Parameter Aligned Trimmed Surfaces

Shannon Halbert*
University of Calgary

Faramarz Samavati†
University of Calgary

Adam Runions‡
Max Planck Institute for Plant Breeding Research
University of Calgary

ABSTRACT

We present a new representation for trimmed parametric surfaces. Given a set of trimming curves in the parametric domain of a surface, our method locally reparametrizes the parameter space to permit accurate representation of these features without partitioning the surface into subsurfaces. Instead, the parameter space is segmented into subspaces containing the trimming curves, the boundaries of which are aligned to the local parameter axes. When multiple trimming curves are present, intersecting subspaces are further segmented using local Voronoï curve diagrams which allows the subspace to be distributed equally between the trimming curves. Transition patches are then used to reparametrize the areas around the trimming curves to accommodate the trimmed edges. This allows for high quality interpolation of the trimmed edges while still allowing parametric referencing and trimmed surface sampling.

Index Terms: I.3.5 [COMPUTER GRAPHICS]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations;

1 INTRODUCTION

When choosing a representation for a surface, one has many options. Commonly employed representations include polygonal meshes, as well as subdivision, implicit, and parametric surfaces. The choice of representation is often dictated by the advantages offered by each particular representation. Parametric surfaces are often employed because of their efficient surface point referencing and rendering capabilities. NURBS surfaces in particular are the standard surface representation for CAD/CAM modeling systems due to their geometric properties and potential for flexible editing through control point manipulation. However, introducing trimmed holes into a parametric surface is much less straightforward than doing so for other surface representations, as holes cannot be represented using a single continuous parametric domain. Consequently, the most common way to trim these surfaces (see [4]) is to convert them to a surface representation that supports holes. Once the surface has been converted, the advantages of initially choosing a parametric representation are lost. The guaranteed smoothness, simple and efficient surface point sampling, and simple editing granted by the parametric representation are lost. These properties are the result of having direct access to a parametric function defined over a simple parametric domain (see [2]). In practice, some editing capabilities can be recovered with the use of complex data structures (see [3]); however, editing the trimmed surface typically either results in large deviations from the original surface or necessitates returning to the untrimmed surface which must then be re-trimmed and retessellated. T-splines [16] are a promising alternative for commercial software, but require a parametric surface to have a NURBS representation. This is not the case when modeling structures, such as trees, that can be produced using parametric primitives such as cylinders.

An alternative approach is to split the surface into separate parametric surfaces (as is done in [7, 8]). This allows the newly trimmed surface to remain parametric and be edited via manipulation of the original surface. Although the individual surfaces are parametric and can be defined in the original parametric domain, they each have an independent local parametric subdomain as well. Finding a correspondence between these local parametrizations and the original global parametric domain, commonly represented by the parameters u and v, is non-trivial. Additionally, dividing the original parametric domain, a simple rectangular region, into a set of arbitrary surfaces destroys one of the advantages of parametric surfaces: the simple u, v traversal of the domain which aids the fast lookup of points and rendering capabilities of these surfaces.

In this paper we introduce a new method for representing trimmed parametric surfaces which allows us to preserve the parametric properties of the original surface. The method consists of three stages: isolation of trimming curves, Voronoï diagram creation, and reparametrization. The surface can then be rendered using a two-stage rendering process.

For a given set of trimming curves (Figure 1a), the first stage of trimming, shown in Figure 1b, localizes the influence of the trimming curves on the parameter space by creating polygonal subspaces referred to as *axis-aligned trimming regions* (AATRs) around groups of trimming curves. The boundaries of these subspaces are aligned with the parameter axes for easy identification. During the second stage, shown in Figure 1c, local Voronoï diagrams are calculated for intersecting AATRs. Bisectors are found and their curves smoothed and uniformly sampled to equally share the subspace between adjacent curves. In the third stage, the untrimmed regions of the AATRs are reparametrized using *transition patches* to interpolate both the trimmed edge and the Voronoï cell boundary. To do this, we identify points of interest (points that must be interpolated when rendering) along both the trimming curve and its boundary curve. Corresponding points are found and connected on the boundary or curve respectively to define the inner boundaries of the transition patches. Finally, the patch spaces are reparametrized using Coons patches.

The surface is rendered in two stages: during the first stage the subspace outside of the AATRs is rendered, followed by the transition patches. This allows sample points to be matched along shared boundaries.

Since all subregions are created in the parametric domain, it is easy to ensure no gaps are introduced between adjacent patches. The trimmed surface can be edited via the original surface points in an intuitive manner, as with any other parametric surface, without requiring retessellation. This results in real-time editing capabilities without requiring the implementation of more complex surfaces.

2 RELATED WORK

We provide a brief overview of existing trimming techniques for parametric surfaces. Parametric trimming methods typically employ one of the two following strategies: direct triangulation, or segmenting the trimmed surface into multiple patches or surfaces.

*e-mail: shalbert@ucalgary.ca
†e-mail:samavati@ucalgary.ca
‡e-mail:runions@mpipz.mpg.de

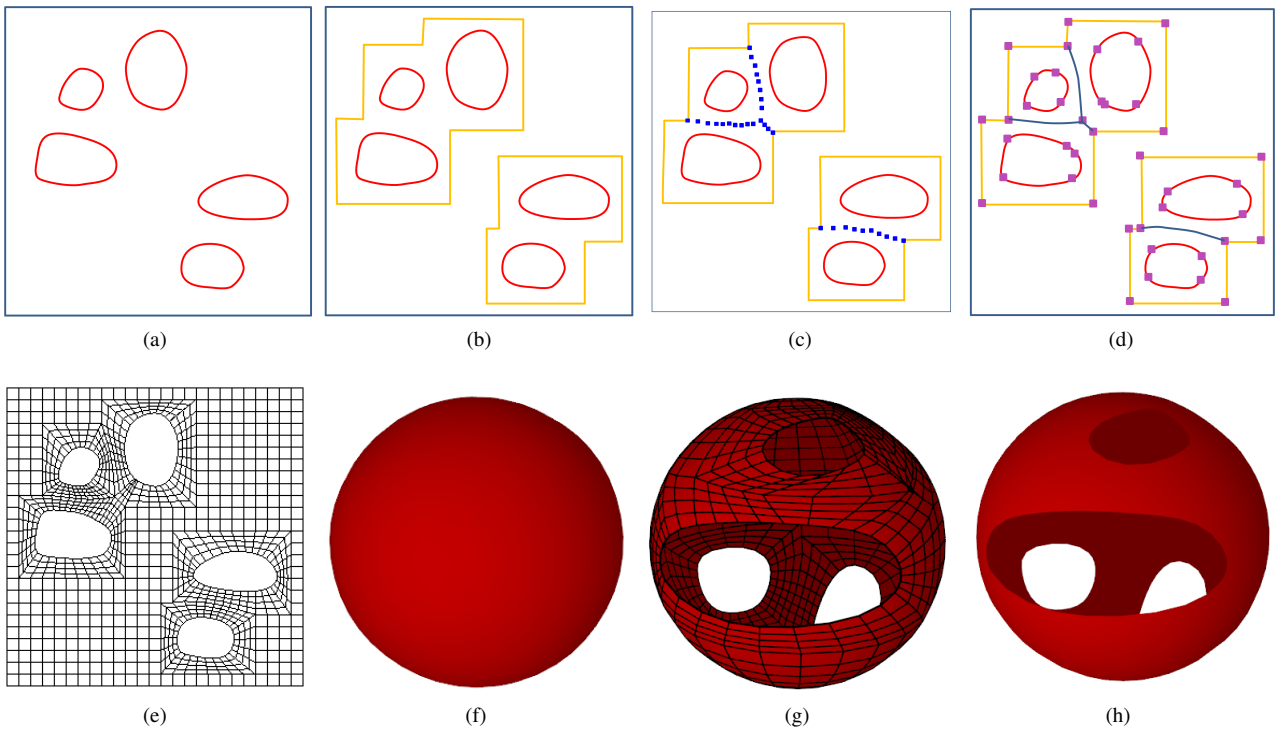


Figure 1: Overview of trimming process. (a) Untrimmed parametric domain; (b) Axis-aligned trimming regions; (c) Local Voronoï diagrams; (d) Coons patch boundary identification; (e) Final tessellated parametric domain; (f) Untrimmed sphere; (g,h) Final model.

Rockwood et al. [13] first triangulated trimmed parametric surfaces by converting them into individual Bézier patches bounded by the trimming curves. The valid region (i.e. the portions remaining after trimming) of each patch is further subdivided into *uv*-monotone regions which are triangulated independently. As points along region boundaries are not guaranteed to align, gaps can occur between adjacent patches. These small gaps or *cracks* are a common problem when trimming parametric surfaces. Sheng and Hirsh [17] instead convert the surface into a mesh. A minimum 2D triangle edge length is chosen such that the mapped 3D triangle does not deviate from the surface by more than a user specified tolerance. The parameter space of the surface is first split into polygonal subregions based on the boundaries of the surface's Bézier patches which also make them axis-aligned. Delaunay triangulation [5] is then used to triangulate the subregions with constraints to ensure vertex matching across neighboring subregions. Utilizing the subregions localizes extraordinary triangles. Piegl and Richard [11] expand this technique to NURBS, triangulating the entire space at once using digital terrain modeling techniques [6]. The scanline algorithm they employed to tessellate the trimmed surface produces a more regular mesh than [17]; however, both methods produce a mesh rather than a parametric surface.

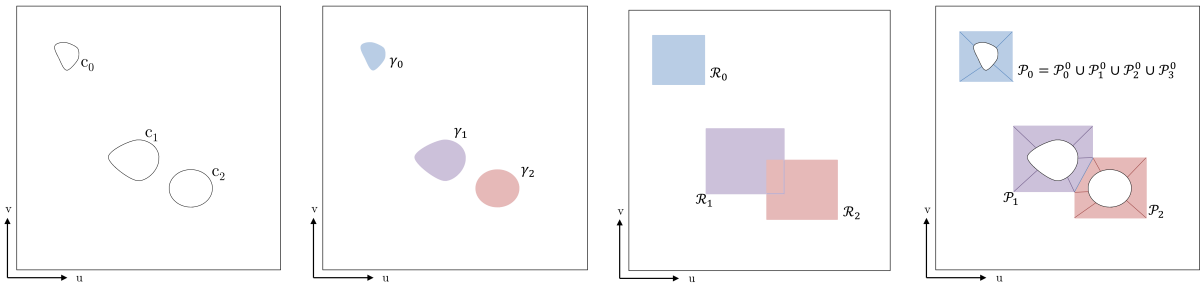
Kumar and Manocha [10] divide the parametric domain into uniform rectangular cells with an edge length determined similarly to [17]. Trimming curves are approximated by piece-wise linear segments. Each grid cell is subsequently labeled as being inside, outside or intersecting with the valid part of the surface. Cells within the valid part of the surface are then triangulated with special consideration for intersecting cells. This method provides improvements in speed and efficiency over previous methods. It also produces crack-free surfaces. However, the resulting surface representation is still a general mesh and there is a tendency towards irregular vertices in triangles along the trimmed boundaries. Piegl and Tiller [12] use a more complex labeling system than [10] but instead of a regular grid, the parametric domain of the surface is subdivided based on surface geometry, allowing them to guarantee a minimum overall deviation between the mesh and surface. However, since every cell must be checked for intersections with each trimming curve, both of these methods tend to be inefficient.

Hamann and Tsai [7] decompose a trimmed NURBS surface into a set of ruled surfaces in the parametric domain. A Voronoï diagram is used to segment the space evenly between the trimming curves. A scanline algorithm is then used to create the boundaries of the ruled surfaces used to interpolate the trimming curves. Although the surface remains parametric, the ruled surfaces can be long and irregular in shape creating ill-shaped surface meshes. Also, changes to the surface are global, as one trimming curve affects the entire surface.

Hui and Wu [8] take decomposition of the surface one step further and decompose the trimmed surface into separate NURBS surfaces based on Hamann and Tsai's Voronoï decomposition. They do this by detecting features along the trimming curves. These points are then used to segment the cells, creating the boundaries of the new surfaces. However, the resulting surfaces are independent from one another. As a result, extra work must be done to ensure continuity when editing these surfaces. The subsurface boundaries also do not respect the original parameter axes making determination of the subregion which contains a given point in the original parametric domain more computationally complex.

Other methods such as [4] and [15] build on these fundamental techniques through algorithmic improvements or by increasing efficiency through the use of GPU programming.

Notably, T-splines [16] have been successfully used to produce watertight trimmed models. By allowing each control point to have



(a) Trimming curves. (b) Curve spaces. (c) Axis-aligned trimming regions. (d) Transition patches.

Figure 2: Definition of the subspaces and piecewise definition of the parameter space. Given our parameter space A and trimming curves $C = \{c_0, c_1, c_2\}$ in (a), we wish to remove the subspaces (b) enclosed within the trimming curves from A. As this cannot be done directly, we instead define a region around each curve (c) we wish to reparametrize to accommodate its associated curve. In (d), Voronoï diagrams are used to segment overlapping subspaces and transition patches are created to reparametrize A to accommodate the curves. The final trimming subspace is defined as $A^T = A \setminus R^* \cup P^*$.

its own knot vectors and blending functions, they control the connectivity of each point. Compared to NURBS, this relaxes the requirement for a regular grid of control points, allowing for the introduction of T-junctions along trimmed boundaries and eliminating the dreaded cracks between intersecting surfaces. However, this method only works with NURBS surfaces and must be used in conjunction with a T-grid to store the extra information and allow for the specialized structure of the final model. It forces the conversion of a simple parametric surfaces into a more complex surface. In contrast, our parameter-aligned strategy works for general parametric surfaces and relaxes the need for T-junctions.

In this paper, similar to the Hamann and Tsai and Hui and Wu works, we employ general Voronoï diagrams. This refers to the generalization of Voronoï diagrams to construct sites for point sets, line segments, polygons, circles and general curves. Alt and Schwarzkopf [1] first introduced the idea of Voronoï diagrams for general curved objects. In this paper, it is this definition we use when referring to Voronoï diagrams.

Hui and Wu's method successfully preserves the parametric properties of the surface but at the cost of the global parametrization and straightforward editing. Rather than segment the surface into independent surfaces, we redefine the parameter space to accommodate and accurately represent the trimmed edges. The result is a trimmed surface that has the main properties of the original parametric surface including editing capabilities.

3 Trimmed Parameter Space Representation and Method Overview

Given a parametric surface S which is the result of mapping the parameter space $A : [0,1] \times [0,1] \subset \mathbb{R}^2$ using the parametric function:

$$f(u,v) = (x(u,v), y(u,v), z(u,v)), \tag{1}$$

and a set of trimming curves $C = \{c_0, c_1, ..., c_{\ell-1}\} \in A$ (see Figure 2a), we produce a trimmed parameter space $A^T \subset A$, shown in Figure 2d, such that applying f to A^T produces the desired trimmed surface S^T. We distinguish between the final surface $S \subset \mathbb{R}^3$ and the parametric mapping function $f : \mathbb{R}^2 \to \mathbb{R}^3$ which can be used to produce S. This is because if f is applied to a different parameter space, such as A^T, a slightly different surface (S^T) will be generated.

In this paper, the trimming curves are B-spline curves, though any curve in A may be used. The i-th curve is defined as:

$$c_i(r) = \sum_{j=0}^{m-1} N_{j,k}(r) q_j^i, \quad r \in [0,1] \tag{2}$$

where $q_j^i = (u_j, v_j), 0 \le j \le m-1$, are the control points defining the curve c_i, and $N_{j,k}(r)$ are B-spline basis functions.

Let the subspace $\gamma_i \in A$, referred to as the *curve space* and shown in Figure 2b, be the region enclosed by the trimming curve c_i and $\gamma^* = \cup_{i=0}^{\ell-1} \gamma_i$ be the union of all the curve spaces. Then $A^T = A \setminus \gamma^*$ is the trimmed parametric domain which produces the surface S^T when mapped using f. However, excluding only γ^* when rendering the parametric surface is not easy as each γ_i can be any shape and their boundaries need to be accurately interpolated. To simplify the exclusion of the curve spaces, we define local subspaces around each trimming curve. Beginning with an axis-aligned bounding box, shown in Figure 2c and discussed in section 4, we use the space between c_i and the boundary of its AATR, referred to as $\mathcal{R}_i$, to create a continuous transition between these two curves. If any AATRs intersect, the shared subspace is divided using local Voronoï diagrams (section 5), as with $\mathcal{R}_1$ and $\mathcal{R}_2$ in Figures 2c and 2d. As the boundary of $\mathcal{R}_i$ is rectangular and c_i is an arbitrarily-shaped closed curve, we produce multiple transition patches $\mathcal{P}_i^j$ for $j = 0, 1, ..., k-1$ for each AATR (see Figure 2d and section 6). The set of patches which define $\mathcal{R}_i - \gamma_i$ is defined as $\mathcal{P}_i = \cup_{j=0}^{k-1} \mathcal{P}_i^j$. Each transition patch is then locally reparametrized so the trimmed edges are interpolated. Letting $\mathcal{P}^* = \cup_{i=0}^{\ell-1} \mathcal{P}_i$, then $A^T = A \setminus \mathcal{R}^* \cup \mathcal{P}^*$ will produce the desired trimmed surface S^T while ensuring the interpolation of the trimmed edges.

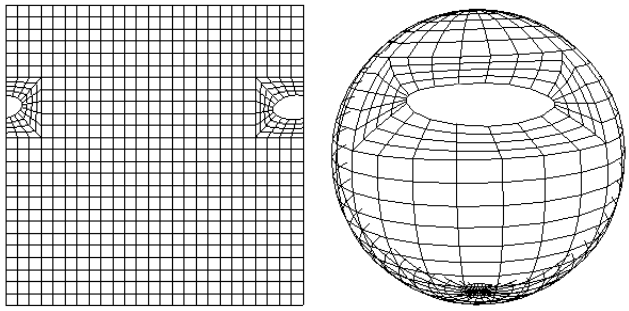
The surface is rendered in two stages. During the first stage, the unaffected region of A^T, $A \setminus \mathcal{R}^*$ is rendered, followed by the transition patches $\mathcal{P}^*$. This is discussed further in section 7.

4 Defining AATRs

To create the trimmed parameter space A^T, we first isolate the subspace around each trimming curve to provide sufficient room to create regular, high quality transition patches that interpolate the new trimmed edges. Respecting the rectangular structure of A, we isolate these areas using disjoint polygonal regions, referred to as axis-aligned trimming regions. The AATR of curve c_i is denoted $\mathcal{R}_i$ while $\cup_{i=0}^{\ell-1} \mathcal{R}_i = \mathcal{R}^*$ (see Figure 2c).

To calculate $\mathcal{R}_i$, the minimum axis-aligned bounding box of c_i is calculated and expanded by a predetermined increment which must be large enough to avoid narrow patches while still minimizing the

number of intersecting curve regions. This value can be modified later due to boundary intersections or during rendering. The sub-



(a) (b)

Figure 3: A trimming curve on the boundary of the parametric domain of a sphere is split into two curves with two independent AATRs: (a) Tessellated parametric domain; (b) Trimmed surface mesh.

spaces shared by overlapping AATRs are distributed using Voronoï diagrams, which is outlined in the next section.

Special cases occur when a trimming curve $c_i(r)$ lies on a periodic boundary of the parametric domain (Figure 3) or there is an open curve which intersects with the boundary of the parameter space. In the first case, the points of intersection between c_i and the boundary are found and the curve is split using two independent bounding boxes and thus is contained in separate AATRs. The second case is a degenerative case of the first with only one AATR required to enclose the curve.

5 Creating Localized Voronoï Diagrams

An AATR may intersect with any number of neighbouring AATRs. To evenly distribute the shared spaces, we use Voronoï curve diagrams to segment the intersecting subspace, allowing the transition patches to be built upon a single curve. As Voronoï cell boundaries equally bisect the space between neighbouring curves, this also facilitates the creation of patches of consistent size within a single AATR.

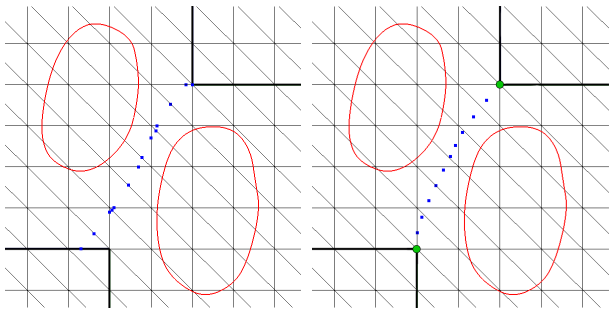
As the cell boundaries are continuous curves consisting of all points equidistant to neighbouring curves, we closely approximate them by finding a series of bisection points, which are smoothed and redistributed, to form the final cell boundaries. Our technique for calculating the Voronoï diagrams is based on that of Hamann and Tsai [7]. First, they create a triangulation of the parametric domain. Bisectors are then found on the triangle edges which results in a piecewise linear approximation of the bisection curves (as seen in Figure 4a). Details of this process can be found in [6].

We calculate the distance from a point p to curve c_i using the Haussdorff distance (see [9]):

$$d(p,c_i) = \min_{r\in[0,1)} \{\|p - c_i(r)\|\}. \tag{3}$$

Also, only triangles lying inside an AATR or on its boundary are considered, as bisection points outside of the region are irrelevant.

As the bisection points are a linear sampling of the boundary curves lying on triangle edges, the resulting curve may be jagged and have clustering of points near triangle vertices (see Figure 4a). To smooth and evenly distribute the cell boundaries while still adhering to the definition of a Voronoï curve diagram, the points are redistributed by minimizing the following energy function:



(a) (b)

Figure 4: (a) As bisection points are found on triangle boundaries, the resulting curve can be jagged; (b) Cell boundary points are smoothed using energy minimization.

$$E\left(B_{\ell,k}\right) = \sum_{j=0}^{n-1} \left\|\Delta b_j\right\|^2 w_f + \left\|D\left(b_j, c_\ell\right) - D(b_j, c_k)\right\|^2 w_v \tag{4}$$

where $B_{\ell,k} = \{b_0, b_1, ..., b_{n-1}\}$ is the set of shared boundary points between curves c_k and c_ℓ, w_f and w_v are scalar weights indicating the relative importance of fairing (term 1) versus approximating the cell boundary (term 2) respectively, and Δb_j is the first finite difference of b_j.

A simple analytical form of the gradient of E in equation (4) can be used to smooth the boundary points iteratively. Given a boundary point $b_j, 0 < j < n-1$, let b_j^i be the point after i iterations of smoothing, and p_k^i and p_ℓ^i be the closest points to b_j^i on curves c_k and c_ℓ respectively with distances d_k^i and d_ℓ^i. Then for $i > 0$,

$$b_j^i = b_j^{i-1} + w_f \Delta^2 b_j^{i-1} + w_v(\vec{v}_\ell + \vec{v}_k)$$

where $\vec{v}_\ell = (d_\ell^i - d_k^i)(p_\ell^i - b_j^{i-1})$ and similarly for $\vec{v}_k$. All examples in this paper required only two iterations.

We now have at least two sets of points associated with each curve: a collection of shared boundary points with each neighbouring curve cell, shown as the blue points in Figure 4b, and possibly a set of points from the outer boundary of the AATR, shown as the dark lines in Figure 4b. To merge these into a single boundary curve, they are sorted in consistent direction with the trimming curves (we use counter-clockwise). Each cell now has associated with it a trimming curve and a single boundary curve and can be locally reparametrized using two-dimensional patches to define the trimmed parameter space, A^T. Overlapping AATRs are replaced with the local Voronoï cell as with $\mathcal{R}_1$ and $\mathcal{R}_2$ in Figure 2.

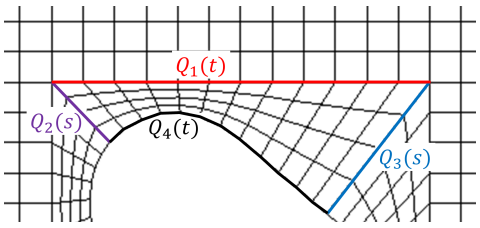


Figure 5: A discretized Coons patch with boundary curves labeled.

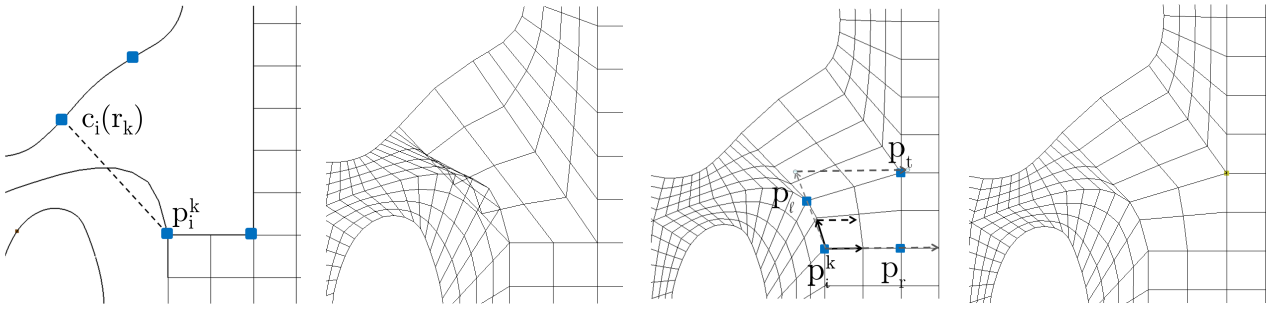


Figure 6: (a) Corners require LoS with their corresponding curve points. (b) If this is not enforced, overlapping of patches will occur. (c) Patches that are not adjacent to the cell curve are created to avoid the case in (b).

6 Segmenting the Cells

The final step in creating the trimmed parameter space is to redefine the valid regions of the AATRs ($\mathcal{R}_i - \gamma_i$ for $0 \leq i \leq n-1$) using patches in A. The subspace a transition patch defines is denoted $\mathcal{P}_i^j \subset \mathcal{R}_i \subset A$. We chose to use Coons patches because of their regularity, parametric properties, and their ability to easily transition from the trimmed edge to the surrounding parametrizations. A Coons patch, shown in Figure 5, is defined by:

$$\begin{aligned} P(s,t) =& (1-s)Q_1(t) + sQ_4(t) + \\ & (1-t)Q_2(s) + tQ_3(s) - \\ & [(1-s)(1-t)Q_2(0) + s(1-t)Q_2(1) + \\ & t(1-s)Q_3(0) + stQ_3(1)], \\ & t \in [0,1],\ s \in [0,1], \end{aligned} \tag{5}$$

where Q_i, $i = 1,2,3,4$, are the curves defining the patch's boundary.

Each cell must be split into four-sided regions, the boundaries of which define the four curves required to create the Coons patches. First, corners are detected along each cell boundary (shown in Figure 7a). As the boundary of the Voronoï diagram is aligned with the parameter axes, there are no other features along cell boundaries. Corresponding points are then located on the trimming curve (shown in Figure 7b). These point pairs are connected, creating the Coons patch's boundary. Although a single patch could be used for most AATRs, we segment $\mathcal{R}_i - \gamma_i$ using multiple patches to reduce twisting during polygonization.

To detect corners, we march along the cell boundary, finding interior angles $< \phi$, where ϕ is a predefined threshold. The boundary points between successive corners define the curve $Q_1(t)$.

To define the remaining three curves of the Coons patch, correspondences are found between corner points and points on the trimming curve. As demonstrated in Figure 7b, given a corner point p_i^k which is on the boundary of the cell surrounding the trimming curve $c_i(r)$, we wish to find r_k such that $f(c_i(r_k))$ is the closest point on c_i to $f(p_i^k)$. Notice this step is performed in object space, not parameter space. Establishing correspondences based on distances in object space produces better results on surfaces with high curvature. The portion of the trimming curve between successive matched points defines the curve $Q_4(t)$.

To complete the Coons patch, it is left to define the curves $Q_2(s)$ and $Q_3(s)$. We do this by simply connecting each corner point with its corresponding curve point. The number of points generated along Q_2 and Q_3 can be adjusted to make quads in the patch similar in size to those of the parent surface. However, the number of points on these curves must be consistent between all adjacent patches which can cause much smaller quads in interior patches. Once all curves are found, we then create each Coons patch as in equation (5). Each patch $\mathcal{P}_i^j$ represents a mapping from the locally defined subspace to A.

6.1 Isolated Patches

As seen in Figure 6a, occasionally, a corner point p_i^k does not have line-of-sight (LoS) with its corresponding curve point $c_i(r_k)$ (Figure 6a). As illustrated in Figure 6b, this can result in overlapping patches when the above method is applied. We resolve this by creating isolated patches (patches that are not adjacent to the c_i). To create an isolated patch, four non-collinear segments along the cell boundary must be identified to represent the four Coons patch curves. Since p_i^k is a corner point, we know there is one segment to the left and one to the right of the point (Figure 6c). It is left to identify a third segment and enclose the area with a fourth, creating the shared boundary with adjacent patch(es).

Testing for LoS is straightforward and can be determined by testing for an intersection between the cell boundary curve and the line segment $\overline{p_i^k c_i(r_k)}$. Finding the third curve segment can be done in a variety of ways including counting left and right hand turns while checking for LoS. We chose to use a "growing quad" algorithm, as outlined in Figure 6c. Given the corner point p_i^k, we define the points p_r and p_ℓ to be the cell boundary points to the right and left of p_i^k respectively. We also define the vectors $v_1 = p_r - p_i^k$ and $v_2 = p_\ell - p_i^k$. The fourth point of the quad, p_q, is defined as $p_q = p_\ell + v_1$. The algorithm is iterative and stops when the fourth point p_t is found. Either p_r or p_ℓ will be an interior point of the

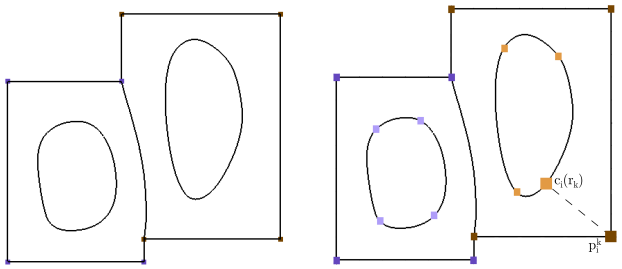
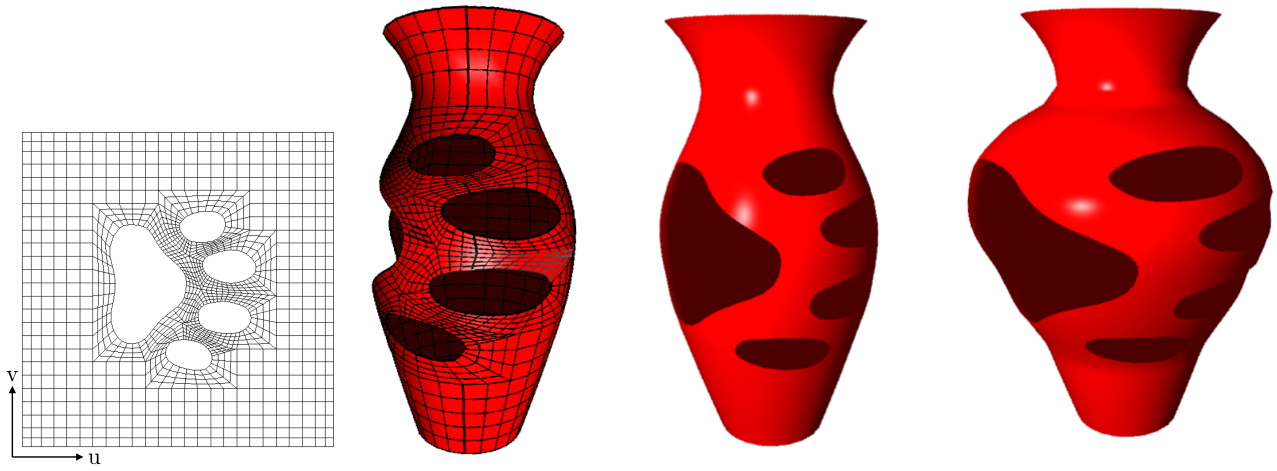
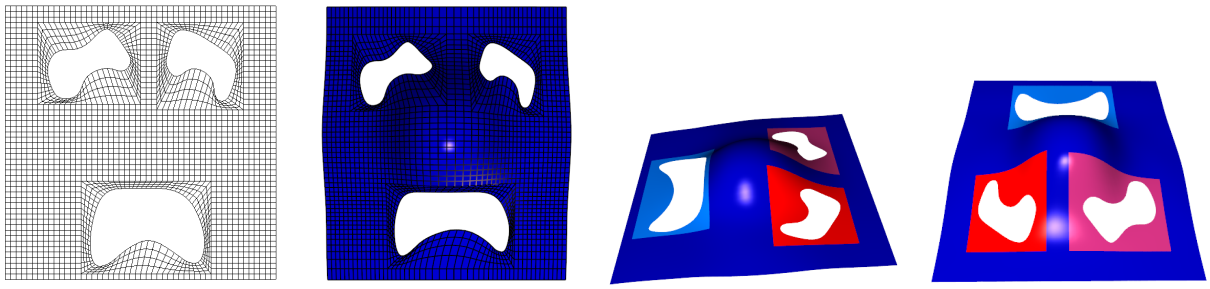


Figure 7: Segmenting two adjacent cells: (a) Corners are detected along the cell boundary; (b) Corresponding curve points are found on the cell curve. These points are connected to form the 4-sided regions required for Coons patch creation.



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

Figure 8: Example of a NURBS surface trimmed and edited using our method.



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

Figure 9: Results of trimming a surface using curves with concave regions. The AATRs have been color-coded for easy identification.

patch meaning that it, along with p_t, requires LoS with the cell curve. During iteration i of the algorithm, $v'_{1,i} = v_1 \alpha i$, where α is a small increment greater than 1. During each iteration, we find any cell boundary points enclosed by the quad and update p_ℓ and p_r as needed until p_t is found. This defines three of the segments required to create the Coons patch. The fourth segment is created by connecting the two interior points of the quad.

Once we construct any isolated patches, we have a seamless parameter space $A^T = A \setminus \mathcal{R}^* \cup \mathcal{P}^*$ which can be used to produce the trimmed surface S^T. In the following section, we describe how this surface can be tessellated.

7 Rendering the Trimmed Surface

We have created a trimmed parameter space A^T such that $f(u,v)$ for $(u,v) \in A^T$ produces the desired trimmed surface S^T. Rendering of the surface is performed in two passes to avoid gaps and ensure the trimmed edges are accurately interpolated. During the first pass, the unaffected parameter space $A \setminus \mathcal{R}^*$ is rendered as with any other parametric surface. During the second pass, the transition patches $\mathcal{P}_i^j$ for $i = 0, \ldots, \ell - 1$ are rendered to match the sampling of $A \setminus \mathcal{R}^*$, resulting in a gap-free surface that preserves the properties of the original parametric surface.

7.1 Avoiding Surface Gaps

As with any technique using adjacent surface patches, the discretization of the surface can cause gaps in the trimmed surface model if care is not taken. Gaps can be introduced into two areas of the mesh: between a patch and the trimmed parent surface; and between adjacent patches.

Gaps between a patch and the parent surface can be avoided by either forcing the parent surface to interpolate the boundaries of the AATRs or choosing the AATR boundaries wisely during their creation. We choose increments by which a curve's AABB is enlarged in section 4 such that its boundary coincides with the initial sampling of the surface. In addition to this, by using the same sampling to create the patch mesh as the surface mesh, we ensure that no gaps between a patch and the trimmed parent surface occur.

To avoid gaps between adjacent patches, it is necessary the same surface points along shared Voronoï cell boundaries are used. We do this by using the evenly distributed boundary points to create the shared Coons patch curve and then using the same increment to create each patch.

8 Results and Discussion

In this paper, a method for trimming parametric surfaces and representing the resulting surfaces has been presented. We demonstrated the trimming method using a series of example shown in Figures 1, and Figures 8-10. Subspaces aligned to the parameter axes are

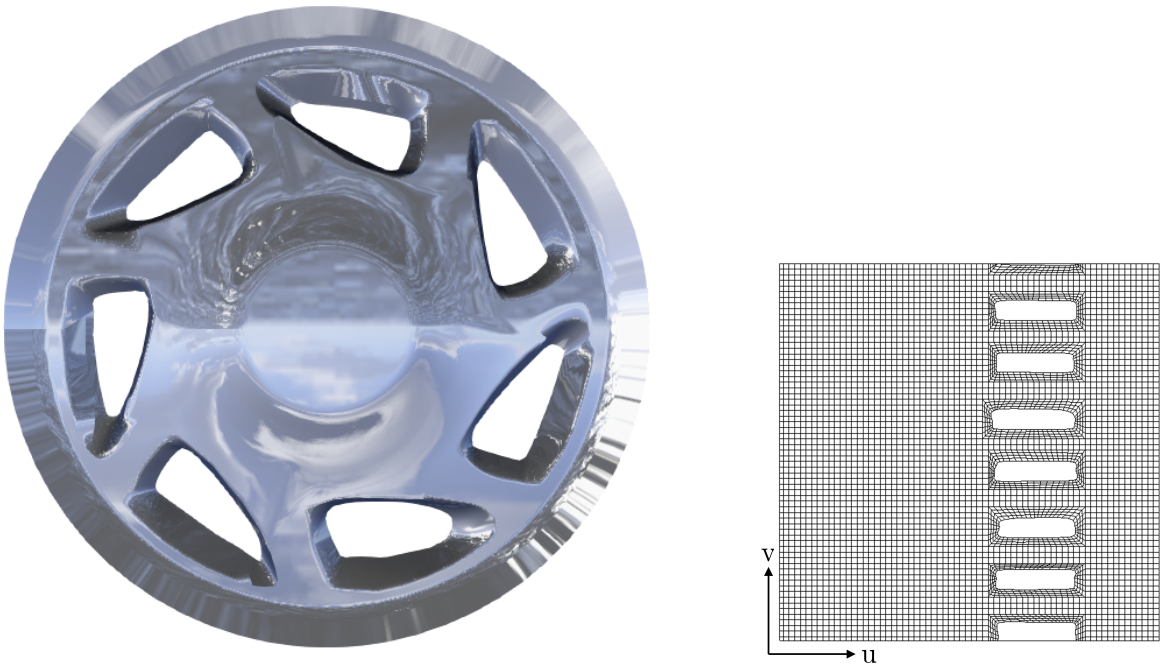


Figure 10: A trimmed hubcap model.

created around the trimming curves. Localized Voronoï diagrams are then used to segment any shared space between adjacent AATRs. The Voronoï cells are segmented by identifying sharp features along each cell boundary and then filled with Coons patches. The surface is rendered in two stages. First, the original surface is sampled excluding the AATRs. Second, the Coons patches partitioning the regions are sampled to match the sampling of the parent surface to avoid the introduction of gaps between the surface and patches.

Our technique permits simple uniform surface evaluation after trimming. The parametric properties of the surface are preserved throughout the trimming process, including the ability to edit the trimmed surface without the need to re-trim or retessellate the edited surface as demonstrated in Figure 8. In the case of NURBS surfaces, the original control points and their connection to the surface model remain intact. There is no need to add points to the trimmed surface to ensure the geometric continuity is preserved.

Curves of various shapes, including concave shapes, may be used. As seen in Figure 9, our transition patch construction is suitable for non-extreme concave shapes where each point $Q_4(t')$ on the Coons patch boundary curve has LoS with the point $Q_1(t')$ on the opposite boundary curve.

In the case of multiple trimming curves, as the curves become closer to one another, the effects the weights become more apparent when smoothing the Voronoï cell boundaries. More emphasis on approximating the cell boundary is important as too much weight on fairing can cause the boundary curve to intersect one of the trimming curves (see Figure 11a and 11b), causing anomalies in the final trimmed surface model. The gap-prevention heuristic requires that the number of quads per patch not be affected by the surface area a patch encompasses. As shown in Figures 11c and 11d, this means that as two curves approach one another, the number of quads between them does not change drastically and the quad-size shrinks. Since there are a sufficient number of quads inside this narrow space, the surface is rendered properly (with two holes separated by a narrow, crack-free, band) (see Figure 11e), even when the sampling increment is larger than the space between the curves.

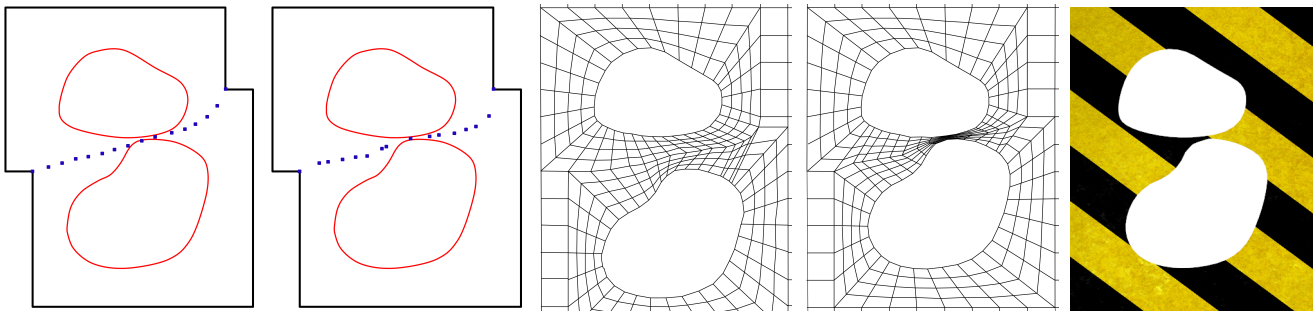
8.1 Surface Continuity

Manipulating the parameter space may affect the parametric continuity of the surface. Introducing trimmed regions into A introduces discontinuity into S as u and v are undefined within the trimmed regions. Also, reparametrization of transitional areas can effect the continuity. Each transition patch has a different parametrization from the original surface parametrization. This may change the parametric continuity inside of the AATRs. This is mainly due to the change of partial derivatives inside the AATRs. However, the geometric continuity of the surface is not affected as we only change the sampling of the surface points and not the surface itself. For example, in Figure 1f, although the parametric continuity has been changed (see Figure 1g), the geometric continuity has not (see Figure 1h).

9 Conclusion and Future Work

Creating axis-aligned subregions allows for easy identification of the subregion a point is in when evaluating the surface. Compared to Hui and Wu, our method produces more regular meshes with generally fewer patches; we take into account the sampling of the surface as the patches are created, making finding subregions much more straightforward; and because our subregions are in the parametric domain, no continuity constraints need be implemented to ensure the trimmed surface has the same geometric continuity as the original. Additionally, our method works for all parametric surfaces including general surfaces (such as generalized cylinders), B-Spline, NURBS and PUPs [14] surfaces.

We would like to expand our AATR segmentation algorithm such that it is capable of segmenting highly concave curves. In this case, parts of a trimming curve may not have line-of-site with the boundary of its associated transition patch. A more robust cell decomposition which detects and isolates these regions would be required. The method of cell segmentation used will not impact any of the other steps in the trimming process as long as sample points along shared transition patch boundaries are matched during rendering as this is crucial to avoiding cracks in the final trimmed model.



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

Figure 11: As two curves approach one another, the number of quads between them does not change drastically and the size of the quads between them shrink. When they become very close, boundary smoothing weights are adjusted. (a) Smoothed bisection points with fairing and distance of equal weight; (b) Smoothed bisection points with fairing weight of 0.125 and distance 0.875; (c) Final polygonized model with equal weights and curves farther apart; (d) Final polygonization of (b); (e) The final model with texture.

ACKNOWLEDGMENTS

This work has been partially supported by GRAND NCE and NSERC.

REFERENCES

[1] H. Alt and O. Schwarzkopf. The voronoi diagram of curved objects. In *Proceedings of the Eleventh Annual Symposium on Computational Geometry*, SCG '95, pp. 89–97. ACM, New York, NY, USA, 1995. doi: 10.1145/220279.220289

[2] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy. *Polygon Mesh Processing*. AK Peters, 2010.

[3] G. K. L. Cheung, R. W. H. Lau, and F. W. B. Li. Incremental rendering of deformable trimmed NURBS surfaces. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology - VRST '03*, p. 48, 2003. doi: 10.1145/1008653.1008664

[4] R. Cripps and S. Parwana. A robust efficient tracing scheme for triangulating trimmed parametric surfaces. *Computer-Aided Design*, 43(1):12–20, Jan. 2011. doi: 10.1016/j.cad.2010.08.009

[5] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications 3ED*. Springer Berlin Heidelberg, 2008.

[6] T.-P. Fang and L. Piegl. Delaunay triangulation using a uniform grid. *Computer Graphics and Applications, IEEE*, 13(3):36 –47, may 1993. doi: 10.1109/38.210490

[7] B. Hamann and P.-Y. Tsai. A tessellation algorithm for the representation of trimmed surfaces with arbitrary trimming curves. *Computer-Aided Design*, 28(6-7):461–472, July 1996. doi: 10.1016/0010-4485(95)00043-7

[8] K. Hui and Y.-B. Wu. Feature-based decomposition of trimmed surface. *Computer-Aided Design*, 37(8):859–867, July 2005. doi: 10.1016/j.cad.2004.09.014

[9] R. Klein and W. Straber. Large mesh generation from boundary models with parametric face representation. In *Proceedings of the Third ACM Symposium on Solid Modeling and Applications*, SMA '95, pp. 431–440. ACM, New York, NY, USA, 1995. doi: 10.1145/218013.218097

[10] S. Kumar and D. Manocha. Efficient rendering of trimmed nurbs surfaces. *Computer-Aided Design*, 27(7):509 – 521, 1995. doi: 10.1016/0010-4485(94)00003-V

[11] L. A. Piegl and A. M. Richard. Tessellating trimmed nurbs surfaces. *Computer-Aided Design*, 27(1):16 – 26, 1995. doi: 10.1016/0010-4485(95)90749-6

[12] L. A. Piegl and W. Tiller. Geometry-based triangulation of trimmed NURBS surfaces. *Computer-Aided Design*, 30(1):11–18, Jan. 1998. doi: 10.1016/S0010-4485(97)00047-X

[13] A. Rockwood, K. Heaton, and T. Davis. Real-time rendering of trimmed surfaces. *SIGGRAPH Comput. Graph.*, 23(3):107–116, July 1989. doi: 10.1145/74334.74344

[14] A. Runions and F. F. Samavati. Partition of unity parametrics: a framework for meta-modeling. *The Visual Computer*, 27:495–505, 2011. doi: 10.1007/s00371-011-0567-x

[15] A. Schollmeyer and B. Fröhlich. Direct trimming of NURBS surfaces on the GPU. *ACM Transactions on Graphics*, 28(3):1, July 2009. doi: 10.1145/1531326.1531353

[16] T. W. Sederberg, G. T. Finnigan, X. Li, H. Lin, and H. Ipson. Watertight trimmed NURBS. *ACM Transactions on Graphics*, 27(3):1, Aug. 2008. doi: 10.1145/1360612.1360678

[17] X. Sheng and B. E. Hirsh. Triangulation of trimmed surfaces in parametric space. *Computer-Aided Design*, 24(8):437–444, 1992.