulting patches back into Euclidean space. Several solutions to the problem of tiling the trimming loops have been proposed in the past. In [16], the parametric domain is partitioned vertically and horizontally at points where the direction of the derivative of the trimming loop is diagonal in the UV-space. In [19], two solutions to the tiling problem are proposed: one by a line-sweep process, and another by a dynamic programming algorithm for optimizing a user-defined function of the properties of the tiles. Another approach for tiling the interior of a planar boundary is based on meshing, and is used extensively in Isogeometric Analysis (IGA), and is potentially applicable to untrimming. Examples of this approach are [20, 25]. After the interior of the trimming loops is tiled with tensor-product surface patches, the surface patches are mapped back to Euclidean space either precisely, by functional composition, as in [19], or by piecewise-polynomial approximation, which can limit the polynomial degrees of the resulting patches, as in [16].

Herein, we use the untrimming approach of [19] to convert trimmed surfaces to tensor product surfaces. Algorithm 2 is similar to the one in the previous section, with the added step of converting the trimmed surface patches into tensor product surface patches. As in the previous algorithm, all the trimmed surface patches generated in Step 2 are fully contained in a single

Algorithm 2 Composition algorithm resulting in tensor product surfaces

Input:
S: a B-spline surface;
T: a multivariate B-spline, such that $S \subset D^T$;

Output:
A set of tensor product surfaces, the union of which forms the composition $T(S)$;

Algorithm:
```
 1: $\boldsymbol{\mathcal{T}} = \{T_i\}$ := subdivide $T$ at all its interior knots in all directions, resulting in polynomial multivariate patches;
 2: $\boldsymbol{\mathcal{S}} = \{S_i^t\}$ := trim $S$ against all hyperplanes in $D^T$ induced by all interior knot values of $T$, as trimmed surfaces;
 3: $\boldsymbol{\mathcal{R}} := \emptyset$; // Initialize the set of solutions.
 4: for all $S_i^t \in \boldsymbol{\mathcal{S}}$ do
 5:   $\boldsymbol{\mathcal{S}}_i^U := untrim(S_i^t)$;
 6:   $T_j$ := the polynomial patch in $\boldsymbol{\mathcal{T}}$ that contains $S_i^t$;
 7:   $T_j^e$ := expand the domain of $T_j$ to fully contain all the surfaces in $\boldsymbol{\mathcal{S}}_U$;
 8:   for all $S_i^u \in \boldsymbol{\mathcal{S}}_i^U$ do
 9:     $R_i := T_j^e(S_i^u)$;
10:     $\boldsymbol{\mathcal{R}} := \boldsymbol{\mathcal{R}} \cup \{R_i\}$;
11:   end for
12: end for
13: return $\boldsymbol{\mathcal{R}}$;
```

polynomial patch of T, T_i. The untrimming algorithm we used produces a precise partitioning of the trimmed surface into tensor product surfaces. After untrimming S_i^t, (Step 5), the resulting set of tensor product surfaces, $\boldsymbol{\mathcal{S}}_i^U$, are no longer intersecting knot lines of T (up a certain tolerance).

The presented approach requires a careful choice of the algorithm which partitions S into trimmed surface patches, and the algorithm which converts the trimmed surface patches back into tensor product surface patches (recall the above discussion on the various approaches to untrimming in [19, 16, 25, 20]). For once, it is preferable to choose a partitioning into trimmed surface patches, and an untrimming algorithm which, together, result in a relatively small number of tensor product surface patches, and do not elevate their degree too much.

Another problem which can arise in implementing Algorithms 1 or 2 is that the tolerance of the trimming curves that result from intersecting S with the knot lines of T (Step 2) is determined by the precision of surface-hyperplane intersection computation.

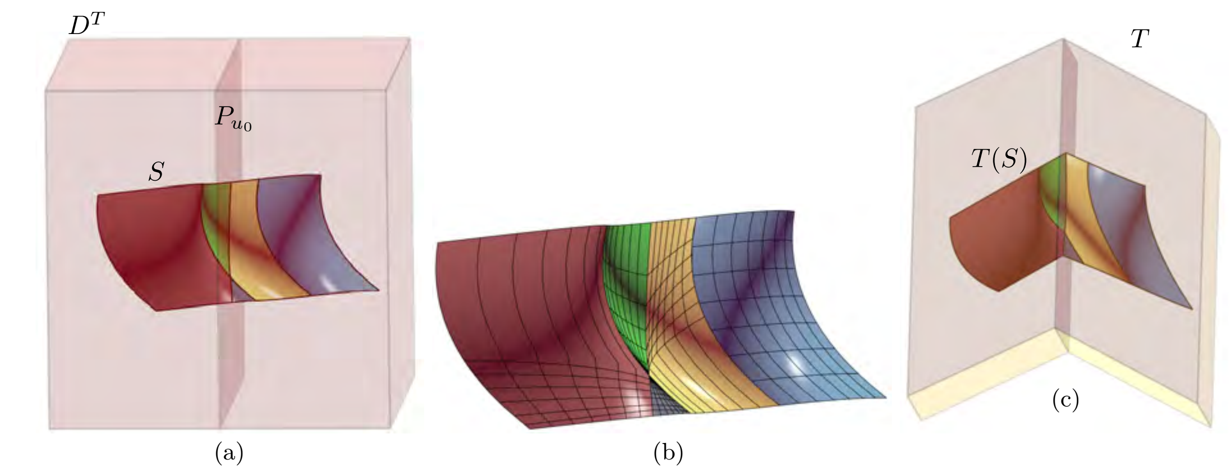


Figure 2: A tensor product surface S is embedded in the domain of tensor product trivariate T, D^T. S is subdivided into two trimmed surfaces, as in Figure 1(b), that are again subdivided into the five tensor product surfaces shown in (a). The five tensor product surfaces (b), some of them singular, due to the untrimming algorithm, are each residing in a single polynomial domain of T and hence can be composed with no knot line crossing, resulting in (c). See also Figure 1.

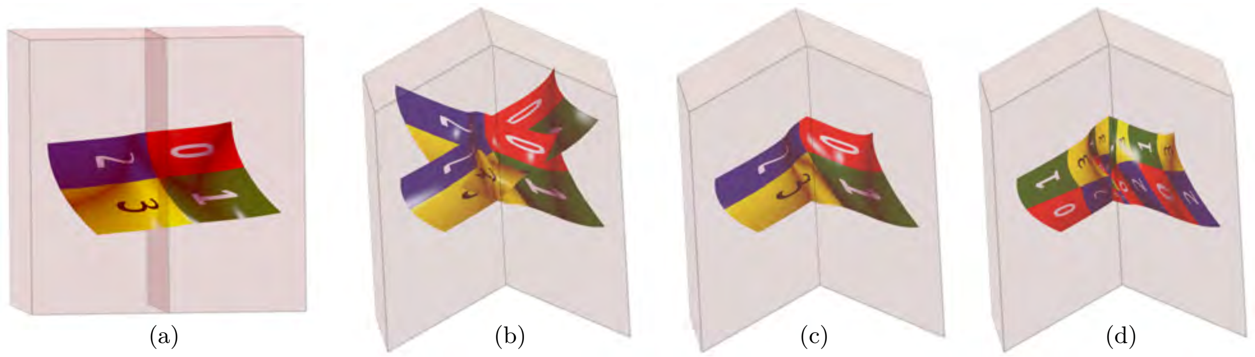


Figure 3: A numbered checkerboard texture applied to the example from Figures 1 and 2. (a) The texture applied to the input surface, S. (b) S composed with the two polynomial trivariates $T_i, i = 1, 2$, as in Figure 1. The texture is deformed, along with S itself, in accordance with the mapping functions T_i. (c) In the resulting trimmed surfaces R_i^t, the texture is trimmed, along with the surface itself, yielding a correctly stitched texture in the output model. (d) In contrast, the tensor-product surfaces, $\boldsymbol{\mathcal{S}}_i^U$, which result from the untrimming step (Step 5 in Algorithm 2), get an entirely new parametrization, which does not preserve the texture. This becomes clearly evident by counting the instances of each number in the resulting textures in (c) and (d).

As a result, the trimmed surfaces, S_i^t (and therefore the untrimmed surfaces, S_i^u), can extend outside of the domain D^{T_j}, the tolerance amount. Therefore, and in order to prevent the inaccuracy from aggregating, we expand T_j (Step 7 in Algorithm 2) in both Algorithms.

Additionally, the untrimming algorithm we used can potentially result in triangular, and hence singular at one corner, rather than quadrilateral tensor product surface patches [19] (follow the isoparametric lines in Figure 2(b)). The functional composition operation (Step 9) is not affected by such singularities that occur in the corners of triangular patches. However, it is possible to use Algorithm 2 with a different untrimming approach which results only in regular quadrilateral patches, if the application requires it.

The algorithm presented in this section, only yields tensor product surfaces in the output. However and unlike the algorithm from Section 2.1, herein, the original parametrization (i.e. Definition 1) is lost.

2.3 Comparison

Although the two algorithms may appear similar, and have the first two steps in common, they differ in the properties of their output, and their potential applications. Recall the property of preservation of parametrization from Definition 1: any point, $(u, v) \in D^S$, is mapped to the same point in Euclidean space by either applying $T(S(u, v))$ directly, or by applying $S(u, v)$ and then applying T on the result. One application in which this property is important, is texture mapping. Texture mapping is a mapping from UV coordinates on the surface of a geometric model to an image (texture). The texture, in turn, governs properties such as color and shading. For volumetric models, texture mapping (generalized to mapping UVW coordinates to a 3D map) can be used to control the properties of the interior of the model, for example in multi-material additive manufacturing (e.g. [2, 21, 22, 9]). Since the mapping of the texture onto the geometry (surface or trivariate) depends on the parametrization, it requires the preservation of the parametrization, when composing B-spline models. This is demonstrated in Figure 3.

In contrast, other applications do not necessitate the preservation of the parametrization property, but require the geometry to be tensor-product. Operations such as integration, and computation of surface area and enclosed volume are very easy to perform on tensor-product geometry, compared to trimmed geometry. These operations are crucial for IGA, for example.

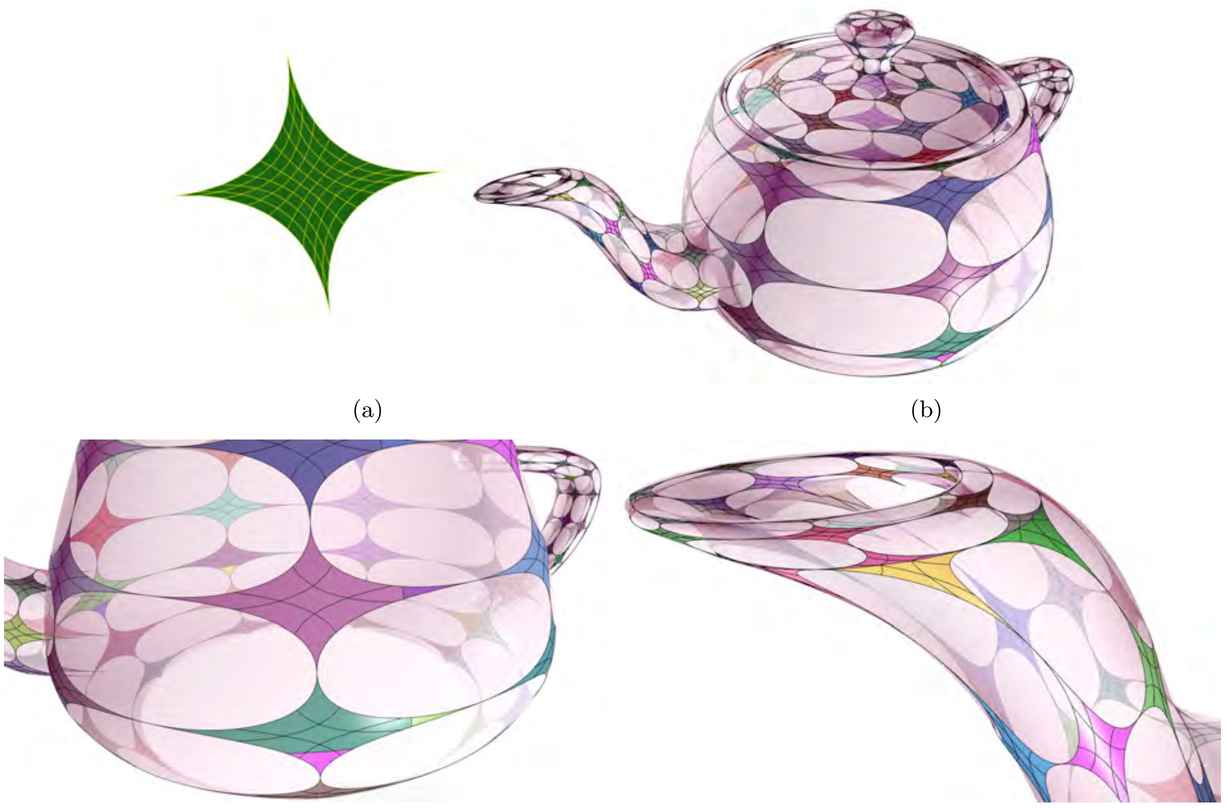

(a) (b) (c) (d)

Figure 4: The covering of the Utah teapot with a star pattern using surface-surface composition which crosses knot lines using Algorithm 1. The resulting trimmed surfaces are randomly colored, to show the partitioning of the stars due to the knot lines of the mapping functions, the surfaces of the teapot in this example. (a) The star-shaped bi-quadratic (orders $(3,3)$) Bézier surface which was used for the pattern. (b) The teapot (in transparent color) covered in a (7×7) pattern of stars using Algorithm 1. The teapot consists of four B-spline surfaces, all bi-cubic (orders $(4,4)$). In (c), the bottom part of the body of the teapot, and in (d), the spout, are shown from different angles, showing the partitioning of the star surfaces at the interior knots of the teapot's surfaces. Compare with Figure 5.

3 Examples

Our first example is a surface-surface composition problem, in which we use Algorithm 1 to compose a pattern of star-shaped surfaces (see Figure 4 (a)) into the four surfaces of the Utah teapot: the body of the teapot, the spout, the lid, and the handle (see Figure 4(b)). The star shape is a bi-quadratic Bézier surface, and the parts of the teapot are all bi-cubic B-spline surfaces, all with multiple interior knots in both directions. For this example, the star shape was replicated to form a (7×7) grid, for each part of the teapot, to make sure that the star surfaces cross knot lines of the surfaces of the teapot transversely. Note the trimming induced by the knot lines are magnified in Figure 4(c) and 4(d). The resulting composed trimmed surface patches are of polynomial orders $(13, 13)$.

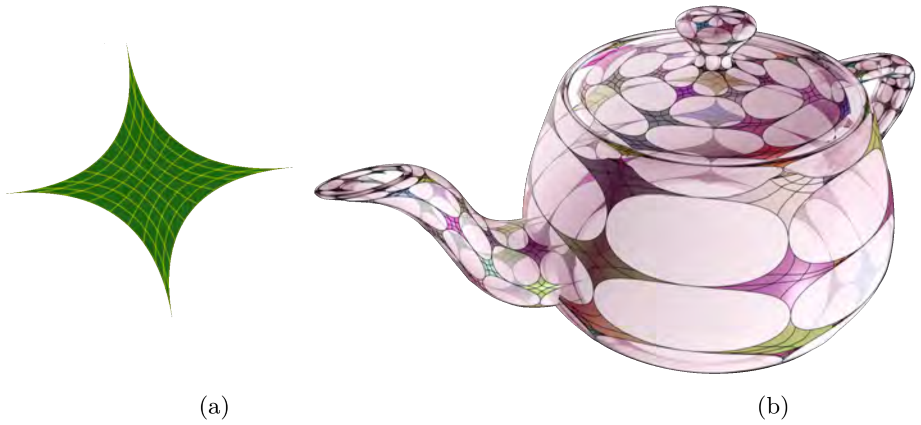
For the second example, in Figure 5, we compose the same pattern of stars into the Utah teapot, using Algorithm 2. As expected, the resulting covering of the teapot is visually similar to the example in Figure 4. However, in this example, the composed surface patches are all tensor product. The untrimming algorithm we used for our implementation further elevates the degrees of the surface patches, and therefore, the orders of the composed tensor product patches are $(25, 25)$.

The next two examples we present are surface-trivariate composition problems. First, in Figure 6, we composed a pattern of cross-shaped structures into a trivariate duck model. The duck model, in Figure 6(a), consists of a single B-spline trivariate of polynomial orders $(3, 3, 4)$, with $(9, 9, 14)$ knot intervals in the u, v, w directions, respectively. The cross structure, shown in Figure 6(b), consists of 24 Bézier surfaces of polynomial orders $(2, 3)$, and we composed a pattern of $(3 \times 3 \times 23)$ crosses into the parametric domain of the duck model. The orders of the composed patches are $(8, 25)$.

Next, in Figure 7, we compose a structure consisting of three pipes that curve around each other, each of them is a bi-cubic B-spline surface (Figure 7(b)). Our mapping trivariate, in Figure 7(a), is a twisting torus-like shape, with a rectangle cross-section, of polynomial orders $(2, 2, 3)$, and eight polynomial intervals in the circumference direction. We used a pattern of $(4 \times 4 \times 23)$ pipe structures to tile the parametric domain of the torus. The orders of the composed patches are $(13, 13)$.

In the next example, in Figure 8, we composed a B-rep model of a bushing with holes into a trivariate. The bushing model consists of 12 trimmed B-spline surfaces, of polynomial orders $(2, 3)$. The mapping trivariate is of polynomial orders $(2, 3, 3)$ in the u, v, w directions, respectively, with a single interior knot, u_0, in the u direction. Composing a model consisting of trimmed surfaces works similarly to composing tensor product surfaces, except the partitioning of the input surfaces into trimmed surface patches (Step 2 in both Algorithms 1 and 2) now requires intersecting trimmed surfaces with (hyper-) planes (i.e. by Boolean operations on B-rep models, in our example). Otherwise, the two algorithms remain unchanged. The composed trimmed surface patches are of orders $(11, 6)$.

Finally, we experimented with composing trivariate models into other trivariate models. Trivariate-trivariate composition is conceptually similar to composing surfaces, except that intersecting a trivariate with the knot lines of the T trivariate results in trimmed trivariates. Trimmed trivariates, in a fashion similar to trimmed surfaces, consist of a tensor product trivariate and a hierarchy of trimming surfaces, which themselves can be trimmed surfaces (trimmed by curves, that is). After the trivariate S is



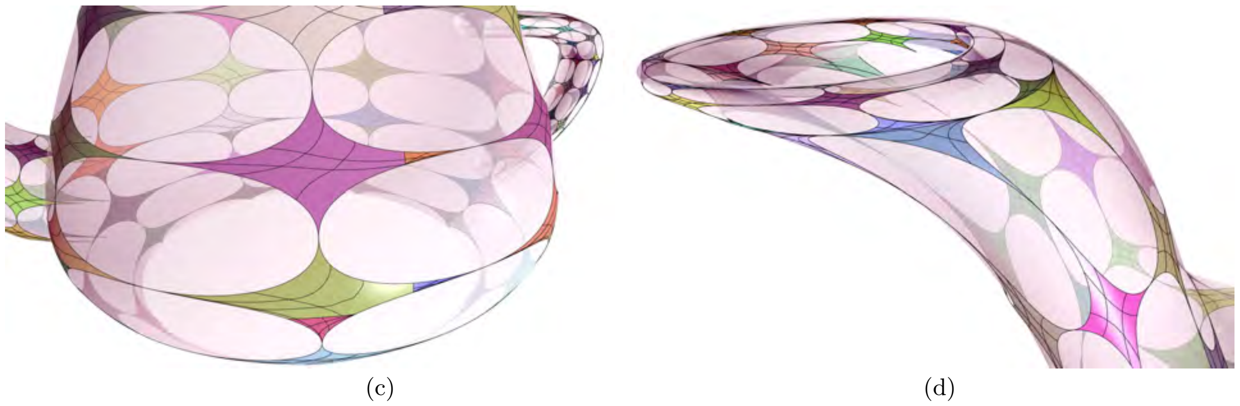
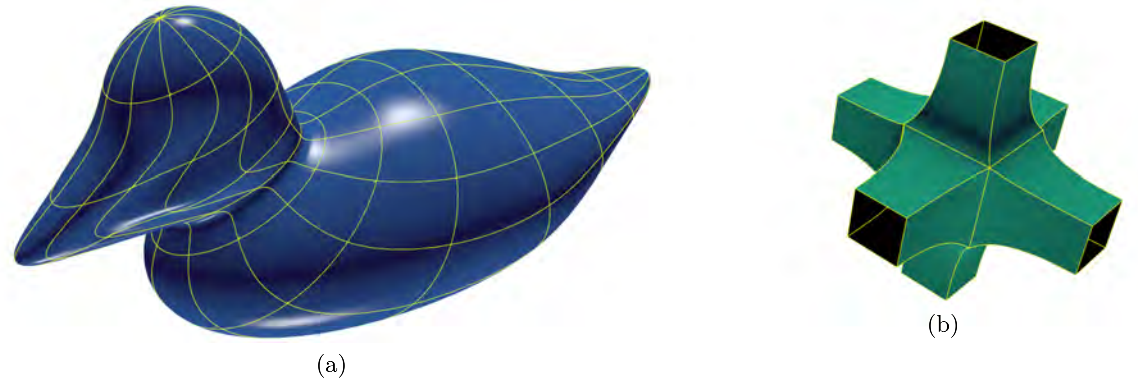


Figure 5: The covering of the Utah teapot with a star pattern using Algorithm 2. The resulting surfaces are randomly colored, to show both the partitioning of the stars due to the knot lines of the mapping functions, and the new parametrization created by the untrimming process. (a) The teapot (in transparent color) covered in the (7×7) pattern of stars using Algorithm 2. In (c), the bottom part of the body of the teapot, and in (d), the spout, are shown from different angles, showing the tensor product parametrization of the resulting surfaces. Compare with Figure 4.



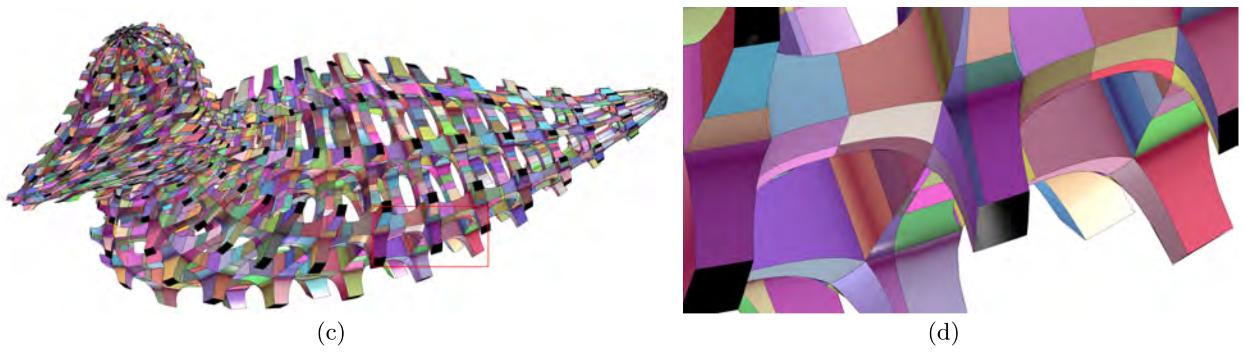


Figure 6: The duck model tiled with a $(3 \times 3 \times 23)$ pattern of cross-shaped structures. (a) The original duck model, a B-spline trivariate of orders $(3, 3, 4)$. (b) A single cross-shaped tile, constructed from 24 Bézier surfaces of orders $(2, 3)$. (c) The composition results using Algorithm 1, with the composed trimmed surfaces (of orders $(8, 25)$) in random colors (back-faces in black). (d) Part of the composition result enlarged, showing the non-isoparametric partitioning of the surfaces of the cross structure (see marked area in (c)).

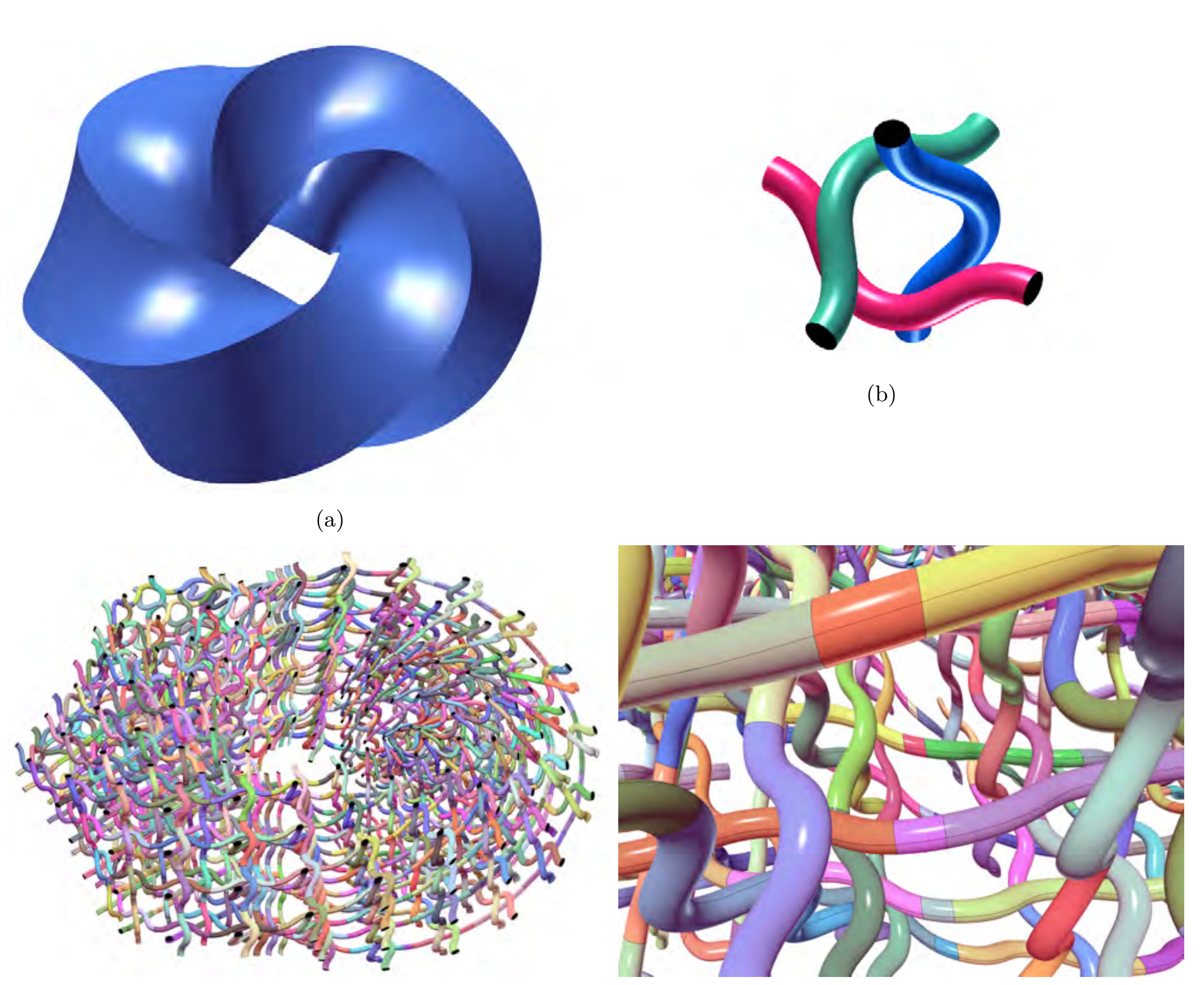
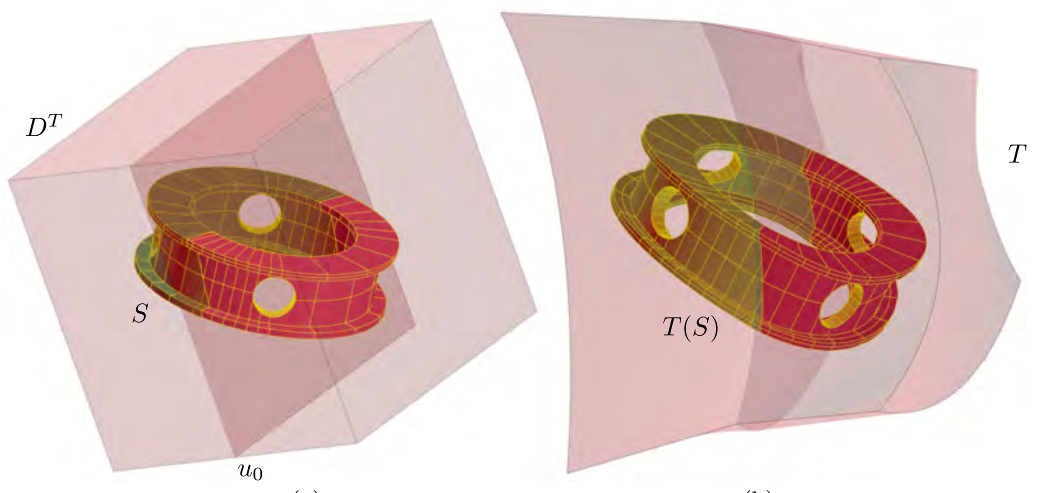

(a) (b) (c) (d)

Figure 7: The twisted torus model tiled with a $(4 \times 4 \times 23)$ pattern of three-pipes structures. (a) The twisted torus trivariate model (B-spline of orders $(2, 2, 3)$). (b) A single three-pipes tile, each pipe is a B-spline surface of orders $(4, 4)$. (c) The composition results, with the composed trimmed surfaces (B-spline of order $(13, 13)$) in random colors (back-faces in black). (d) Part of the composition result enlarged, again, showing the non-isoparametric partitioning of the surfaces of the pipes structure.



(a) (b)

Figure 8: The composition of the bushing model (12 trimmed B-spline surfaces of orders $(2, 3)$) into a trivariate of polynomial orders $(2, 3, 3)$, with a single interior knot u_0. (a) The bushing model in D^T, already partitioned at the knot line u_0. (b) The bushing model composed into the trivariate (trimmed surfaces of orders $(11, 6)$).

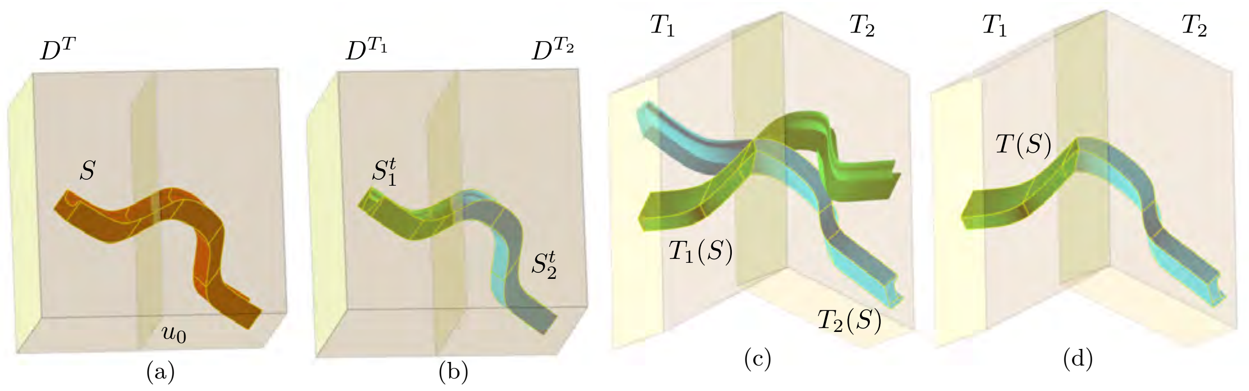


Figure 9: The composition of a trivariate bridge model (S) into another trivariate (T). (a) the bridge, inside D^T, with the single interior knot u_0 of T. (b) the partitioning of S into two trimmed trivariates: S^t_1, S^t_2. (c) S composed into T_1 and T_2. (d) the final composition result, after the application of the trimming information of S^t_i into $T_i(S)$, consisting of two trimmed trivariates.

partitioned into trimmed trivariates at the knot lines of T, their tensor product trivariate can be composed into the (expanded) Bézier patch extracted from T (T^e_j in Algorithms 1,2), and the trimming data is propagated back to the composition results.

In Figure 9, we present an example of trivariate-trivariate composition. For this purpose, we used the V-Model framework, of [18], to compute the intersection of a trivariate bridge model with a single knot line of another trivariate model, resulting in two trimmed trivariates, S^t_1, S^t_2. Then, we computed the composition of S into the two expanded Bézier patches T^e_1, T^e_2, and added back the trimming data of S^t_1, S^t_2 to each of the composed trivariates. To the best of our knowledge, untrimming algorithms do not currently support converting trimmed trivariates into tensor product trivariates, so for the problem of trivariate-trivariate composition, only Algorithm 1 is applicable.

4 Conclusion and future work

In this paper, we have presented two solutions to the problem of performing composition of B-spline geometry, $T(S)$, for cases where S crosses knot lines of T. Our first solution produces trimmed geometry, and preserves the parametrization of the input surface (trivariate), while our second solution (which is currently not applicable to trivariates) results in tensor product surfaces, but with the loss of the original parametrization. We have tested and demonstrated our algorithm on surface-surface, surface-trivariate, trimmed surface-trivariate, and trivariate-trivariate composition problems. Our algorithm is also capable of composing complex models and large patterns of tiles, such as microstructures.

4.1 Future work

The partitioning of S into trimmed surface patches is a surface-hyperplane intersection problem, which, in most implementations, yields a piecewise-linear trimming curves. These trimming curves, are used as input for the untrimming step (Step 5 in Algorithm 2), and, again, depending on the untrimming algorithm used, can result in untrimmed tensor product surface patches with a large number of control points. Consequently, applying composition operation $T^e_j(S^u_i)$ (in Step 9 of Algorithm 2) might be computationally expensive. Hence, non-linear approximations of the trimming curves, which reduce their size, might be beneficial for making Algorithm 2 more practical for high-complexity problems, such as our examples in Figures 6 and 7.

Recent advances in IGA have made operations on B-spline geometry relevant in the context of analysis. The composition operator is a powerful tool for creating complex geometry, and the proposed algorithms can increase its applicability for IGA, among other fields. Cohen et. al. [3] proposed a method for evaluating the quality of geometric models, with respect to its suitability to IGA. This can by applied to composed geometric structures, such as ones which result from our algorithms, to evaluate their quality, and find ways of improving it. This is particularly relevant to Algorithm 2, in which the choice of untrimming algorithm is a degree of freedom which can affect the parametrization of the resulting geometry. Furthermore, Haberleitner and Jüttler [12] proposed a method for computing cutting surfaces for partitioning a B-spline B-rep model into IGA-suitable components. This, in combination with our algorithms, can be used, for example, to apply IGA methods to geometry which consists of microstructures. Alternatively to this approach, Al Akhras et. al. [1] proposed a method for constructing IGA-suitable volumetric B-spline geometry from B-rep models (such as our example in Figure 8).

Finally, functional composition of (piecewise) polynomial functions is useful for general algebraic computations, and not only geometric modeling. Such computations, typically, require the preservation of parametrization property (Definition 1) of Algorithm 1. An important step towards using our proposed algorithm for general algebraic computations, is developing a data structure which allows efficient evaluation of a set of a set of disjoint trimmed B-spline functions, such as $\boldsymbol{\mathcal{R}}$, in our algorithms.

Acknowledgment

This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA), under contract HR0011-17-2-0028. The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

References

[1] H. Al Akhras, T. Elguedj, A. Gravouil, and M. Rochette. Isogeometric analysis-suitable trivariate nurbs models from standard b-rep models. *Computer Methods in Applied Mechanics and Engineering*, 307:256–274, 2016.

[2] A. Biswas, V. Shapiro, and I. Tsukanov. Heterogeneous material modeling with distance fields. *Computer Aided Geometric Design*, 21(3):215–242, 2004.

[3] E. Cohen, T. Martin, R. Kirby, T. Lyche, and R. Riesenfeld. Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199(5):334–356, 2010.

[4] S. Cohen, G. Elber, and R. Bar-Yehuda. Matching of freeform curves. *Computer-Aided Design*, 29(5):369 – 378, 1997.

[5] T. D. DeRose, R. N. Goldman, H. Hagen, and S. Mann. Functional composition algorithms via blossoming. *ACM Trans. Graph.*, 12(2):113–135, Apr. 1993.

[6] G. Elber. *Free form surface analysis using a hybrid of symbolic and numeric computation.* PhD thesis, CS, The University of Utah, 1992.

[7] G. Elber. Symbolic and numeric computation in curve interrogation. *Computer Graphics Forum*, 14(1):25–34, 1995.

[8] G. Elber and M.-S. Kim. Technical note: Modeling by composition. *Comput. Aided Des.*, 46:200–204, Jan. 2014.

[9] B. Ezair, D. Dikovsky, and G. Elber. Fabricating functionally graded material objects using trimmed trivariate volumetric representations. In *Proceedings of SMI'2017 Fabrication and Sculpting Event (FASE), Berkeley, CA, USA*, June 2017.

[10] J. Feng, T. Nishita, X. Jin, and Q. Peng. B-spline free-form deformation of polygonal object as trimmed bézier surfaces. *The Visual Computer*, 18(8):493–510, Dec 2002.

[11] J. Feng and Q. Peng. B-spline free-form deformation of polygonal objects through fast functional composition. In *Geometric Modeling and Processing 2000. Theory and Applications. Proceedings*, pages 408–414. IEEE, 2000.

[12] M. Haberleitner and B. Jüttler. Isogeometric segmentation: Construction of cutting surfaces. *Computer-Aided Design*, 2017.

[13] F. Jieqing and P. Qunsheng. Functional compositions via shifting operators for bézier patches and their applications, 1999.

[14] K. Kim and G. Elber. A symbolic approach to freeform surface blends. *THE J. OF VISUALIZATION AND COMPUTER ANIMATION*, 8(2):69–80, 1997.

[15] D. Lasser. Composition of tensor product bézier representations. In *Geometric modelling*, pages 155–172. Springer, 1993.

[16] X. Li and F. Chen. Exact and approximate representations of trimmed surfaces with nurbs and bezier surfaces. In *2009 11th IEEE International Conference on Computer-Aided Design and Computer Graphics*, pages 286–291, Aug 2009.

[17] W. Liu and S. Mann. An optimal algorithm for expanding the composition of polynomials. *ACM Trans. Graph.*, 16(2):155–178, Apr. 1997.

[18] F. Massarwi and G. Elber. A b-spline based framework for volumetric object modeling. *Computer-Aided Design*, 78:36–47, 2016.

[19] F. Massarwi, B. van Sosin, and G. Elber. Untrimming: Precise conversion of trimmed-surfaces to tensor-product surfaces. *Computers & Graphics*, 70:80–91, 2017.

[20] X. Nian and F. Chen. Planar domain parameterization for isogeometric analysis based on teichmüller mapping. *Computer Methods in Applied Mechanics and Engineering*, 311:41–55, 2016.

[21] X. Qian and D. Dutta. Feature-based design for heterogeneous objects. *Computer-Aided Design*, 36(12):1263–1278, 2004.

[22] Y. Sasaki, M. Takezawa, S. Kim, H. Kawaharada, and T. Maekawa. Adaptive direct slicing of volumetric attribute data represented by trivariate b-spline functions. *The International Journal of Advanced Manufacturing Technology*, 91(5-8):1791–1807, 2017.

[23] T. W. Sederberg and S. R. Parry. Free-form deformation of solid geometric models. In *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '86, pages 151–160, New York, NY, USA, 1986. ACM.

[24] T. Surazhsky and G. Elber. Artistic surface rendering using layout of text. *Computer Graphics Forum*, 21(2):99–110, 2002.

[25] G. Xu, M. Li, B. Mourrain, T. Rabczuk, J. Xu, and S. P. Bordas. Constructing iga-suitable planar parameterization from complex cad boundary by domain partition and global/local optimization. *Computer Methods in Applied Mechanics and Engineering*, 328:175–200, 2018.