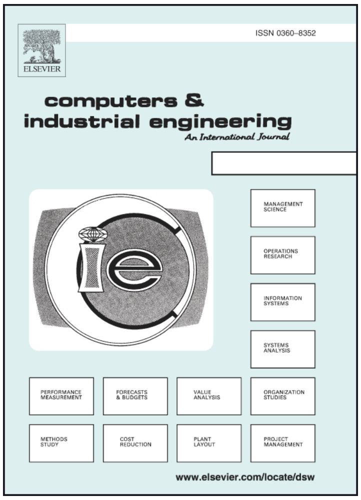
Accepted Manuscript



Optimized normal and distance matching for heterogeneous object modeling

Kuntal Samanta, Ibrahim T. Ozbolat, Bahattin Koc

PII: S0360-8352(13)00390-2
DOI: http://dx.doi.org/10.1016/j.cie.2013.12.010
Reference: CAIE 3614

To appear in: *Computers & Industrial Engineering*

Received Date: 16 October 2012
Revised Date: 30 August 2013
Accepted Date: 16 December 2013

Please cite this article as: Samanta, K., Ozbolat, I.T., Koc, B., Optimized normal and distance matching for heterogeneous object modeling, *Computers & Industrial Engineering* (2013), doi: http://dx.doi.org/10.1016/j.cie.2013.12.010

Optimized normal and distance matching for heterogeneous object modeling

Kuntal Samanta[1], Ibrahim T. Ozbolat[2,3], Bahattin Koc [4,*]

[1] *Industrial and Systems Engineering Department, University at Buffalo (SUNY)*
438 Bell Hall, Buffalo, NY 14260, USA

[2] *Mechanical and Industrial Engineering Department, The University of Iowa*
2130 Seamans Center, Iowa City, IA 52242, USA

[3] *Center for Computer-Aided Design, The University of Iowa*
216 Engineering Research Facility, Iowa City, IA 52242, USA

[4] *Faculty of Engineering and Natural Sciences, Sabanci University, FENS G013*
Orhanli-Tuzla, Istanbul, 34956, Turkey

Abstract

This paper presents a new optimization methodology of material blending for heterogeneous object modeling by matching the material-governing features. The proposed method establishes point-to-point correspondence represented by a set of connecting lines between two material directrices. To blend the material features between the directrices, a heuristic optimization method is developed to maximize the sum of the inner products of the unit normals at the end points of the connecting lines and minimize the sum of the lengths of the connecting lines. The geometric features with material information are matched to generate non-self-intersecting and non-twisted connecting surfaces. By subdividing the connecting lines into an equal number of segments, a series of intermediate piecewise curves is generated to represent the material metamorphosis between the governing material-features. A dynamic programming approach developed in our earlier work is presented for comparison purposes, and the computational efficiency of the proposed heuristic method is also compared with earlier techniques in the literature. Computer interface implementation and illustrative examples are also presented in this paper.

Keywords: Heterogeneous object modeling, normal and distance matching, curved matching, ruled surface, metamorphosis.

1. Introduction

Heterogeneous objects are made of different materials where each material contributes to a certain property. By properly controlling the constituent material compositions, a heterogeneous object can be designed to exhibit different properties in different areas of the object. If designed properly, these objects can perform better than their homogeneous counterparts in many different engineering applications because of their ability to satisfy multiple property requirements spatially [1]. In our earlier work [1], we presented a feature-based heterogeneous object modeling method. Object features that control material composition are identified as material-governing features because they dictate the material variation inside the object. A lofting-based method [1] was used to blend the material features.

[*] Corresponding author: Tel.: +90-216-483-9557; fax: +90-216-483-9550. *E-mail address:* bahattinkoc@sabanciuniv.edu (B. Koc).

To generalize material blending between any material-governing features, a new method of material blending between the material governing features is presented in this paper. As shown in Fig. 1, a smooth transition between two given material-governing features is provided to blend the material properties for heterogeneous object modeling. This can be also used as shape blending, morphing, or metamorphosis, which is represented by a series of "in-between" curves generated as a linear combination of the input curves. In other words, a ruled surface is generated using the inputs as directrices [2, 3], and each isoparametric curve on the surface constitutes a stage of the metamorphosis.

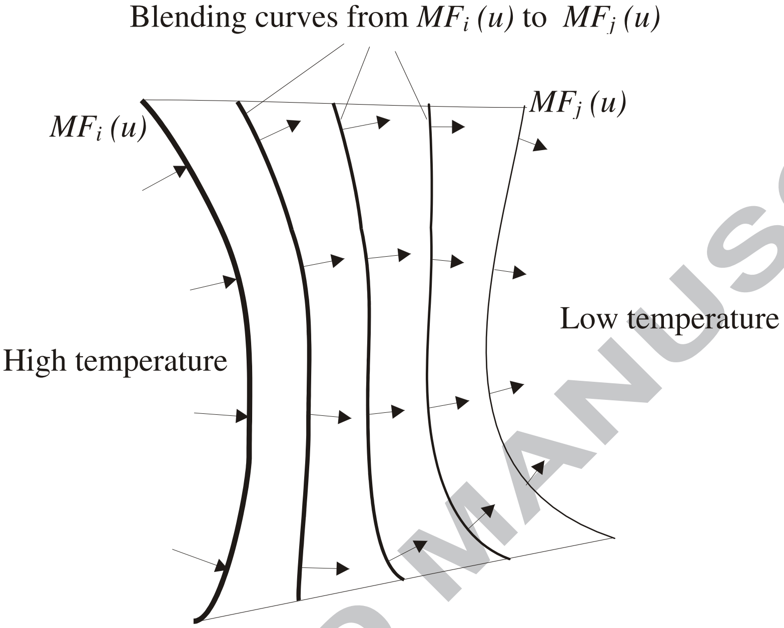


Figure 1: Material (property) blending between two material-governing features $MF_i\ (u)$ and $MF_j\ (u)$

In our previous work [1], we found that the property requirement at each material-governing feature followed the normal direction. The outcome of the lofting process represents a smooth blending among its generators. In the case of surface lofting from generator curves, the resulting surface passes smoothly through each of the curves. In the same way, lofting can be used to get a smooth transition from one governing feature to another. As shown in Fig. 1, the property requirements at each of the generators are blended along the normal direction from the generators. It is assumed that each isoparametric entity in the blend direction will represent constant property requirements. Fig. 1 shows how two different material property requirements are blended using a lofting process. Two material governing features (curves) MF_i (u) and MF_j (u) are exposed to high and low temperatures, respectively, and therefore exhibit different property requirements. It is a known fact that, from a heated body, heat flows in the normal direction from every point of the body. Therefore, the isoparametric curves on the loft surface will represent iso-conditions (iso-temperatures). In the figure, the thicknesses of the curves show the temperature intensity and therefore different property requirements. To be able to blend the material between two or more generators along their normal directions, the normal vectors of the material-governing features must match. While matching the normal vectors, the following conditions must be met for smooth transition:

(a) The connecting lines must not intersect (there should not be any intersections of two or more ruling lines).
(b) The generated surface must not be twisted, i.e., the length of the ruling lines must be as short as possible.

(c) The end points of each ruling line must be *matching,* i.e., they must have some common or similar properties.

This problem can be generalized by generating a ruling surface between two directrices. A naïve way of constructing the ruling lines is by parametrically connecting the points on the two directrices. The rationale here is that both end points of each ruling line have the same parameter values. This does not guarantee a non-twisted ruled surface, particularly in the case of directrices given as closed curves. The surface may also become self-intersecting and therefore unsuitable for the material blending of metamorphosis, as also pointed out by Elber [4] and Surazhsky *et al.* [5]. Therefore, other sophisticated methods are required to find the "best" set of ruled lines that satisfy all the conditions mentioned above.

In the literature, several methods have been proposed, mostly catering to specific applications. For shape blending, attention is focused on establishing matching between the input curves based on common local properties such as position, edge/arc length, angle, parameter, tangent and curvature. Sederberg and Greenwood [6] established vertex-to-vertex matching based on locations and angles at the vertices of 2D polygonal curves. The intermediate shapes are obtained by linear interpolation of the matched vertices. The method also tries to avoid local self-intersections. The same authors have used a similar method [7] in which the inputs are given as piecewise polynomial curves. Meek *et al.* [33] developed a blending formula for two open curve segments to generate accurate blending with better approximation.

When combinations of some of the common characteristics are found over a range of the input curves, they are referred to as "features." Although the exact definitions of "features" vary by methods, the common goal is to establish matching between the features and to ensure that they are maintained during the blending process. Hui and Li [8] used 2D shapes composed of curve segments and defined rules in order to identify features. They developed algorithms to match the features based on their positions and shapes.

The approach of Cohen *et al.* [9] is to establish matching between two given C^1 continuous parametric curves based on tangent maps. To construct a set of non-self-intersecting ruling lines, one curve is reparameterized with respect to the other by means of formulating an optimization problem. The objective is to maximize the sum of the dot products between the unit tangents to the curves at the matched points. Constraints are specified to avoid local self-intersection. An approximate solution to the problem is obtained by first discretizing the curves into piecewise polygonal curves and is subsequently solved using dynamic programming. However, the method is explained in detail only for matching open curves, where matching of the end points of the curves is constrained. For matching closed curves, the authors suggest using a *k*-shift of tangents approach.

Johan *et al.* [10] also used optimization to match equally spaced sampled points from two given input curves. The objective here is to minimize the sum of a cost function, which is composed of a weighted sum of the difference of angles and difference of parameters at the matched vertices. Dynamic programming is used to solve the problem. The overall approach, as the authors mentioned, is an extension of earlier work [6, 9].

It is argued that feature identification and matching is important so that the matched features transform into each other smoothly while ensuring no deformation or distortion of the in-between curves. In most cases of shape blending, feature identification is easy because the input shapes already contain some similarities between themselves. Although this helps in achieving visually pleasing animations, the resulting ruled surface may not always be twist- or stretch-free. Moreover, in cases of input curves that exhibit very little or no similarity, it may be very difficult to identify common features. Therefore, in such applications, other local properties are used for establishing the matching.

In another work [11], the authors presented optimal boundary triangulations of interpolating ruled surfaces. This paper presented an algorithm for constructing an optimal triangulated ruled surface that interpolates two discrete directrices. The developed algorithm, called the multilayer directed graph, was used to establish an equivalence between the optimal triangulation and the single-source shortest path problem on the graph.

In the use of ruled surfaces in Computer-Aided Design and Manufacturing (CAD/CAM) applications, such as the adaptive ruled layers approximation for multi-axis machining in rapid prototyping, the directrices are given as two consecutive piecewise linear curves (or polygonal contours). In work by Koc *et al.* [12], the directrices were obtained by slicing a stereolithography (STL) file. The authors found the matching points so as to minimize the twisting of the resulting ruled surface by minimizing the total ruling line length. Both curves were reparameterized by means of inserting points so that one-to-one matching could be achieved.

For multi-axis machining of ruled surfaces [12-21], the cylindrical or conical cutter contacts the surface along a ruling line. In the case of developable ruled surfaces, the axis of the cutter is parallel to the ruling line by a constant offset. Therefore there is no deviation between the machined surface and the designed surface. However, this is not the situation in the case of machining undevelopable ruled surfaces where it is impossible to avoid machining errors such as gouging and interference. Bohez *et al.* [14], Yang *et al.* [15], Tsay *et al.* [16] and Senatore *et al.* [20] showed that, for a cutter used for side milling of a ruled surface, the machining error is a function of the angle between the normal vectors to the ruled surface at the two ends of the ruling line. To reduce the errors, alignment of the cutter axis must minimize the angular difference between the normals. This problem can also be viewed as a matching problem between the directrices, i.e., to determine which ruling lines the cutter will touch so that the machining errors are minimized. Usually any feature resemblance between the directrices is irrelevant here. Therefore, matching by normals is considered a neutral and general approach that can have a multitude of applications.

Surface reconstruction has been used for biomedical applications, such as organ reconstructions and tissue engineering devices. Ozbolat *et al.* [28] proposed feature-based design of biodegradable micro-patterned structures by varying surface pattern architecture throughout the structure according to tissue engineering needs. Surface pattern was developed over uniform degradation regions. These regions were generated by surface blending between two patterning features defined by nonintersecting freeform curves. Surface pattern varied smoothly based on a formulated surface architecture optimization problem. Kale [29] studied the bio-mimetic design of heterogeneous scaffolds such as bandages for improved wound healing. Bio-molecule distribution was controlled over several uniform regions throughout the scaffold designed for solid freeform fabrication techniques. Firstly, geometric features of wound were captured using imaging techniques. Uniform regions were then generated by a blending process for geometric features defined by nonintersecting freeform curves, in which one curve encloses the other. Lim *et al.* [30] reconstructed 3D solid biological objects such as a femur, a knee and vertebrae from a cross-sectional data set using 2D Delaunay triangulation and a non-linear optimization process with a cost function. After non-linear optimization, a blending process with weighted layer information was utilized to join individual 2D layers into 3D objects, resulting in a smooth transition between cross-sections.

In work by Marsan *et al.* [31], surface reconstruction from planar contours for branched objects was developed using contour generation from a stack of 2D binary images. Generation of a transition contour between the base group contours and the branched contour was performed for the skinning process using parameter correspondence. In parameter correspondence, a periodic parametric spline surface was generated between a set of points on two non-intersecting curves. First, a set of control points was generated with a parametric equidistance on two contours. While keeping points on the outer contour fixed, points on the inner contour were shifted by matching the angular angle with respect to the center point of the inner contour. Hence twisting was relatively prevented on the skinned surface, but center point approximation generated distorted results. Bajaj *et al.* [32] reconstructed 3D objects from 2D scalar-valued slices obtained by an imaging system. A Voronoi diagram was used to generate triangles of iso-surfaces between slices. An optimization procedure was developed for tiling triangles by selecting candidate triangles with the shortest chord lengths. Near-optimal results were obtained, but distorted results could be generated under certain circumstances. Marker *et al.* [34] used Hermite cubic splines to construct blending slices between parallel input contours for medical imaging applications. A monotonicity constraint was incorporated to guarantee locally monotone spline ensuring smooth slices but resulted in artifacts under some circumstances.

This paper proposes a new optimization methodology for matching between two material-governing features [1] in closed freeform B-spline curves for heterogeneous object modeling. This methodology introduces a novel normal matching scheme between freeform curves to ensure smooth and optimum transition in feature variation over object geometry. The proposed methodology has been used in the application of computer-aided modeling of novel implantable devices for emerging research areas such as biomedical engineering and computer-aided tissue engineering [28-29]. The conditions for the matching of two points, each on one of the given curves, are specified as follows: (i) the sum of the angular distances between the normals to the curves at the matching points must be minimized, and (ii) the sum of the Euclidean distances between the matching points must be minimized. The normal vector-matching criterion has been selected because the matching is not dependent on any feature identification. The problem is formulated as a continuous optimization problem. A Greedy Ruling Line Construction (*GRLC*) method is proposed to find the global optimum discrete matching that establishes point-to-point correspondence between all vertices. A condition that the total number of ruling lines generated in the process is minimized is also ensured. A vertex insertion method is also proposed to ensure one-to-one correspondence. Alternatively, a dynamic programming approach is presented to compare the results and efficiency.

2. Feature-based representation of heterogeneous objects

In this paper, the directrices used for generating the ruled line set are given as planar, freeform, closed B-spline curves [22]. The proposed method utilizes the normals at the sampled points for matching. Therefore it is necessary that both the curves be at least C^1 continuous so that well-defined normals exist at all points on the curves. In cases where the inputs are given as point sets or polygonal contours, one can fit a closed B-spline curve that satisfies at least C^1 continuity. In the following subsections, the details of constructing closed and at least C^1 continuous B-spline curves are presented.

A B-spline curve of degree p is defined as follows [22]:

$$\mathbf{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\mathbf{P}_i \tag{1}$$

where the $\{\mathbf{P}_i\}_{i=0,\ldots,n}$ are the control points and $\{N_{i,p}(u)\}$ are the p-th degree B-spline basis functions defined on the knot vector $U=\{u_0,\ldots,u_h\}$, where $h=n+p+1$;

B-spline curves C(u) in this work are assumed to be closed and satisfy at least C^1 continuity. Any degenerate cases (i.e., the whole curve degenerates to a point where C^1 continuity is not maintained) are ruled out.

Two directrices given as two non-self-intersecting, closed, at least C^1 continuous, planar, B-spline curves, $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$, of degrees p_1 and p_2, respectively. Planes of both curves are assumed to be parallel to each other. The degrees p_1 and p_2 need not be the same. Both curves are non-self-intersecting (except at the joining point), and they do not intersect each other in IR^3, i.e., the following conditions in Equation (2) are satisfied:

$$\begin{aligned}
\mathbf{C}_1(u_i)\neq\mathbf{C}_1(u_j)\;\; & u_i\in(u_{p_1},u_{h_1-p_1});u_j\in(u_{p_1},u_{h_1-p_1});\; u_i\neq u_j\\
\mathbf{C}_2(v_i)\neq\mathbf{C}_2(v_j)\;\; & v_i\in(v_{p_2},v_{h_2-p_2});v_j\in(v_{p_2},v_{h_2-p_2});\; v_i\neq v_j\\
\mathbf{C}_1(u)\neq\mathbf{C}_2(v)\;\; & u\in[u_{p_1},u_{h_1-p_1}];v\in[v_{p_2},v_{h_2-p_2}]
\end{aligned}\qquad (2)$$

As have been mentioned before, the ruled surface constructed by parametrically connecting the points on the curves $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$ is not guaranteed to be twist-free. Therefore, it is required that $\mathbf{C}_2(v)$ be reparameterized with respect to $\mathbf{C}_1(u)$ before generating the surface and the ruling lines. The reparameterization domain is the same as that of the curve $\mathbf{C}_1(u)$, i.e., $u\in[u_{p_1},u_{h_1-p_1}]$. Let this domain be renamed as $u\in[u_{low},u_{high}]$. If the reparameterized version of $\mathbf{C}_2(v)$ is denoted as $\mathbf{C}_2(v(u))$, then a ruling line $RL(t)$ with end points $\mathbf{p}=\mathbf{C}_1(u_k)$ and $\mathbf{q}=\mathbf{C}_2(v(u_k))$ is given as

$$RL(t)=t\mathbf{p}+(1-t)\mathbf{q};\quad t\in[0,1]\qquad (3)$$

The ruling line $RL(t)$ in Equation (3) represents a match between its end points. Therefore, it is required that the reparameterization of $\mathbf{C}_2(v)$ with respect to $\mathbf{C}_1(u)$ is such that the end points of all the ruling lines are matched. A ruling line $RL(t)$ is introduced only when the following two conditions are satisfied:

(a) In order to blend a heterogeneous object between two or more generators (ruling lines) along their normal directions, the normal vectors of the features must match for smooth transitions. Perfect matching of $\mathbf{p}$ and $\mathbf{q}$ is obtained when both unit normals become collinear with the ruling line $RL(t)$. Hence, the inner product of the unit normal vectors to the curves $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v(u))$ at $\mathbf{p}$ and $\mathbf{q}$, respectively, is maximized.

(b) To prevent twisting of the ruled surface, lengths of ruling lines need to be minimized. Thus, the square of the length of the ruling line $\left|\overrightarrow{\mathbf{p}-\mathbf{q}}\right|$ is minimized.

To mathematically express these two conditions, a function *f* can be defined that assigns a value to each ruling line as follows.

$$f(\mathbf{p},\mathbf{q}) = \frac{\langle \mathbf{N}(\mathbf{p}), \mathbf{N}(\mathbf{q}) \rangle^{w_n}}{\left| \overrightarrow{\mathbf{p}\text{ - }\mathbf{q}} \right|^{w_d}} \tag{4}$$

In the formulation, w_n and w_d are the weight factors of the normal matching and ruling line lengths, respectively. Different weight factors can be assigned to each condition in Equation (3) depending on the application.

Without loss of generality, it can be assumed that $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$ are mapped on the *xy*-plane and that they lie on the *xy*-plane. Although $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$ do not intersect in 3D space, their projections on the *xy*-plane can intersect. The length of a ruling line *RL*(*t*) between two points $\mathbf{p}$ and $\mathbf{q}$ on curves $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$, respectively, determines twisting on the ruling surface. While two curves lie on two planes that are parallel to each other, distance in the -*z* direction can be ignored during ruling length calculation. As described earlier, twisting is prevented by minimizing the total length of the ruling lines. While the space curves in this paper are mapped on an *x*-*y* plane and considered as plane curves,the unit normal vector of a plane curve, which is the rotation of unit tangent vector by an angle of $\pi/2$to the left (if the orientation is positive) [35], is calculated for curves $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$ as follows:

$$\mathbf{N}(\mathbf{p}) = \frac{\mathbf{C}_1'(u_k)}{\left|\mathbf{C}_1'(u_k)\right|} \times \mathbf{k} \text{ and, } \mathbf{N}(\mathbf{q}) = \frac{\mathbf{C}_2'(v_k)}{\left|\mathbf{C}_2'(v_k)\right|} \times (\text{-}\mathbf{k}) \tag{5}$$

where **k** is the unit vector in the positive z-direction.

Now the global curve-matching problem can be formulated as a continuous optimization problem where the objective is to maximize the sum of the function *f* over the entire parameter domain of the curve $\mathbf{C}_1(u)$, i.e., $u \in [u_{low}, u_{high}]$.

$$\text{Maximize} \int_{u_{low}}^{u_{high}} \frac{\langle \mathbf{N}(\mathbf{C}_1(u)), \mathbf{N}(\mathbf{C}_2(u(v))) \rangle^{w_n}}{\left| \overrightarrow{\mathbf{C}_1(u) - \mathbf{C}_2(u(v))} \right|^{w_d}} \tag{6}$$

subject to the following constraints:

1. Two consecutive ruling lines $\overline{\mathbf{C}_1(u_i)\text{ - }\mathbf{C}_2(v_j)}$ and $\overline{\mathbf{C}_1(u_{i+1})\text{ - }\mathbf{C}_2(v_{j+1})}$ should not intersect each other.
2. No ruling line $\overline{\mathbf{C}_1(u_i)\text{ - }\mathbf{C}_2(v_j)}$ should intersect the directrices $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$.

Note that no initial reparameterization is specified as a constraint because the curves are not open.

$\mathbf{C}_1(u)$ is sampled into a set of (*a* + 1) points **P** as follows:

$$\mathbf{P} = \{\mathbf{p}_i\}_{i=0,\ldots,a}\text{; where, } \mathbf{p}_i = \mathbf{C}_1(u_i)\text{; } u_i \in [u_{p_1}, u_{h_1-p_1}]\text{; } u_i < u_{i+1}\text{; } u_0 = u_{p_1}\text{; } u_a = u_{h_1-p_1}$$
$$\mathbf{p}_0 = \mathbf{C}_1(u_0) = \mathbf{C}_1(u_{p_1}) \text{ and } \mathbf{p}_n = \mathbf{C}_1(u_a) = \mathbf{C}_1(u_{h_1-p_1}) \tag{7}$$

Similarly, $\mathbf{C}_2(v)$ is a reparameterization into a set of $(b + 1)$ points $\mathbf{Q}$ as follows:

$$\mathbf{Q} = \{\mathbf{q}_j\}_{j=0,\ldots,b} \text{; where, } \mathbf{q}_i = \mathbf{C}_2(v_j)\text{; } v_j \in [v_{p_2}, v_{h_2-p_2}]\text{; } v_j < v_{j+1}\text{; } v_0 = v_{p_2}\text{; } v_b = v_{h_2-p_2}$$
$$\mathbf{q}_0 = \mathbf{C}_2(v_0) = \mathbf{C}_2(v_{p_2}) \text{ and } \mathbf{q}_b = \mathbf{C}_2(v_b) = \mathbf{C}_2(v_{h_2-p_2}) \tag{8}$$

Note that the parameters u_i's and v_j's are not necessarily evenly distributed in their respective domains. Moreover, a and b are not assumed to be equal. Although $\mathbf{q}_j$ is obtained by reparameterization of $\mathbf{C}_2(v)$ with respect to $\mathbf{C}_1(u)$, ruling line construction does not necessarily follow the same parametric sequence. A ruling line can be introduced between a combination of any two points $\mathbf{p}_i \in \mathbf{P}$ and $\mathbf{q}_j \in \mathbf{Q}$ on curves $\mathbf{C}_1(u)$ and $\mathbf{C}_2(v)$, respectively. While sampling at every point $\mathbf{p}_i$ and $\mathbf{q}_j$, the unit normals $\mathbf{N}(\mathbf{p}_i)$ and $\mathbf{N}(\mathbf{q}_j)$ are also calculated. Now the discrete version of the function *f* in Equation (4) becomes

$$f(\mathbf{p}_i, \mathbf{q}_j) = \frac{\left\langle \mathbf{N}(\mathbf{p}_i), \mathbf{N}(\mathbf{q}_j) \right\rangle^{w_n}}{\left|\overrightarrow{\mathbf{p}_i - \mathbf{q}_j}\right|^{w_d}} \tag{9}$$

Therefore, the discrete approximation of the original continuous optimization problem in Equation (6) can be expressed as:

$$\text{Maximize} \sum_{i=0}^{a} \sum_{j=0}^{b} \frac{\left\langle \mathbf{N}(\mathbf{p}_i), \mathbf{N}(\mathbf{q}_j) \right\rangle^{w_n}}{\left|\overrightarrow{\mathbf{p}_i - \mathbf{q}_j}\right|^{w_d}} \tag{10}$$

Subject to the following constraints.

1. So that $j(i) < j(i+1)$ is a valid discrete sampling, two consecutive ruling lines $\overline{\mathbf{p}_i \mathbf{q}_{j(i)}}$ and $\overline{\mathbf{p}_{i+1} \mathbf{q}_{j(i+1)}}$ should not intersect each other, i.e., $j(i) < j(i+1)$.
2. No ruling line $\overline{\mathbf{p}_i \mathbf{q}_{j(i)}}$ should intersect the polygons $\mathbf{P}$ and $\mathbf{Q}$.

3. Optimum matching of reparameterized geometric features

This section describes the solution methodology to the discrete optimization problem for ruled line construction in Equation (10). The Greedy Ruled Line Construction (GRLC) approach is proposed to find a set of ruling lines *RL* that maximizes the objective function in Equation (10). The underlying principle of the GRLC approach is to construct the set *RL* by adding one ruling line at a time that increases the objective function value the most. At every stage of *RL* construction, the ruling line added to *RL* is chosen from a set of candidates named *RL_candidate_list*. Each candidate in *RL_candidate_list* is called a *maximum valued ruling line* $(MVRL_i)$ and satisfies both constraints 1 and 2 in Equation (10). Construction of *RL* continues until all the vertices on both polygons $\mathbf{P}$ and $\mathbf{Q}$ are connected by at least one ruling line.

A maximum valued ruling line $(MVRL_i)$ represents the best match for a given vertex $\mathbf{p}_i \in \mathbf{P}$. If the ruling line $\overline{\mathbf{p}_i \mathbf{q}_j}$ satisfies both constraints in Equation (10) and at the same time it so happens that

$f(\mathbf{p}_i,\mathbf{q}_j)=\max\{f(\mathbf{p}_i,\mathbf{q}_j)\}_{j=0,\ldots,b}$ is true, then $\overline{\mathbf{p}_i\mathbf{q}_j}$ is designated as the $MVRL_i$. Since at every stage a new ruling line is added to *RL*, the $MVRL_i$ may not always remain the same for the same $\mathbf{p}_i$. This is why the *RL_candidate_list* is emptied and reconstructed at every stage of *RL* construction. It is assumed that $MVRL_i$ is found while there are already some ruling lines in *RL*, none of which has $\mathbf{p}_i$ as an end point. In other words, the GRLC method has already progressed to a stage when $MVRL_i$ will be one of the candidates in the *RL_candidate_list* and will enter *RL*.

The second constraint in Equation (9) is met by a visibility checking method as defined below.

Definition: Visibility – A point $\mathbf{q}_j\in\mathbf{Q}$ is visible to $\mathbf{p}_i$ if the ruling line $\overline{\mathbf{p}_i\mathbf{q}_j}$ does not intersect any edges of either of the polygons $\mathbf{P}$ and $\mathbf{Q}$.

While finding the $MVRL_i$ of the given vertex $\mathbf{p}_i$, only those vertices in $\mathbf{Q}$ are considered which are visible to $\mathbf{p}_i$. A function $\mathbf{IsVisible}(\mathbf{p}_i,\mathbf{q}_j)$ is defined which returns *true* only if $\mathbf{q}_j$ is visible to $\mathbf{p}_i$. Let $\mathbf{V}_i$ be a subset of $\mathbf{Q}$ so that all vertices in $\mathbf{V}_i$ are visible to $\mathbf{p}_i$.

$$\mathbf{V}_i=\{\mathbf{q}_j\in\mathbf{Q}\,|\,\mathbf{IsVisible}(\mathbf{p}_i,\mathbf{q}_j)=true\}_{j=0,\ldots,b} \tag{11}$$

Figure 2 gives an example of how a ruling line $\overline{\mathbf{p}_i\mathbf{q}_j}$ enters *RL*. Figure 2(b) explains Equation (11) where the vertex $\mathbf{p}_i$ is connected by broken ruling lines to all the vertices in $\mathbf{V}_i$. The $MVRL_i$ is one of the broken lines, but is not identified yet.

In order to meet the first constraint in Equation (10), all vertices in $\mathbf{P}$ are traversed in the counterclockwise direction starting from $\mathbf{p}_{i+1}$ and ending at $\mathbf{p}_{i-1}$. While traversing, let $\mathbf{p}_{i'}$ and $\mathbf{p}_{i''}$ be the first and last connected vertices encountered, i.e., $\overline{\mathbf{p}_{i'}\mathbf{q}_{j'}}$ and $\overline{\mathbf{p}_{i''}\mathbf{q}_{j''}}$ are two ruling lines already in *RL*. Then all the ruling lines $\overline{\mathbf{p}_i\mathbf{q}_j};\ j''\le j\le j'$ satisfy constraint 1.

A function $\mathbf{IsValid}(\mathbf{p}_i,\mathbf{q}_j)$ is defined which returns *true* only if $j''\le j\le j'$. The function is so named because each $j(i),\ j''\le j(i)\le j'$, qualifies to represent the discrete version of the valid reparameterization $v_{j(i)}$. Let $\mathbf{R}_i$ be a subset of $\mathbf{Q}$ containing all $\mathbf{q}_j,\ j''\le j\le j'$.

$$\mathbf{R}_i=\{\mathbf{q}_j\in\mathbf{Q}\,|\,\mathbf{IsValid}(\mathbf{p}_i,\mathbf{q}_j)=true\}_{j=0,\ldots,b} \tag{12}$$

Now $MVRL_i$ can be found from the set $\mathbf{V}_i\cap\mathbf{R}_i$, as shown in Figs. 2(c) and 2(d). If $MVRL_i=\overline{\mathbf{p}_i\mathbf{q}_j}$, then, by definition of $MVRL_i$, the condition $f(\mathbf{p}_i,\mathbf{q}_j)=\max\left(\left\{f(\mathbf{p}_i,\mathbf{q}_j)\big|_{\mathbf{q}_j\in\mathbf{V}_i\cap\mathbf{R}_i}\right\}\right)$. The pseudocode of finding $MVRL_i$ is given in Algorithm I below.

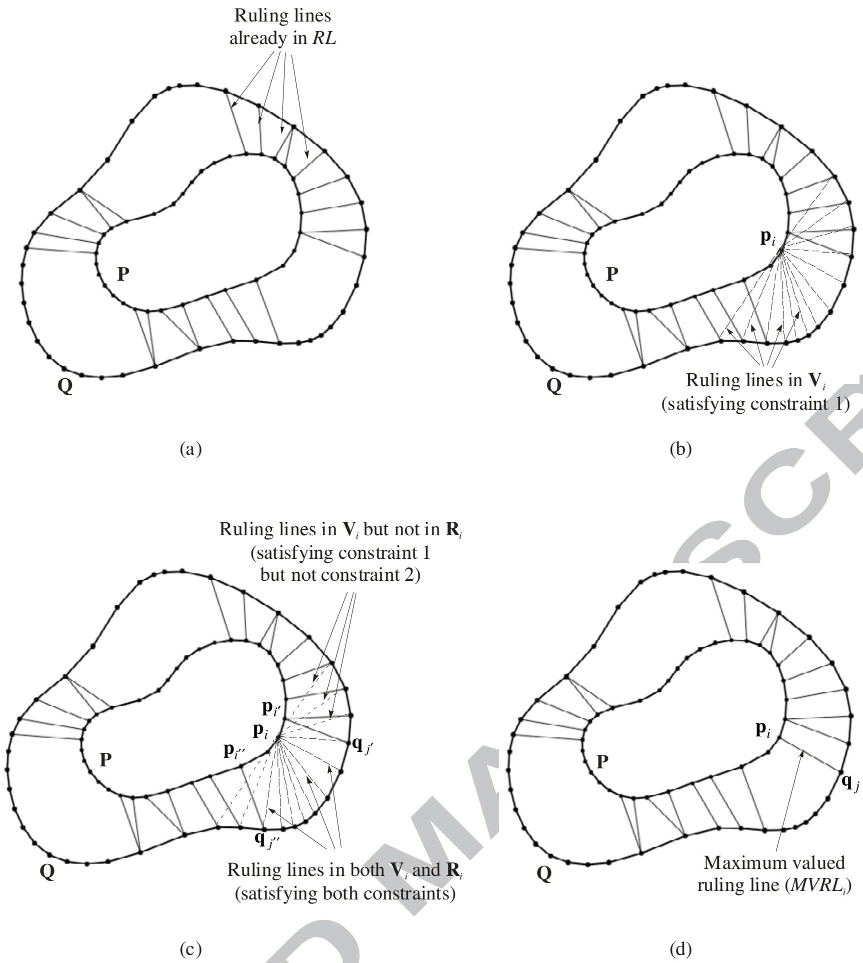


Figure 2: Depiction of a step in the *GRLC* method when the *maximum valued ruling line* ($MVRL_i$ **) =** $\overline{\mathbf{p}_i\mathbf{q}_j}$ is added to *RL*, (a) a step in the *GRLC* method when *RL* already contains a few ruling lines, (b) a visibility example for vertex $\mathbf{p}_i$, (c) finding ruling lines satisfying both constraints 1 and 2, (d) $MVRL_i$ is found and subsequently added to *RL.*

Algorithm I: Finding Maximum Valued Ruling Line

```
INPUT:    $\mathbf{p}_i \in \mathbf{P}$ : A point from P
          $\mathbf{N}(\mathbf{p}_i)$ : Unit normal vector associated with $\mathbf{p}_i$
          $\mathbf{Q} = \{\mathbf{q}_j\}_{j=0,\ldots,b}$ : A set of points sampled from curve $\mathbf{C}_2$ ;
          $\mathbf{N_Q} = \{\mathbf{N}(\mathbf{q}_j)\}_{j=0,\ldots,b}$ : A set of the unit normal vectors associated with each $\mathbf{q}_i \in \mathbf{Q}$
OUTPUT: $\overline{\mathbf{p}_i\mathbf{q}_{j'}}$ : Maximum valued ruling line
Initialize max $\Leftarrow$ lowest_number;
FindMVRL ( $\mathbf{p}_i, \mathbf{N}(\mathbf{p}_i), \mathbf{Q}, \mathbf{N_Q}$ ) {
    FOR (j = 0 to b, j++) {
        IF ( IsVisible( $\mathbf{p}_i, \mathbf{q}_j$ ) and IsValid( $\mathbf{p}_i, \mathbf{q}_j$ )) {
            Calculate value = $f(\mathbf{p}_i, \mathbf{q}_j)$ from Equation (9).
4.          IF (value > max) {
5.                max ⇐ value;
6.                $j' \Leftarrow j$; }}
7.  }  /*** end of for-loop ***/
8.  return $\overline{\mathbf{p}_i\mathbf{q}_{j'}}$ ;
} /*** end of finding maximum valued ruled line ***/
END.
```

Since both polygons are closed, no initial match conditions are specified. As mentioned before, the greedy approach constructs the *RL_candidate_list*. Among all candidates in the *RL_candidate_list*, the one with the maximum value is chosen and added to *RL*. The end vertices of this ruling line are marked as connected. Then the candidate set is emptied and a fresh set is reconstructed excluding all the ruling lines that are already in *RL*. Again the "best" one is chosen from the set and stored in *RL*. This is performed repeatedly until all the vertices of both polygons are connected by at least one ruling line.

Algorithm II: Finding Optimal Set of Ruling Lines

INPUT: $\mathbf{P} = \{\mathbf{p}_i\}_{i=0,\ldots,a}$: A set of points sampled from curve $\mathbf{C}_1$;

$\mathbf{N_P} = \{\mathbf{N}(\mathbf{p}_i)\}_{i=0,\ldots,a}$: A set of the unit normal vectors associated with each $\mathbf{p}_i \in \mathbf{P}$;

$\mathbf{Q} = \{\mathbf{q}_j\}_{j=0,\ldots,b}$: A set of points sampled from curve $\mathbf{C}_2$;

$\mathbf{N_Q} = \{\mathbf{N}(\mathbf{q}_j)\}_{j=0,\ldots,b}$: A set of the unit normal vectors associated with each $\mathbf{q}_i \in \mathbf{Q}$

OUTPUT: $RL = \{\overline{\mathbf{p}_i\mathbf{q}_j}\}$: A set of ruling lines that maximizes the objective function in Equation (6).

Initialize *RL* $\Leftarrow$ *empty*, *end_check* $\Leftarrow$ 0; *RL_candidate_list* $\Leftarrow$ *empty*;

```
RulingLineGeneration $(\mathbf{P}, \mathbf{N_P}, \mathbf{Q}, \mathbf{N_Q})$ {
1.  WHILE (end_check $\neq$ 1) {
2.      RL_candidate_list ⇐ empty;
3.      FOR (i = 0 to a, i++)
4.      {       /*** For a point in P, find the point in Q that has the maximum function value ***/
5.              RL_candidate_list ⇐ RL_candidate_list ∪ (FindMVRL ($\mathbf{p}_i, \mathbf{N}(\mathbf{p}_i), \mathbf{Q}, \mathbf{N_Q}$));
6.      } /*** end of for-loop ***/
7.      $\overline{\mathbf{p}_i\mathbf{q}_j}$ ⇐ max(RL_candidate_ list);        /*** Find the maximum ***/
8.      IF (Last edge in RL and $\overline{\mathbf{p}_i\mathbf{q}_j}$ are same)
9.              THEN {end_check ⇐ 1;}
10.     ELSE
11.             {RL ⇐ RL ∪ $\overline{\mathbf{p}_i\mathbf{q}_j}$ ;}               /*** Insert the maximum valued ruled line into RL ***/
12. } /*** end of while-loop ***/
} /*** end of ruled line generation ***/
END.
```

In each iteration of the while loop from steps 1 to 12 in Algorithm II, exactly one new ruling line is inserted in *RL* and the objective function value increases by the value of the new ruling. When the loop terminates, there are a number of one-to-many connections, i.e., one vertex in **P** is connected to many vertices in **Q** and vice versa. Figure 3 shows the result after Algorithm II is executed on the polygons **P** and **Q**. Since one of the requirements is that each vertex is connected by the minimum number of ruling lines, a few redundant ruling lines need to be eliminated from *RL.*

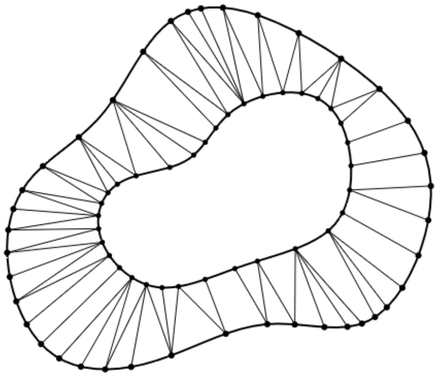


Figure 3: Ruling line set *RL* at end of Algorithm II

To identify the redundant ruling lines, the following terms are defined.

Definition: Degree of a point - the number of ruling lines connected to that point
Definition: Redundant ruling line - a ruling line with both end points having degree ≥ 2

The redundant ruling lines are removed using the following subroutine.

```
/*** Remove redundant ruling lines ***/
numRedundant_rl ⇐FindNumOfRedundantRL(RL);
WHILE (numRedundant_rl  > 0) {
        Remove the lowest valued ruling line from RL;
        numRedundant_rl ⇐FindNumOfRedundantRL(RL);
} /*** end of while-loop ***/
```

When the above subroutine is run on the result obtained at the end of Algorithm II, all redundant ruling lines are identified, as shown in Fig. 4.

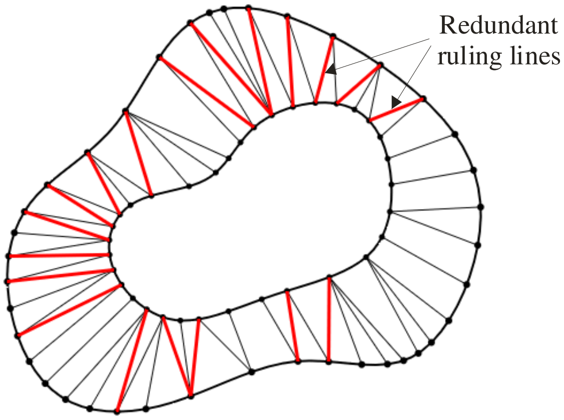


Figure 4: Redundant ruling lines identified

The redundant ruling lines are removed as shown in Fig. 5.

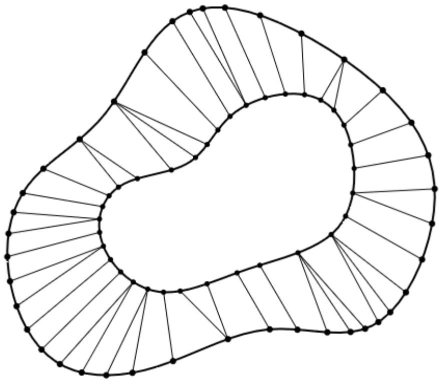


Figure 5: Redundant ruling lines removed

After the redundant ruling lines are removed, one-to-many matching of the vertices can still exist. This happens when the curvatures of the curves differ significantly at the vertices. It is neither intuitive nor visually pleasing that one vertex on one directrix matches with many points on the other directrix. This means that while metamorphosis is occuring, one vertex of the source will metamorphose into an arc on the target and vice versa. Therefore, a vertex insertion method outlined below is developed that "spreads out" the ruling lines so that all ruling lines have one-to-one correspondence. This is achieved by inserting more vertices near the vertex with degree more than one and connecting each of them with new ruling lines.

If $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ are two consecutive vertices on $\mathbf{P}$, both of which may have degrees of more than one. Let, out of all the ruling lines connected to $\mathbf{p}_i$, the line $\overline{\mathbf{p}_i\mathbf{q}_j}$ have the highest function value. Similarly, out of all the ruling lines connected to $\mathbf{p}_{i+1}$, the line $\overline{\mathbf{p}_{i+1}\mathbf{q}_{j+k}}$ has the highest function value. Therefore, the k – 1 points between $\mathbf{q}_j$ and $\mathbf{q}_{j+k}$ have to be detached from their connections on $\mathbf{P}$ and be relocated because the two ruling lines $\overline{\mathbf{p}_i\mathbf{q}_j}$ and $\overline{\mathbf{p}_{i+1}\mathbf{q}_{j+k}}$ are the locally best matches. The points $\mathbf{q}_{j+1},\ldots,\mathbf{q}_{j+k-1}$ were connected to either $\mathbf{p}_i$ or $\mathbf{p}_{i+1}$ because there were no other points available in between.

The insertion of points between $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ is done according to proportional parametric increments of $\mathbf{q}_j$ and $\mathbf{q}_{j+k}$. Let the parameter associated with $\mathbf{q}_{j+x}$ be $v_{j+x}, x = 0,\ldots,k$. Since k – 1 vertices are going to be inserted between $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$, rename $\mathbf{p}_{i+1}$ to $\mathbf{p}_{i+k}$ and let the parameters associated with $\mathbf{p}_i$ and $\mathbf{p}_{i+k}$ be u_i and u_{i+k}, respectively. Then k – 1 points are sampled from $\mathbf{C}_1$ between $\mathbf{p}_i$ and $\mathbf{p}_{i+k}$ as follows.

$$\mathbf{p}_{i+x} = \mathbf{C}_1(u_{i+x}) \qquad \text{where, } u_{i+x} = u_i + (u_{i+k} - u_i)\frac{v_{j+x} - v_j}{v_{j+k} - v_j}; x = 1,\ldots,k-1 \qquad (13)$$

Using the proportional parameter increments,new points are inserted usinga discrete reparameterization between the original points only. In Equation (13), if $v_{j+x+1} > v_{j+x}$ then $u_{i+x+1} > u_{i+x}$ or vice versa. This guarantees no intersection of ruling lines.

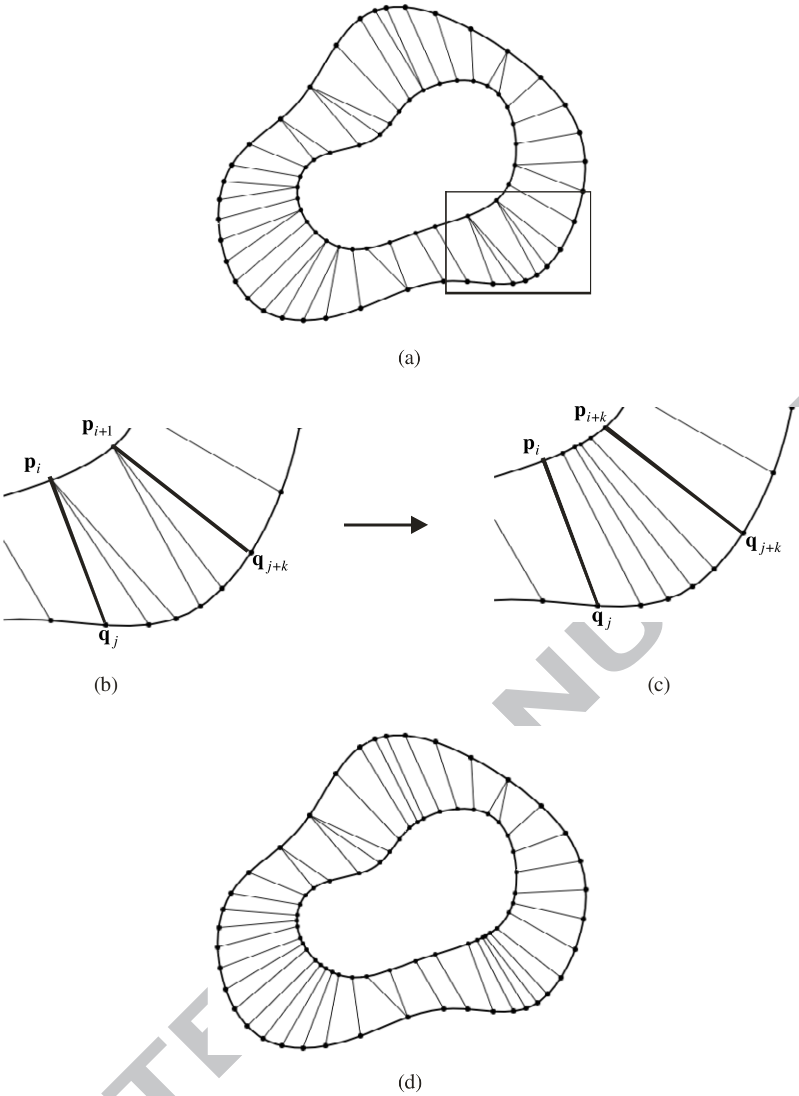


Figure 6: Insertion of points, (a) two consecutive points and the ruling lines connected to them, (b) the ruling lines with highest function value are identified, (c) *k* – 1 points are inserted, here *k* = 5, (d) the ruled line set after the first pass

The vertex insertion is done in two stages. As shown in Fig. 6, the first stage involves marching along $\mathbf{C}_1$ and inserting points using Equation (13). Later, at the second stage, points are inserted on $\mathbf{C}_2$ using the same procedure discussed above.

6. A Dynamic Modeling Approach for Comparison

For comparison purposes, we employ dynamic programing to solve the problem. The optimization model formulated in our earlier work [36] is used for ruling line insertion to obtain optimum matching of two curves for an intuitive metamorphosis. In this formulation, the solution methodology slightly differs from the *GRLC* approach in that each sampled point can be occupied by only one ruling line during the matching process. In addition, a ruling line insertion technique is proposed for non-occupied points after the matching process. Dynamic programming is formulated as in Equation (14), where RL_t stands for the ruling line inserted at stage t and $f_t(\mathbf{p}_i, \mathbf{q}_j)$ is the corresponding objective function at that stage:

$$f_t(\mathbf{p}_i,\mathbf{q}_j)=\max_{i,j}\left\{\frac{\langle \mathbf{N}(\mathbf{p}_i),\mathbf{N}(\mathbf{q}_j)\rangle^{w_n}}{\left|\overrightarrow{\mathbf{p}_i,\mathbf{q}_j}\right|^{w_d}}+f_{t-1}(\mathbf{p}_i,\mathbf{q}_j)\right\} t=2,\ldots,T$$

s.to

(14)

$RL_t:\overline{\mathbf{p}_i\mathbf{q}_j}\cap \mathbf{C}_1(u)=\varnothing\ \forall i,j$ (Visibility Constraint)

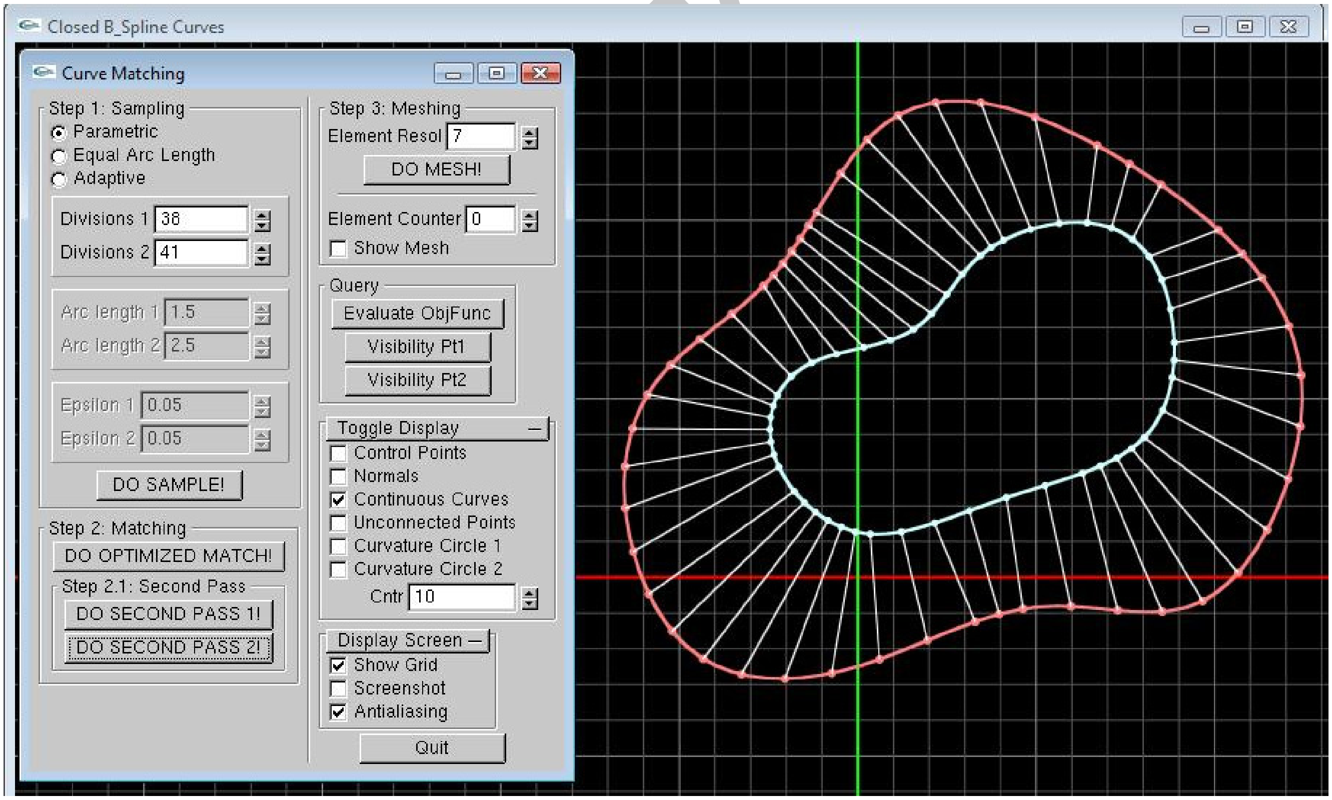
$RL_t:\overline{\mathbf{p}_i\mathbf{q}_j}\cap \mathbf{C}_2(v)=\varnothing\ \forall i,j$ (Visibility Constraint)

$RL_t:\overline{\mathbf{p}_i\mathbf{q}_j}\cap RL_m=\varnothing\ \forall i,j,m:m=1,\ldots,t-1$ (Ruling Line Intersection Constraint)

Combining the above-mentioned three constraints, a sub-algorithm is developed to check if it is feasible to insert a ruling line during curve matching [36]. A ruling line in this work is feasible as long as it satisfies both cases. A ruling line at stage *t* is not considered as a candidate for insertion if it is not feasible.

7. Implementation and examples

The proposed GRLC methodology is implemented in Microsoft Visual C++. OpenGL library functions are used for visualization. GLUI library functions are used for the user interface with numerous point-sampling options, such as parametric, equal arc length or adaptive sampling, and display features, such as unconnected points, normal vectors, and control points (see Fig. 7).



Closed B_Spline Curves

Curve Matching

Step 1: Sampling
● Parametric
○ Equal Arc Length
○ Adaptive
Divisions 1: 38
Divisions 2: 41
Arc length 1: 1.5
Arc length 2: 2.5
Epsilon 1: 0.05
Epsilon 2: 0.05

Step 2: Matching

Step 2.1: Second Pass

Step 3: Meshing
Element Resol: 7

Element Counter: 0
□ Show Mesh

Query

Toggle Display
□ Control Points
□ Normals
☑ Continuous Curves
□ Unconnected Points
□ Curvature Circle 1
□ Curvature Circle 2
Cntr: 10

Display Screen
☑ Show Grid
□ Screenshot
☑ Antialiasing

Figure 7: Final ruled line set after two passes of point insertion. Each ruling line represents one-to-one matching between the directrices.

Figure 7 shows the user interface with the results after optimum matching and the two stage insertions. Curve matching is performed in 3.44 sec with an objective function of 3.50 for 44 ruling lines.

Moreover, the visual appearance of the ruling line set is pleasing and intuitive. The developed *GRLC* approach in this paper is compared with the *k*-shift approach proposed in the literature [9]. In Fig. 8, both directrices, $\mathbf{C}_1$ and $\mathbf{C}_2$, are uniformly parameterized so that 44 vertices are generated from each curve. The vertices are connected one-to-one parametrically to generate the ruling lines, as shown in Fig. 8(a). The result is a twisted ruled surface, and the value of the objective function is a low 1.18. *k*-shift is utilized as a naïve approach to untwist the surface, the ruling lines are shifted by a constant number of vertices, and the objective function value is noted. The highest objective function value observed is 3.01 for a 4-vertex shift, as shown in Fig. 8(b). This is less than the result of the proposed *GRLC* technique shown in Fig. 7. Furthermore, the *k*-shift approach results in local untwisting on a ruled surface and hence does not guarantee global untwisting.

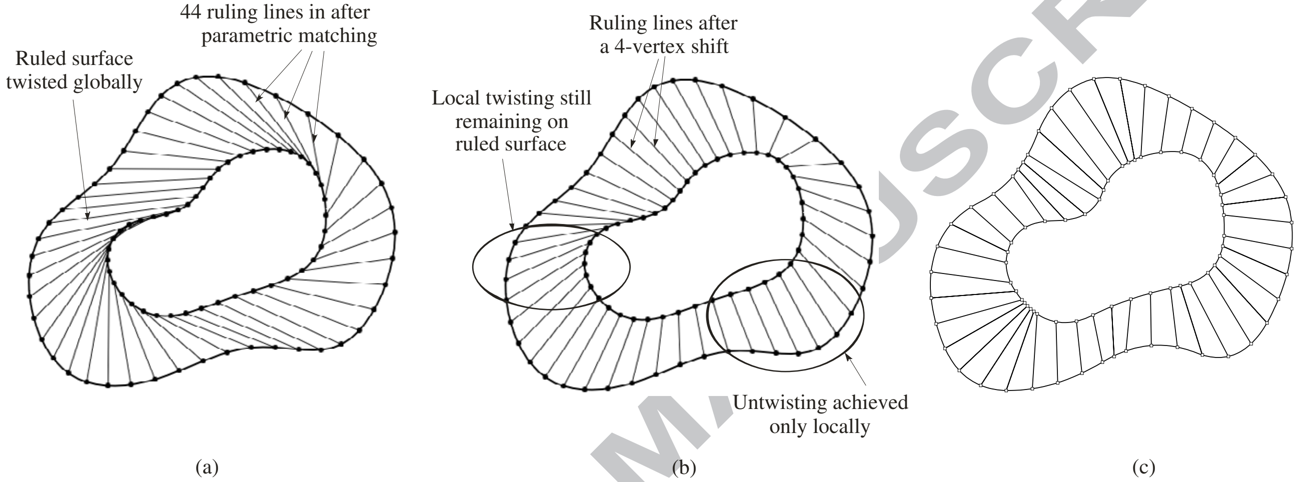

44 ruling lines in after parametric matching
Ruled surface twisted globally
Ruling lines after a 4-vertex shift
Local twisting still remaining on ruled surface
Untwisting achieved only locally
(a)
(b)
(c)

Figure 8: Comparison of results of *GRLC* with parametric matching, (a) globally twisted ruled surface produced, (b) global untwisting is not guaranteed by *k*-shift of ruling lines, and (c) optimum matching using dynamic programming.

The proposed heuristic *GRLC* approach is tested by comparing it with the dynamic programming proposed in our earlier study [36], in which 44 ruling lines were generated on the same curves shown in Fig. 8(c). Although the optimum result of 3.63 is slightly greater than the objective function obtained in *GRLC* (3.5% error), its computation time is 7.94 sec, which is more than double that of the *GRLC* approach.

Furthermore, the developed heuristic *GRLC* approach is compared with the commercial software Rhinoceros [37] to test its fidelity. A concave U-shape internal curve is selected for this purpose. The *GRLC* approach generates the non-intersecting and pleasing set of ruling lines shown in Fig. 9(a); however, Rhinoceros generates distorted results at the edges of the inner U-shape curve with the intersecting ruling lines illustrated in Fig. 9(b).

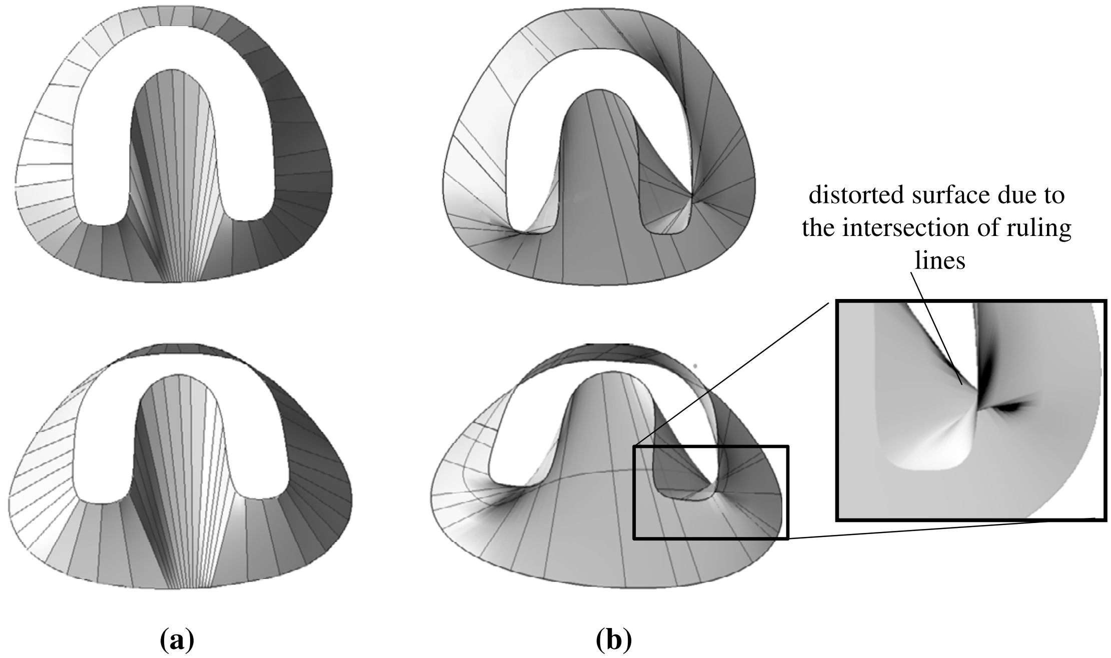


Figure 9: Ruled surface generation using the complex curves $\mathbf{C}_1$ and $\mathbf{C}_2$ using (a) heuristic *GRLC* approach and (b) commercial Rhinoceros software [37]

The directrices can be considered as key curves, and a series of metamorphosis curves (iso-property curves) for heterogeneous object modeling can be generated between them by using the ruled lines. Figure 10 shows an example where the ruling line set is divided into five uniform segments (material blending).

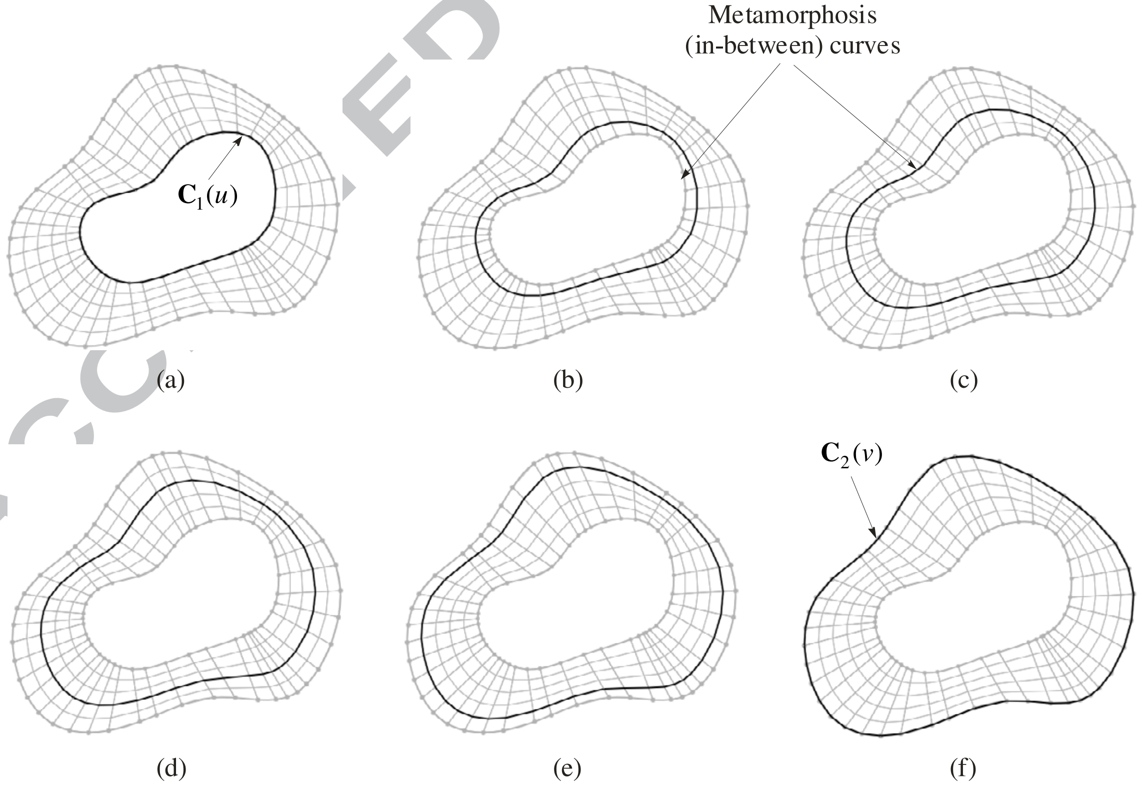


Figure 10: Material-blending iso-condition curves between the curves $\mathbf{C}_1$ and $\mathbf{C}_2$ using a ruled line set with $M = 5$ and $j = 0, \ldots, 5$, (a) $j = 0$ is same as the first generator curve, (b) $j = 1$, (c) $j = 2$, (d) $j = 3$, (e) $j = 4$, (f) $j = M$, is the same as the second generator curve

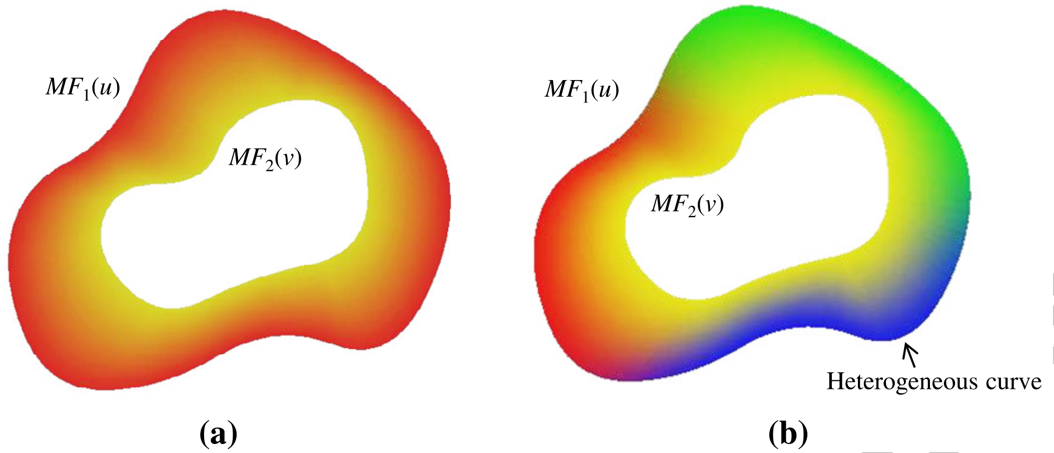


Figure 11: Application of curve matching in the domain of heterogeneous object modeling and visualization: (a) material blending between two homogenous curves, and (b) material blending between a heterogeneous and a homogenous curve

Figure 11 shows an application of the curve-matching technique for heterogeneous object modeling and visualization purposes. Material properties of two homogeneous directrices are blended to construct a heterogeneous object as shown in Fig. 11(a). Material properties are mapped over the unit cells generated by the ruling lines and in-between curves shown in Fig. 10. Mapping heterogeneous composition on one of the directrices, however, increases the complexity of the problem as demonstrated in Fig. 11(b). In this case, smoother blending is crucial for the accurate representation of the material composition. The material composition in Fig. 11(b) varies in multiple directions: (i) variation in the metamorphosis direction (see Fig. 10) and (ii) variation along the trajectory of the heterogeneous curve. A large number of ruling lines is needed to avoid any sudden sharp changes in the material composition, otherwise material properties such as material incompatibilities and stress concentrations may worsen. A large number of ruling lines, on the other hand, results in a higher computation time, which the *GRLC* approach alleviates significantly.

Tissue engineering and regenerative medicine have great potential in the application of objects with composition heterogeneity in the future. For instance, synthetic engineered implantable tissues and organs should have functional gradients to mimic their original counterparts in addition to their conforming geometries; this has been extensively studied in our earlier works [28, 38-40]. Development of multi-functional tissue scaffolds with superior control over cell-biomaterial-protein interactions might be an engineering challenge and a promising approach for improved and efficient tissue regeneration. Recently, we developed a pressure-assisted multi-material printing system for heterogeneous tissue scaffold fabrication [39]. Following that, a new multi-material toolpath plan was proposed to enable continuous material deposition [41]. In this work, ruling lines were used to construct the toolpath in two consecutive layers: the first layer with a ruling-line-based zigzag pattern and the next layer with a spiral pattern deposition. The new toolpath in this work [41] follows the metamorphosis direction and prevails on the traditional Cartesian toolpath in terms of accurate representation of the heterogeneity. Figure 12(a) shows the application of the multi-material toolpath planning for a femur bone STL model obtained using ITK-Snap software [42]. For a random slice with a femoral artery (see Fig. 12(b)), the toolpath is obtained as in Fig. 12(c) in 19 sec using the *GRLC* approach with 64 ruling lines. Dynamic programming also generates intuitive matching; however, the computation time turns out to be 37 sec. Although the computation time may not be too significant for toolpath generation on a single slice, generation of the entire model with a great number of slices makes a tremendous difference in the computation time.

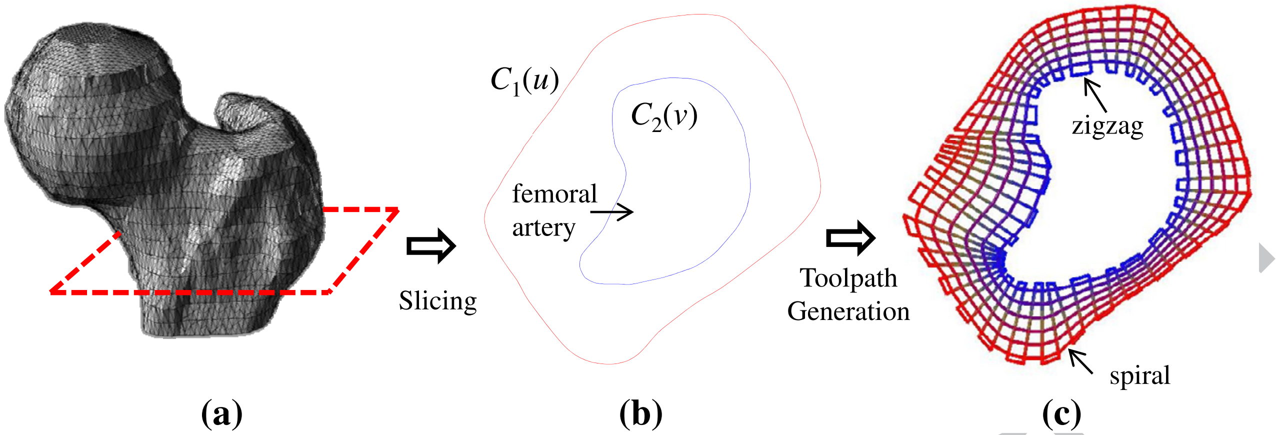


Figure 12: Application of curve matching in the domain of heterogeneous tissue scaffold fabrication: (a) a STL femur bone model, (b) slicing the model around the femoral artery and generation of a multi-material toolpath using the proposed *GRLC* approach

The proposed *GRLC* approach in this paper generates near optimum results as presented using different cases. The optimality conditions of the GRLC are to have continuous contours and not to have sudden changes between matching contours. The other condition is discontinuities of the normal vectors of the curves, such as sharp corners. In this case, the model generates acceptable results, but they are not as appealing as the optimum condition. However, in real applications, the changes between contours will not be abrupt for smooth objects. The presented methods only work for matching of two contours at a time. If one-to-many contour matching is required, the proposed method may not work. In that case, a Voronoi diagram based algorithm could be used [43].

9. Conclusions

In this paper, a new method is proposed to generate matching lines between two material governing features (directrices) for heterogeneous object modeling. Each connecting line represents a match between two points on the directrices. A new method, the Greedy Ruling Line Construction (*GRLC*), is developed to match the directices such that their material features (properties) are blended to each other along their matched normal direction. The *GRLC* method generates non-self-intersecting and non-twisted connecting material governing features. It generates near optimum results and reduces computation time dramatically. By subdividing the connecting lines into an equal number of segments, a series of intermediate iso-property curves are obtained to represent the material metamorphosis between the governing material features. Furthermore, the *GRLC* approach generates blending with nonintersecting ruling lines for complex shapes, while the top commercial software in surface modeling fails to do so. The proposed method deserves particular attention in heterogeneous object modeling with smooth material gradients in addition to curve-matching-based continuous toolpath planning for rapid prototyping. The method can be easily adapted for non-closed and non-planar curves as well. The developed methods can also be used in several other applications, such as getting a smooth transition between two given 2D curves for animation, shape blending, morphing, approximation of 3D solid object models from planar curves, and multi-axis NC machining of ruled surfaces.

References

1. Samanta, K. and B. Koc, *Feature-based design and material blending for free-form heterogeneous object modeling*. Computer-Aided Design, 2005. **37**(3): p. 287-305.
2. Choi, B.K., *Surface Modeling for CAD/CAM*. 1991, New York: Elsevier.

3. Peternell, M., H. Pottmann, and B. Ravani, *On the computational geometry of ruled surfaces.* Computer-Aided Design, 1999. **31**(1): p. 17-32.
4. Elber, G., *Metamorphosis of freeform curves and surfaces*, in *Computer Graphics Developments in Virtual Worlds*, R.A. Earnshaw and J.A. Vince, Editors. 1995, Academic Press Ltd. p. 29-40.
5. Surazhsky, T. and G. Elber, *Metamorphosis of planar parametric curves via curvature interpolation.* International Journal of Shape Modeling, 2002. **8**(2): p. 201-216.
6. Sederberg, T.W. and E. Greenwood, *A physically based approach to 2–D shape blending.* ACM SIGGRAPH Computer Graphics, 1992. **26**(2): p. 25-34.
7. Sederberg, T.W. and E. Greenwood, *Shape blending of 2-D piecewise curves*, in *Mathematical methods for curves and surfaces*, M. Daehlen, T. Lyche, and L. Schumaker, Editors. 1995, Vanderbilt University Press. p. 497-506.
8. Hui, K.C. and Y. Li, *A feature-based shape blending technique for industrial design.* Computer-Aided Design, 1998. **30**(10): p. 823-834.
9. Cohen, S., G. Elber, and R. Bar-Yehuda, *Matching of freeform curves.* Computer-Aided Design, 1997. **29**(5): p. 369-378.
10. Johan, H., Y. Koiso, and T. Nishita. *Morphing using curves and shape interpolation techniques*. in *Proceedings of Pacific Graphics 2000, 8th Pacific Conference on Computer Graphics and Applications*. October 2000. Hong Kong (China).
11. Wang C. and, Tang, K. "Optimal boundary triangulations of an interpolating ruled surface", *ASME Journal of Computing and Information Science in Engineering*, **5**(4), pp. 291-301, 2005.
12. Koc, B. and Y.-S. Lee, *Adaptive ruled layers approximation of STL models and multiaxis machining applications for rapid prototyping.* Journal of Manufacturing Systems, 2002. **21**(3): p. 153-166.
13. Marciniak, K., *Geometric Modelling for Numerically Controlled Machining*. 1992, New York: Oxford University Press.
14. Bohez, E.L.J., et al., *A geometric modeling and five-axis machining algorithm for centrifugal impellers.* Journal of Manufacturing Systems, 1997. **16**(6): p. 422-436.
15. Yang, B., M.C. Leu, and J. Zhou, *Error analysis for four-axis side milling of undevelopable ruled surfaces.* Control Engineering Practice, 1998. **6**(4): p. 481-487.
16. Tsay, D.M. and M.J. Her, *Accurate 5-axis machining of twisted ruled surfaces.* Journal of Manufacturing Science and Engineering, 2001. **123**: p. 731-738.
17. Bedi, S., S. Mann, and C. Menzel, *Flank milling with flat end milling cutters.* Computer-Aided Design, 2003. **35**(3): p. 293-300.
18. Chiou, J.C.J., *Accurate tool position for five-axis ruled surface machining by swept envelope approach.* Computer-Aided Design, 2004. **36**(10): p. 967-974.
19. Monies, F., et al., *Five-axis NC milling of ruled surfaces: optimal geometry of a conical tool.* International Journal of Production Research, 2004. **40**(12): p. 2901-2922.
20. Senatore, J., et al., *Analysis of improved positioning in five-axis ruled surface milling using envelope surface.* Computer-Aided Design, 2005. **37**(10): p. 989-998.
21. Gong, H., L.-X. Cao, and i. Liu, *Improved positioning of cylindrical cutter for flank milling ruled surfaces.* Computer-Aided Design, 2005. **37**(12): p. 1205-1213.
22. Piegl, L. and W. Tiller, *The NURBS Book*. 2nd ed. 1997, New York: Springer.
23. Guenter, B. and R. Parent, *Computing the arc length of parametric curves.* IEEE Computer Graphics and Applications, 1990. **10**(3): p. 72-78.
24. Farin, G. and N. Sapidis, *Curvature and the fairness of curves and surfaces.* IEEE Computer Graph and Applications, 1989. **9**(2): p. 52-57.
25. Liu, D.S. and D.Y. Chiou, *Modeling of inclusions with interphases in heterogeneous material using the infinite element method.* Computational Materials Science, 2004. **31**(3-4): p. 405-420.

26. Liu, D.S., D.Y. Chiou, and C.H. Lin, *3D IEM formulation with an IEM/FEM coupling scheme for solving elastostatic problems.* Advances in Engineering Software, 2003. **34**(6): p. 309-320.
27. Xu, A. and L.L. Shaw, *Equal distance offset approach to representing and process planning for solid freeform fabrication of functionally graded materials.* Computer-Aided Design, 2005. **37**(12): p. 1308-1318.
28. Ozbolat, I.T., Marchany, M., Bright, F.V., Cartwright, A.N., Gardella Jr., J.A., Hard, R., Hicks, Jr. W.L., and Koc, B., *Feature-based Design of Bio-degradable Micro-patterned Structures*, Journal of Computer-Aided Design & Applications, 2009. **6**(5): p. 661-671.
29. Kale, S.M., *Feature-based bio-mimetic modeling of heterogeneous scaffolds for improved tissue engineering*. Master Thesis, University at Buffalo, 2009.
30. Lim, C.T., Ensz, M.T.,Ganter, M.A. and Storti, D.W., *Object Reconstruction from layered data using implicit solid modeling,* Journal of Manufacturing Systems, 1997. **16**(4): p. 260-272.
31. Marsan, A., Dutta, D., *Computational techniques for automatically tiling and skinned branched objects,* Computers & Graphics, 1999. **23:** p. 111-126.
32. Bajaj, C.L., Coyle, E.J., and Lin, K.-L., *Arbitrary Topology Shape Reconstruction from Planar Cross Sections*, Graphical Models and Image Processing, 1996. **58**(6): p. 524-543.
33. Meek, D.S., and Walton, D.J. *Blending two parametric curves*, Computer-aided Design, 2009.**41**: p.423-431.
34. Marker, J., Braude, I., Museth, K., and Breen, D., *Contour-Based Surface Reconstruction using Implicit Curve Fitting, and Distance Field Filtering and Interpolation, International Workshop on Volume Graphics and Applications*. July 2006. Boston, MA.
35. Kühnel, W., *Differential Geometry-Curves-Surfaces-Manifolds*. 2nd ed. 2005, Wiesbaden: AMS.
36. Ozbolat, I.T., *Heterogeneous Tissue Scaffolds for Spatiotemporally Controlled Release Kinetics*. PhD Thesis, University at Buffalo, 2011.
37. Rhinoceros 4.0, http://www.rhino3d.com, Robert McNeel and Associates.
38. Khoda, A.K.M.B., Ozbolat,I.T., and Koc, B., *Engineered Tissue Scaffolds With Variational Porous Architecture.* Journal of Biomechanical Engineering, 2011. **133**(1): p. 011001.
39. Ozbolat, I.T. and Koc, B., *Modeling of Spatially Controlled Biomolecules in Three-Dimensional Porous Alginate Structures*, ASME Transactions Journal of Medical Devices, 2010. **4**(4), p.041003.
40. Ozbolat, I.T. and Koc, B., *Multi-function Based Modeling of 3D Heterogeneous Wound Scaffolds for Improved Wound Healing*, Computer-aided Design and Applications, 2011. **8**(1), p.43-57.
41. Ozbolat, I.T. and Koc, B., *A new Toolpath Planning in solid Freeform Fabrication for Hollowed Scaffolds, Proceedings of Industrial Engineering Research Conference.* May 2011, Reno, Nevada.
42. ITK-Snap, http://www.itksnap.org, 2008.
43. Ozbolat, I.T. and Koc, B., *Multi-directional Blending for Heterogeneous Objects*, Computer-aided Design Journal, 2011. 4**8**(3), p.863-875.

Highlights:

- A matching algorithm was developed to optimize connecting between curves.
- Curves are matched to max. normal directions and min. the distance between them.
- A dynamic programming approach has also been developed and compared.
- Examples for heterogeneous object modeling and tissue engineering are presented.