Matching of Freeform Curves *

Shmuel Cohen, Gershon Elber, and Reuven Bar-Yehuda
Department of Computer Science
Technion, Israel Institute of Technology
Haifa 32000
Israel
Email: {shmuelc|gershon|reuven}@cs.technion.ac.il

Abstract

Freeform parametric curves are extensively employed in various fields such as computer graphics, computer vision, robotics, and geometric modeling. While many applications exploit and combine univariate freeform entities into more complex forms such as sculptured surfaces, the problem of a fair or even optimal *relative* parameterization of freeforms, under some norm, has been rarely considered.

In this work, we present a scheme that closely approximates the optimal relative matching between two or even n given freeform curves, under a user's prescribed norm that is based on differential properties of the curves. This matching is computed as a reparameterization of $n-1$ of the curves that can be applied explicitly using composition. The proposed matching algorithm is completely automatic and has been successfully employed in different applications with several demonstrated herein: metamorphosis of freeform curves with feature preservations, key frame interpolation for animation, self intersection free ruled surface construction, and automatic matching of rail curves of blending surfaces.

Key Words: Dynamic Programming, Tangent/Gauss Map, Feature Recognition, Fairness.

1 Introduction

The piecewise polynomial or rational parametric forms have gained a paramount position as a representation of choice in many applications of computer graphics, geometric modeling, and vision. Many freeform parametric surface constructors are based on univariate primitives. Both the ruled and the blend or fillet surface constructors expect two curves to operate on. It is also common to approximate or interpolate a surface through several curves, two operations that are known as skinning or sweeping.

In this work, we consider the *fairness* of two or more different curves. The notion of fairness has been extensively used a for single freeform curve. Fairness of the shape of a curve has been considered by minimizing, for example, a thin plate spline functional [12]. One instance of a fairness consideration is a fair parameterization, or a parametrization that carries certain properties such as constant speed or even arc length [7], a parameterization that is important, for example, for constant speed motion in animation

*This work was supported in part by grant No. 92-00223 from the United States-Israel Binational Science Foundation (BSF), Jerusalem, Israel. However, opinions, conclusions or recommendations arising out of supported research activities are those of the author or the grantee and should not be presented as implying that they are the views of the BSF.

or in NC machining. Many other fairness criteria were considered in the representation of a single freeform Bspline curve. See [12, 15].

Nevertheless, and while freeform curves are frequently employed in conjunction with other freeform curves, the question of their relative fairness has been rarely considered. We distinguish between two fundamentally different fairness problems,

Definition 1 *Given a freeform curve, C, the* intra-fairness *of C considers the fairness of C with respect to some optimization function. Given n freeform curves, C_i, the* inter-fairness *of C_i considers the* relative *fairness of curves C_i with respect to some optimization function.*

One example to inter-fairness of piecewise linear curves can be found in [2, 3], using the distance norm between two curves in order to *match* the two curves. In [2], the Hausdorff metric is exploited. This metric does not establish a one to one correspondence and in [3], the authors of [2] present the Frechet metric that alleviates this lack of correspondence, and examine the application of "walking the dog". The notion of "walking the dog" is derived from seeking the minimal length of the leash that is required, when one is walking on one curve while its dog is walking on the other curve. The need to match several different piecewise linear curves emerged more than a decade earlier in [10] when a polygonal surface was sought to fit contoured medical data. The work of [10] as well as [21] employ toroidal graphs, a two dimensional data structure to allow the representation of matching of two discrete ordered sets. In this work, we would like to exploit a similar data structure for the continuous case of freeform curves by employing differential properties of the curves. The toroidal graph is employed in previous work in a greedy approach that does not guarantee the detection of the globally optimal path while herein dynamic programming is used for the same task.

In the area of vision [4, 19], matching plays a major role in the recognition of both two dimensional planar shapes as well as three dimensional objects. Typical vision techniques [4] map the geometrical matching problem into the matching of graphs, with a search for isomorphisms in graphs, employing decision and search trees or other pruning methods. In [19], dynamic programming is exploited to match piecewise linear contours at a sub pixel resolution, allowing for matching of vertices to interior locations on edges of the other contour. In [16, 22], dynamic programming is employed for the purposes of matching of discrete sequence in various applications such as signal processing speech recognition and DNA matching in biology.

We would like to examine the inter-fairness of n freeform curves in a larger framework that does not necessarily employ the distance norm. A relative reparameterization will be computed for $n-1$ curves, so as to optimally match the entire set of n curves, under the user's prescribed norm, in a global sense. The simplest case of $n = 2$, or only two curves, will be of particular interest, for its frequent occurance. Consider the example of Figure 1, starting with the planar ruled surface in Figure 1 (a). Clearly, the ruled surface is self intersecting. Nevertheless, by reparametrizing either one of the two boundary curves, it is possible to prevent the self intersection. In Figure 1 (b), two rail curves [9] are selected to smoothly blend together two different regions using cubic Hermite interpolation, with C^1 continuity, following [14]. While the average user should not be occupied with the parameterization of the two rail curves, the resulting blend surface clearly twists, an undesired artifact that is resolved manually in contemporary systems, possibly by the non trivial process of reparametrizing one of the curves. Finally, in Figure 1 (c), two outlines of a desk, $C_1(t)$, and a turtle, $C_2(t)$, are metamorphed in the plane using the affine combination,

$$C(t) = (1-s)C_1(t) + sC_2(t), \; 0 \leq s \leq 1, \tag{1}$$

with shared features such as their legs vanishing only to reappear in a different parametric location. Equation (1) constructs a bivariate ruled surface between $C_1(t)$ and $C_2(t)$ that is parameterized by both t and s. Parameterization as well as translational and rotational invariance and feature preservation are crucial components for an intuitive and appealing self intersection free metamorphosis. The metamorphosis computed in Figure 1 (c) is neither feature preserving nor is it rotational nor parameterization invariance.

The question of appealing metamorphosis of shapes has had the focused attention of numerous researchers, in recent years. Recognizing the significant difficulties in the general freeform metamorphosis, attempts were made to resolve the shape shifting in simpler domains such as piecewise linear approximations [11, 17, 18, 20]. Two fundamental issues need to be addressed. The first, known as the correspondence problem, matches the source and target entities. The second consideration is the inbetweening interpolation method that is employed, bridging between the source and target shapes. Here, we take a step toward the construction of an automatic procedure for an appealing direct metamorphosis of freeform curves a feasible option, by providing a method to automatically compute the correspondence, if such exists.

In this work, we present a matching scheme of freeform curves that produces in the three examples of Figure 1, a non self intersecting, twist free, and feature preserving metamorphose, respectively. While we present only a few applications for the proposed algorithm, the introduced inter-fairness matching scheme can be equally and successfully applied to many other applications such as matching warped animation curves [23] or reconstruction of freeform contoured data following [10].

This paper is organized as follows. Section 2 restates the problem of matching of freeform curves as a continuous optimization problem. In Section 3, a discrete approximation of the continuous optimization problem is employed. We demonstrate how one can overcome the problems in the three examples given in this section in Figure 1 as well as present more examples in Section 4. Finally, we conclude in Section 5.

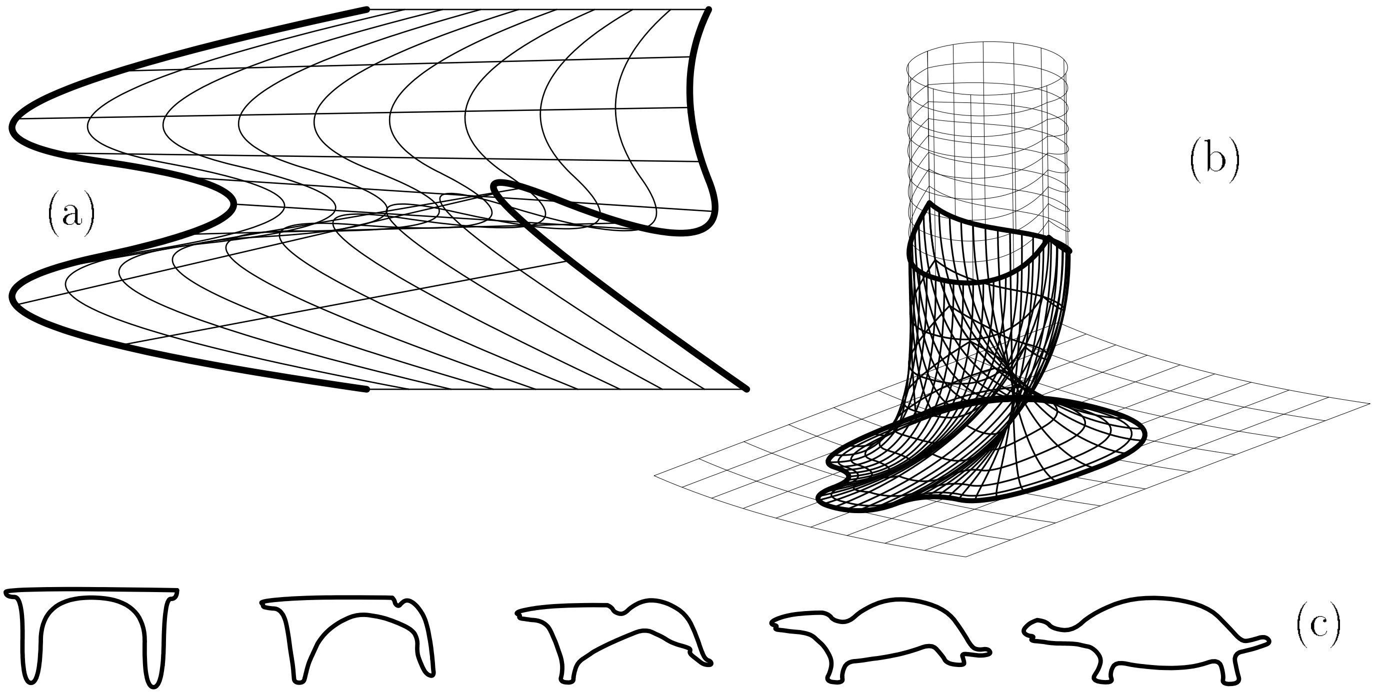


Figure 1: Three examples of a mismatch between curves that creates undesired artifacts. In (a), a planar ruled surface self intersects due to uncorrelated relative parameterization. In (b), the twist in the three dimensional blend surface (thick lines) between the two three dimensional rail curves (thickest lines) is clearly uncalled for. In (c), features like the legs of both the desk and the turtle disappear only to reappear at a different (parametric) location. Compare with Figure 7.

All examples in the paper are the result of an implementation of the proposed algorithm in the solid modeling environment IRIT [13] developed at the Technion. This algorithm was also successfully implemented using the Alpha_1 [1] modeling environment developed at the University of Utah.

2 Background

Consider $\mathcal{S}$, the shape of an engraving on a large planar stone. A plane can approximate $\mathcal{S}$ as closely as the depth of the engravings, considering only the distance norm, while completely losing the engraving. The distance norm is frequently exploited for the purpose of matching of piecewise linear curves [2, 3, 10, 21]. The preservation of higher order differential properties of the shape while approximating $\mathcal{S}$ can alleviate some of these difficulties. For a surface approximating $\mathcal{S}$, the preservation of the Gauss map [5] is sometimes crucial for a successful approximation. The Gauss map preserves the normal field of $\mathcal{S}$, the cross product of the first order derivatives of $\mathcal{S}$, while ignoring the magnitude of the derivatives, which is highly parameterization dependent.

Let $C_1(u)$, $u_0 \leq u \leq u_1$ and $C_2(v)$, $v_0 \leq v \leq v_1$ be two regular C^1 parametric curves in $\mathbb{R}^n$, $n > 0$, that is $\left|\left|\frac{dC_i(u)}{dt}\right|\right| > 0$. Let,

$$T_1(u) = \frac{\frac{dC_1(u)}{du}}{||\frac{dC_1(u)}{du}||}, = \frac{C_1'(u)}{||C_1'(u)||}, \qquad T_2(v) = \frac{\frac{dC_2(v)}{dv}}{||\frac{dC_2(v)}{dv}||} = \frac{C_2'(v)}{||C_2'(v)||},$$

be the unit tangent fields of $C_1(u)$ and $C_2(u)$.

Consider the inner product of $\langle T_1(u), T_2(v)\rangle$ (See Figure 2). If $\langle T_1(u_0), T_2(v_0)\rangle = 1$, $T_1(u_0)$ and $T_2(v_0)$ are parallel and in fact they are equal. Assume two allowable changes of parameters $\mathcal{U} : t \mapsto u$ and $\mathcal{V} : t \mapsto v$ that yield regular parameterizations [5], that is $\mathcal{U}'(t), \mathcal{V}'(t) > 0$. Then,

Definition 2 *if* $\langle T_1(\mathcal{U}(t)), T_2(\mathcal{V}(t))\rangle = 1$, $\forall t_0 \leq t \leq t_1$, *we we say that a* complete match *of the tangent maps has been established for that domain.*

If we assume a complete match for $C_1(\mathcal{U}(t))$ and $C_2(\mathcal{V}(t))$ throughout the domain of t, substitute $C_1(\mathcal{U}(t))$ and $C_2(\mathcal{V}(t))$ into Equation (1), and differentiate with respect to t,

$$\begin{aligned} C'(t) &= (1-s)\frac{C_1(u)}{dt} + s\frac{C_2(v)}{dt} \\ &= (1-s)\frac{C_1(\mathcal{U}(t))}{dt} + s\frac{C_2(\mathcal{V}(t))}{dt} \\ &= (1-s)T_1(\mathcal{U}(t))||C_1'(u)||\mathcal{U}'(t) + sT_2(\mathcal{V}(t))||C_2'(v)||\mathcal{V}'(t) \\ &= (1-s)T_1(t)||C_1'(u)||\mathcal{U}'(t) + sT_1(t)||C_2'(v)||\mathcal{V}'(t) \\ &= \Big((1-s)||C_1'(u)||\mathcal{U}'(t) + s||C_2'(v)||\mathcal{V}'(t)\Big)\, T_1(t), \end{aligned} \tag{2}$$

where we use $T_1(t) = T_1(\mathcal{U}(t)) = T_2(\mathcal{V}(t))$ since the two unit vectors are equal in a complete match. Hence, in a linear interpolation between two completely matched curves, for each s_0 between zero and one, the intermediate curve

$$C(t, s_0) = (1-s_0)C_1(\mathcal{U}(t)) + s_0C_2(\mathcal{V}(t)),\ 0 \leq s \leq 1,$$

is also completely matched with the two curves, $C_1(t)$ and $C_2(t)$.

Let $T(t)$ be the unit tangent field of $C(t)$, in Equation (2). Then, it is clear that $T(t) = T_1(t) = T_2(t)$. Armed with this motivation, one can define the following function to be maximized over all reparametrizations of $C_2(v)$ of the form $v(u) = \mathcal{V}(\mathcal{U}^{-1}(u))$,

$$\max_{v(u)} \int_{u_0}^{u_1} \langle T_1(u), T_2(v(u))\rangle\, du, \qquad v(u_0) = v_0 \quad \text{and} \quad v(u_1) = v_1, \tag{3}$$

subject to $v(u)$ being an allowable change of parameter. In some applications, like [23] where back and forth motion is constructed using animation curves, non regular curves might be important. However, from now on we only consider the case of regular curves which is far more frequent. The reparameterization of $C_2(v)$,

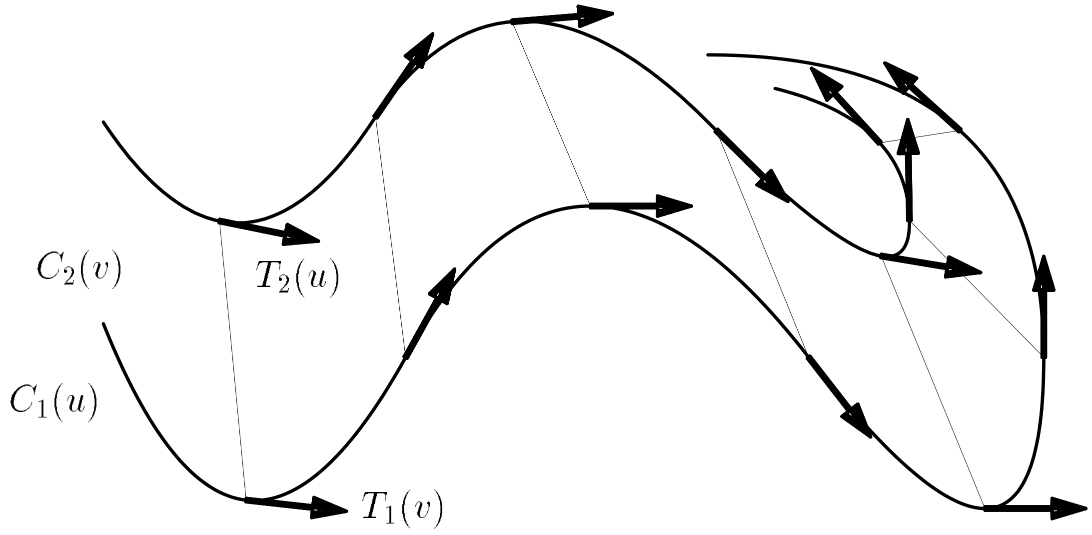


Figure 2: By matching the directions of the tangent fields of two curves, freeform features like legs can be preserved in operations like ruling, sweeping, or morphing.

while asymmetric, can be equally applied to $C_1(u)$, reversing the roles of $C_1(u)$ and $C_2(v)$. Hence, hereafter, we consider only $v(u)$.

We consider two types of self intersection in the affine combinations,

Definition 3 *An affine combination of the form of Equation (1) is said to be* locally self intersection free *if*

$$C(\tau) = (1-s)C_1(\tau) + sC_2(\tau), \ 0 \leq s \leq 1, \quad \tau \in [t-\epsilon, t+\epsilon]$$

is self intersection free for all s and t values in the domain, and a sufficiently small ϵ.

In other words, a locally self intersection free affine combination guarantee that $C(\tau+\epsilon) \neq C(\tau)$, for a sufficiently small ϵ.

Nonetheless as will be shortly demonstrated, a locally self intersection free affine combination can create intersecting curves, globally,

Definition 4 *An affine combination of the form of Equation (1) is said to be* globally self intersection free *if*

$$C(t) = (1-s)C_1(t) + sC_2(t), \ 0 \leq s \leq 1$$

is self intersection free for all s and t values in the domain.

Clearly, globally self intersection free is more strict, yet and unsurprisingly, it is more difficult to achieve.

Under certain considerations, we will require $\langle T_1(u), T_2(v(u))\rangle > 0, \ \forall u$, a less strict constraint than that of a complete match,

Definition 5 *if $\langle T_1(u), T_2(v(u))\rangle > 0, \ \forall u_0 \leq u \leq u_1$, we we say that a* valid match *of the tangent maps has been established for that domain.*

It is clear that it is not always feasible for $\langle T_1(u), T_2(v(u))\rangle$ to be positive throughout. Consider, for example, two parallel parametric lines with one line with a reversed parameterization. While failing to achieve a valid match, one might maximize the prescribed global function of Equation (3), in a hope that the resulting invalid match will at least lead to an improved match, compared to the original, unmatched, curves. Moreover, in Section 4, we will present several refinements to the proposed techniques that will result in a possibly complete match in cases such as the reversed parameterization.

A valid match can help in the prevention of local self intersection in the metamorphosis process between $C_1(t)$ and $C_2(t)$,

Lemma 1 *The affine combination of* $(1-s)C_1(u) + sC_2(v(u))$, $0 \le s \le 1$ *is locally self intersection free provided that* $v(u)$ *is a valid match with* $\langle T_1(u), T_2(v(u))\rangle > 0$.

Proof: Because both curves are C^1, a first order approximation of $C_1(u)$ and $C_2(v(u))$ at $u = u_0$ can be written as $C_1(u) = C_1(u_0) + C_1'(u_0)u$ and $C_2(u) = C_2(v(u_0)) + C_2'(v(u_0))|v'(u_0)|u$. Then,

$$C(u) = (1-s)C_1(u) + sC_2(v(u)) \approx (1-s)(C_1(u_0) + C_1'(u_0)u) + s(C_2(v(u_0)) + C_2'(v(u_0))|v'(u_0)|u).$$

Now consider the projection of $C(u)$ on vector $C_1'(u_0)$, $\langle C(u), C_1'(u_0)\rangle$. Because $\langle C_1'(u_0), C_2'(v(u_0))\rangle > 0$, the value of this projection is increasingly monotone as a function of u, in the local neighborhood of u_0. Hence, a local self intersection cannot occur at a sufficiently small neighborhood of $C(u_0)$. ■

Corollary: By maximizing Equation (3) self intersections in metamorphosis of freeform curves can be alleviated and locally eliminated, provided a valid match is established.

We will demonstrate the capabilities of the maximized function in the coming sections. The result is indeed self-intersection free in the local if the match is valid because then both curves advances forward with respect to each other. While holding for an arbitrary dimensionality, Lemma 1 is unable to guarantee a self intersection free planar ruled surface, in $\mathbb{R}^2$, as it cannot prevent a curve at $s = s_0$ from intersecting a curve at $s = s_1$. Consider the example of Figure 3. While $\langle T_1(u), T_2(v(u))\rangle > 0$ throughout, the ruled surface self intersects. The following modified function for the case of self intersection prevention in planar ruled surface constructors is employed instead,

$$\max_{v(u)} \int_{u_0}^{u_1} \frac{\langle T_1(u) \times (C_1(u) - C_2(v(u))), T_2(v(u)) \times (C_1(u) - C_2(v(u)))\rangle}{||\,(C_1(u) - C_2(v(u)))\,||^2} du, \quad v(u_0) = v_0 \text{ and } v(u_1) = v_1, \tag{4}$$

subject to $v(u)$ being an allowable change of parameter. The motivation for the function of Equation (4) stems from the need to coerce both planar curves $C_1(u)$ and $C_2(v)$ to advance into the same half-plane as is

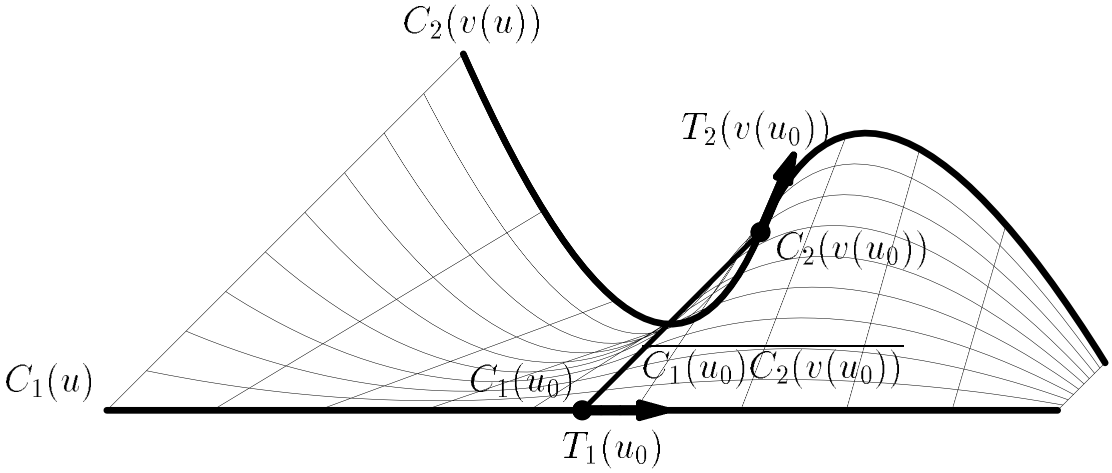


Figure 3: While preserving features, the function of Equation (3) cannot prevent from self intersection in freeform planar ruling operations, as can be seen in this figure. While $\langle T_1(u_0), T_2(v(u_0))\rangle > 0$ throughout, and no intermediate curve intersects with itself, the planar ruled surface does self intersect. A different function (Equation (4)) must be employed.

determined by the ruling line $\overline{C_1(u)\ C_2(v(u))}$ (See Figure 3). The expression of $T_1(u) \times (C_1(u) - C_2(v(u)))$ will point in the negative or positive Z direction depending upon the direction of vector $T_1(u)$ with respect to the two half spaces of the XY plane split by the line through $C_1(u) - C_2(v(u))$. Then, the self intersection free constraint can be reformulated as the need for $\langle T_1(u) \times (C_1(u) - C_2(v(u))), T_2(v(u)) \times (C_1(u) - C_2(v(u)))\rangle$ to be positive, resulting in $T_1(u)$ and $T_2(u)$ pointing into the same half plane. The denominator of the term inside the integral in Equation (4), is a normalization factor only.

Definition 6 *A point of* mutual tangency *in a match of two curves $C_1(u)$, and $C_2(v(u))$ satisfies either $(C_1(u) - C_2(v(u)))||T_1(u)$ or $(C_1(u) - C_2(v(u)))||T_2(v(u))$,*

where || denotes the parallel constraint.

We are now ready to formulate the conditions for locally no self intersection in the ruled surface,

Lemma 2 *If $\langle T_1(u_0) \times (C_1(u_0) - C_2(v(u_0))), T_2(v(u_0)) \times (C_1(u_0) - C_2(v(u_0)))\rangle > 0$ the ruled surface of $R(s,u) = (1-s)C_1(u_0 \pm u) + sC_2(v(u_0 \pm u))$, $|u| < \epsilon$ is self intersection free, for a sufficiently small ϵ.*

Proof: Because both curves are C^1, a first order approximation of $C_1(u)$ and $C_2(v(u))$ at $u = u_0$ can be written as $C_1(u) = C_1(u_0) + C_1'(u_0)u$ and $C_2(u) = C_2(v(u_0)) + C_2'(v(u_0))|v'(u_0)|u$. Because, the four corners of $C_1(u_0) - C_1'(u_0)\delta$, $C_2(v(u_0)) - C_2'(v(u_0))|v'(u_0)|\delta$, $C_1(u_0) + C_1'(u_0)\delta$, $C_2(v(u_0)) + C_2'(v(u_0))|v'(u_0)|\delta$ define a non self intersection quadrilateral and since the ruled surface $R(s,t)$ converges to this quadraliteral as $\epsilon \to 0$, $R(s,t)$ will not self intersect for a sufficiently small ϵ. ■

Special care should be taken at points of mutual tangency where the function of Equation (4) vanishes to zero, rendering the constraint of Lemma 2 unsatisfied.

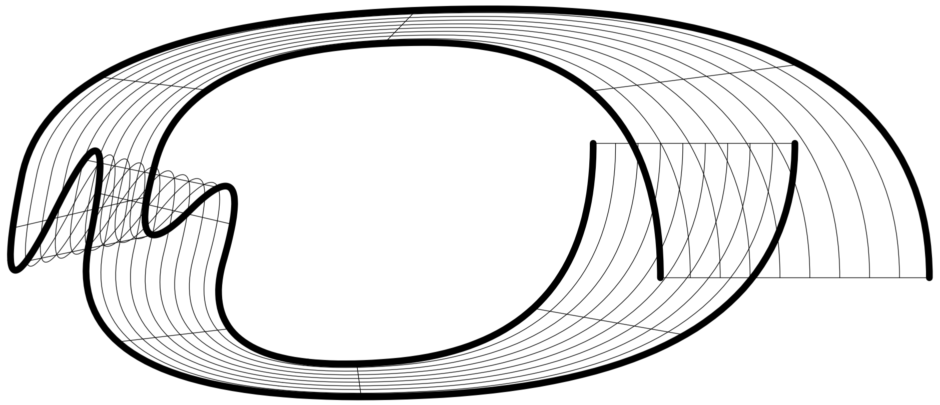


Figure 4: A planar ruled surface might be self intersection free locally, satisfying Lemma 2 at all points except the points of mutual tangency on the left side. The self intersection in the left is a result of a mutual tangent point. On the right, the self intersection occurs when the two end regions overlap. In both cases, it is impossible to construct a self intersection free ruled surface. This entire scene is planar.

Intuitively, Lemma 2 states that as long as both planar curves advance in the same half plane that is defined using the line $\overline{C_1(u)\,C_2(v(u))}$, the ruling will be regular without any self intersection in the neighborhood of $C_1(u_0)$ and $C_2(v(u_0))$. One might argue that if Lemma 2 holds for all points in the parametric domain of the curve, the ruled surface will never self intersect in the global. Nevertheless, disjoint regions of the ruled surface might self intersect as is demonstrated in Figure 4.

The constraint imposed by Equation (4) is stronger than the constraint imposed by Equation (3). Equation (3) and Lemma 1 guarantee that no blended curve $C(t)$ in Equation (1) will locally self intersect. Equation (4) and Lemma 2 also guarantee that planar curve $C(t)$ at blended value s_0 of Equation (1) will not intersect planar curve $C(t)$ at blended value s_1 of Equation (1), $0 \leq s_0 < s_1 \leq 1$.

There exist pairs of curves that no reparameterization can be found that will satisfy the constraints of either Lemma 1 or Lemma 2 for the entire parametric domain. The parallel lines with a reversed parametrization is one example we have already encountered. However, if there is a reparameterization that satisfies and/or maximizes either function, one would like to be able to find and compute it. Further, it is desired to be able to find the optimal parameterization under the prescribed function out of the space of all reparameterizations with a valid match. In Section 3, a discrete space solution that provides an approximation to these optimality goals is considered for the continuous functions of (3) and (4). This discrete solution is then mapped back to a piecewise polynomial reparameterization curve, $v(u)$, that may be symbolically composed with $C(v)$.

3 Algorithm

While it is clear that an analytic solution to either optimization problems of Equations (3) or (4) is too difficult in the general case, one can provide an approximated solution in polynomial time complexity exploiting dynamic programming over the discrete sample sets of the two curves and their unit tangent vector fields. This approximation can be made more precise by increasing the size, m, of the discrete approximation present in this section.

Let C_1^i, $0 \le i < m$, and C_2^j, $0 \le j < m$, be two sequences of m uniform samples in the parametric spaces of $C_1(u)$ and $C_2(v)$, respectively. Let T_1^i, $0 \le i < m$, and T_2^j, $0 \le j < m$, be two sequences of m unit tangent vectors similarly sampled along $C_1(u)$ and $C_2(v)$, respectively. Consider the two problems over the discrete matching $j(i)$ of,

$$\max_{j(i)} \sum_{i=0}^{m-1} \left\langle T_1^i, T_2^{j(i)} \right\rangle, \qquad j(0) = 0 \quad \text{and} \quad j(m-1) = m-1, \tag{5}$$

$$\max_{j(i)} \sum_{i=0}^{m-1} \frac{\left\langle T_1^i \times \left(C_1^i - C_2^{j(i)}\right), T_2^{j(i)} \times \left(C_1^i - C_2^{j(i)}\right)\right\rangle}{||C_1^i - C_2^{j(i)}||^2}, \qquad j(0) = 0 \quad \text{and} \quad j(m-1) = m-1, \tag{6}$$

subject to $j(i) \le j(i+1)$. These two optimization problems are discrete approximations of the continuous optimization problems defined in Equations (3) and (4). The monotonicity constraint $j(i) \le j(i+1)$ is equivalent to the monotonicity constraint on $v(u)$, making it an allowable change of parameter in Equations (3) and (4).

The discrete optimization problems of Equations (5) and (6) can now be efficiently solved:

Definition 7 *Let $Cost(i,j)$ be the optimal cost of matching the first i samples of $C_1(t)$ with the first j samples of $C_2(t)$.*

Assume that $Cost(i,j) = 0$ for $i = j = -1$ and $Cost(i,j)$ approaches infinity for all other cases for which $i < 0$ or $j < 0$. This initial condition forces the selection of first edge to be from $i = 0$ to $j = 0$. Denote the (i,j) entry of the matrix of all possible combinations of inner products, Inner Product Matrix, by $IPM(i,j)$. $IPM(i,j) = \left\langle T_1^i, T_2^j \right\rangle$ if employing Equation (5), or $IPM(i,j) = \frac{\left\langle T_1^i \times \left(C_1^i - C_2^j\right), T_2^j \times \left(C_1^i - C_2^j\right)\right\rangle}{||C_1^i - C_2^j||^2}$ employing Equation (6). Then,

Lemma 3

$$Cost(i,j) = \min\left(Cost(i-1,j-1), Cost(i-1,j), Cost(i,j-1)\right) + IPM(i,j), \tag{7}$$

Proof: By induction. $Cost(0,0)$ is clearly equal to $IPM(0,0)$ since $Cost(-1,-1)$ is equal to zero. Now for the step of the induction, assume that $Cost(i-1,j-1)$, $Cost(i-1,j)$, and $Cost(i,j-1)$ are all known.

When computing $Cost(i, j)$, the i'th edge of C_1^i must be connected to the j'th edge of C_2^j, contributing $IPM(i, j)$ to the cost. There are three possibilities for the previous edge to edge (i, j): $(i-1, j-1)$, $(i-1, j)$ or $(i, j-1)$. Of these three possibilities the minimum is selected. ■

Direct recursive solution of Equation (7) is exponential. However, Equation (7) can be clearly computed more efficiently,

Lemma 4 *Cost(m-1,m-1) can be computed in a quadratic time complexity of* $O(m^2)$

Proof: By employing a dynamic programming solution to $Cost(m-1, m-1)$. The Inner Product Matrix has $O(m^2)$ entries, each can be computed in a constant time (the inner product). With the IPM available, a $Cost$ matrix can be computed row by row, from left to right as each element $Cost(i, j)$ depends on only three of its neighbors $Cost(i-1, j-1)$, $Cost(i-1, j)$, and $Cost(i, j-1)$, that are already computed, resulting in $O(m^2)$ computations. ■

In Figure 5, a simple example for $m = 4$ and discrete function (5) is provided. In Figure 5 (a), the IPM matrix is precomputed. In Figure 5 (b), the Cost matrix is derived with the optimal path in bold. Finally, in Figure 5 (c), the final and optimal match for the discrete problem is presented. The number of edges in the established match can not exceed $2m-1$. Inspecting the matrix of Figure 5 (b), the established path is formed of motion edges in the right, down, or down-right diagonal directions only, starting from the top left corner of the matrix all the way to the bottom right corner of the matrix. The shortest path will include only the diagonal motions with $j(i) = i$ with m edges in all. The longest path will include m down edges following by $m-1$ right edges, $2m-1$ edges in all. Hence, the maximal cost possible for an optimal match is less than $2m$, since the value of the inner product functions in both Equations (5) and (6) cannot exceed one. At invalid locations for which $\left\langle T_1^i, T_2^j \right\rangle < 0$ (Equation (5)) or $\frac{\langle T_1^i \times (C_1^i - C_2^j), T_2^j \times (C_1^i - C_2^j) \rangle}{\|C_1^i - C_2^j\|^2} < 0$ (Equation (6)), we set $IPM(i, j)$ to $2m$, signaling an *invalid match*. In Figure 5 (b), $2m$ is 8. As a result, if at the end of the dynamic programming process, $Cost(m-1, m-1) \geq 2m$, then no valid match, with positive inner products only, exists.

Once the discrete optimal match, $j(i)$, has been computed, one needs to reconstruct a continuous reparameterization function $v(u)$ as in Equations (3) and (4) from $j(i)$ (Equations (5) and (6)). $j(i)$ is a discrete approximation of $v(u)$ that is not even one to one. Hence, one can interpolate or approximate $v(u)$ from $j(i)$ using an interpolation or approximation method that preserves the monotonicity.

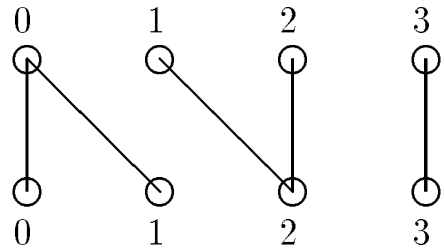
A least squares approximation approach that is based on piecewise polynomial Bspline representations has been exploited [12]. The approximation does a least squares fit on the $j(i)$ data, creating the piecewise polynomial Bspline approximation of $v(u)$. The fact that $j(i)$ is not even one to one stems from the

	0	1	2	3
0	**0.1**	8	0.3	0.7
1	**0.9**	8	0.9	0.2
2	0.1	**0.3**	**0.5**	8
3	8	0.2	8	**0.4**

(a)

	-1	0	1	2	3
-1	0	8	8	8	8
0	8	**0.1**	8.1	8.3	8.7
1	8	**1**	8.1	9.0	8.5
2	8	1.1	**1.3**	**1.8**	9.8
3	8	9.1	1.3	8.3	**2.2**

(b)



(c)

Figure 5: The Inner Product Matrix (IPM) (a) for $m = 4$ is employed to compute the optimal discrete matching cost (b) and its associated optimal path (c). In **bold**, is the optimal path.

fact that several edges in the discrete approximation of the first curve can be connected to one point in the discrete approximation of the second curve and vice versa. because, $v(u)$ must be a continuous and monotone function, the interpolation problem of k different j's for the same i is converted into a sequence $v(u_l) = v_l,\ l = 0, ..., k-1$ with the u_l values close to each other, $0 < |u_l - u_{l-1}| < \epsilon$.

The control polygon is then verified to be monotone, making sure $v(u)$ is an allowable change of parameter, by the variation diminishing property. The loss of monotonicity can occur regardless of the degree of the least squares fit, yet it was found to be less severe as the reduction rates increase. Nevertheless, the error introduced by the correction of a non monotone curve into a monotone one can be clearly bound by the maximal perturbation of the control points.

Hereafter, we refer to this least squares fit as a *reduction* because the number of control points in the Bspline curve is typically much smaller than the discrete match $j(i)$. The higher the order of the least squares that is selected, the smoother the computed reparameterization is, since a higher level of continuity is gained. Because $v(u)$ can only affect the speed and parametric continuity of $C_2(v)$ but not its shape and geometric continuity, orders of $v(u)$ as low as two had very little effect on the final result, in all tested applications. Figure 6 (a) shows one example of a reparameterization curve $v(u)$, derived from the discrete approximation $j(i)$.

With the computed function $v(u)$, the explicit composition of $C_2(v) \circ v(u) = C_2(v(u))$ constructs a new, possibly higher order, curve with the exact same geometry as $C_2(v)$, but with a different first order derivative or speed. If both $C_2(v)$ and $v(u)$ are piecewise polynomials or rationals as is the case at hand, the resulting composition will also be a piecewise polynomial or rational, respectively.

In [6, 14], methods to compute the symbolic composition of univariate functions represented as polynomial or rational Bézier and Bspline curves are discussed. The algorithmic approach of [14] for the explicit computation of the composition was exploited in this work. Appendix A provides a short description of the composition of two Bézier curves. Finally, it should be recalled that the degree of a composition operation

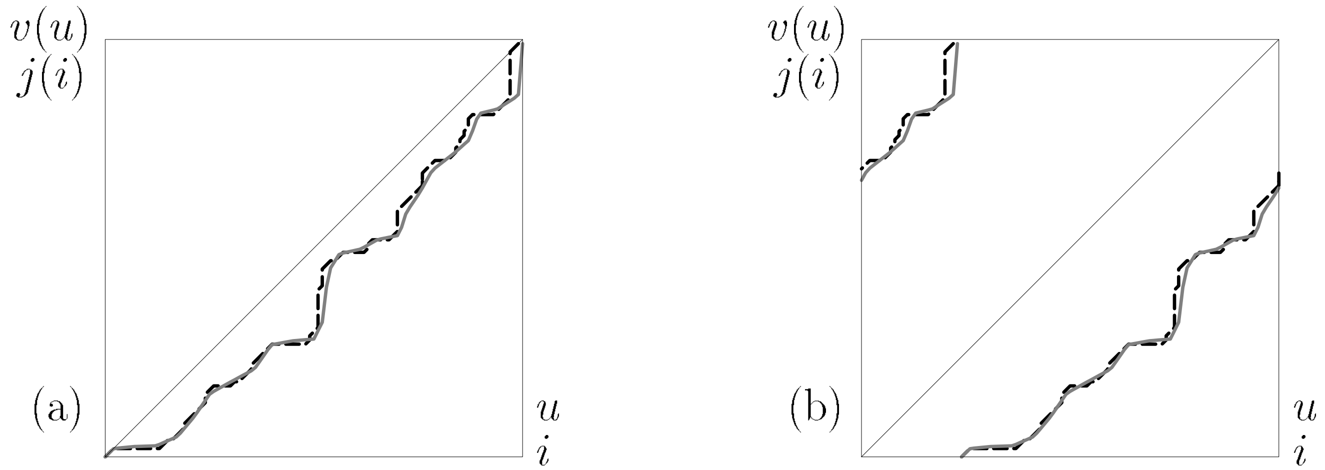


Figure 6: In (a), a piecewise linear reparameterization curve with 30 control points, $v(u)$, in gray, for $C_2(v(u))$, is least squares reduced from an optimal discrete solution with 143 sample points, $j(i)$, in dashed black, of a discrete optimization problem of $m = 100$ samples. In (b), the same example demonstrates the possible result when shifting is employed.

of two curves is the product of the degrees of the two composed curves. if one of input curves is linear, the degree of the result will equal the degree of the other input curve. The resulting curve, even with $v(u)$ being linear, is likely to experience an increase in the size of the resulting curve.

Several examples that employ the proposed inter-fairness matching algorithm with both optimization functions (3) and (4) are presented in Section 4.

4 Examples

In Figure 1, three examples of artifacts resulting from improper relative parameterization between curves were presented. The function of Equation (6) was used to derive a proper reparameterization, $v(u)$, to $C_2(v)$ so no self intersection can occur in the planar ruled surface. The corrected ruled surface is shown in Figure 7 (a). The same matching approach was also exploited to remove the undesired twist shown in the naive Hermite blend of Figure 1 (b). The two rail curves are in three dimensional general positions and the matching of the tangent field is successfully performed in three space. Moreover, nothing in the presented approach prevents the matching from taking place in $\mathbb{R}^3$ or even $\mathbb{R}^n$, for any n, with one exception - the function to optimize must support the dimensionality of the problem. The matching for morphing function (Equation (3)) can clearly match two curves in $\mathbb{R}^n$ while the matching for self intersection free ruled surface function (Equation (4)) is only valid in $\mathbb{R}^2$. The expected blend is automatically derived, requiring no knowledge of the internal parametric representation from the user. See Figure 7 (b). In Figure 1 (c), an improper parameterization has led to an obviously unappealing metamorphosis between two freeform curves. The metamorphosis has been computed explicitly as an affine symbolic combination of the source

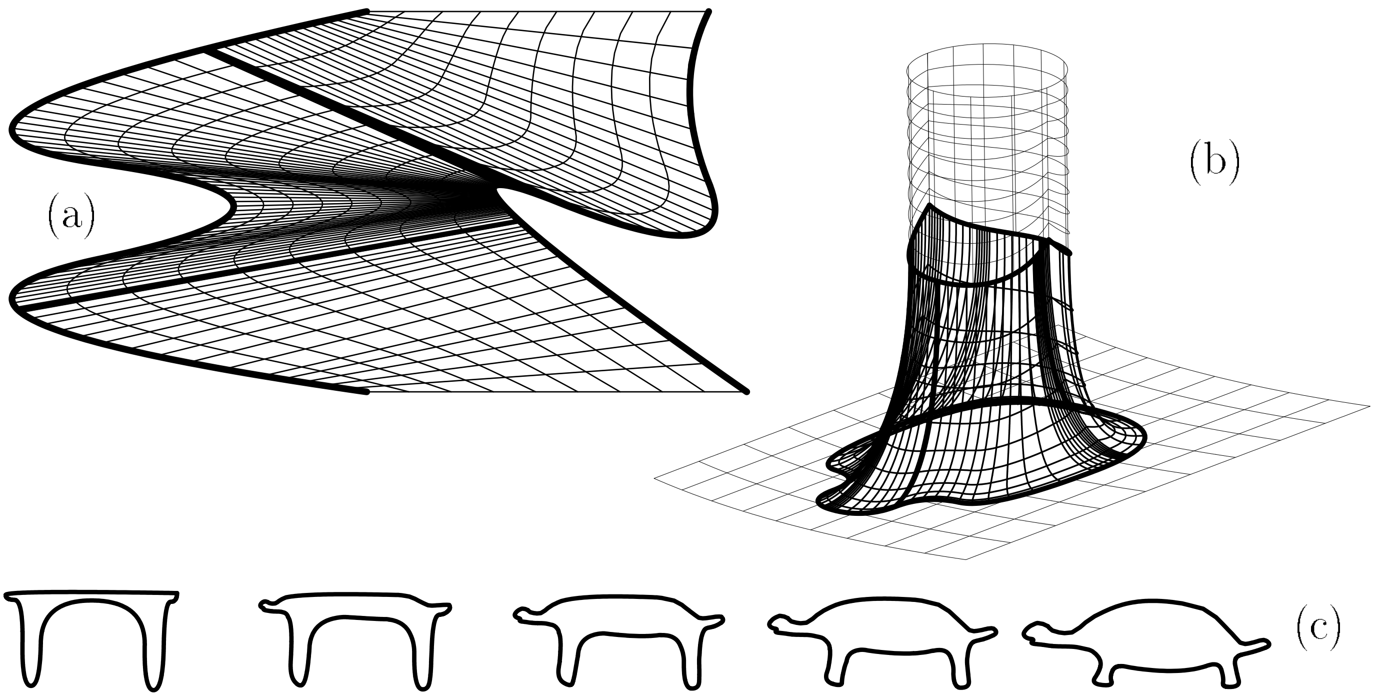


Figure 7: Three examples of an automatically recovered match between freeform curves (Compare with Figure 1). In (a), the ruled surface does not self intersect any more due to the introduced reparameterization computed using the discrete function of Equation (6). In (b), the twist has been completely eliminated with the aid of the reparameterization of $v(u)$, computed with shifting and using Equation (6). In (c), the legs of the desk and the turtle as well as other features are completely preserved throughout the metamorphosis process using the function of Equation (5).

and target curves following Equation (1), once $C_1(t)$ and $C_2(t)$ are brought to the same function space [8]. With the matching computation employing Equation (5) and applying the computed reparameterization using composition, one is able to drastically improve the result that is presented in Figure 1 (c), employing the same affine combination of Equation (1). See Figure 7 (c).

This matching algorithm has been extensively tested on a variety of metamorphosis examples. See Figures 8, 9, 11, 12, and 13 for several more metamorphosis sequences. Several additional optimization techniques were applied in these examples to the basic matching algorithm that was presented in Section 3. We have already considered two parallel parametric lines but with one with a reversed parameterization. Clearly, by reversing the parameterization of one of the curves, one can converge to a much better match. Hence, we apply the discrete matching algorithm twice, once with the given parameterization and once with one of the curves with a reversed parameterization.

In Figure 9, the letter E is metamorphed to the letter F, following [17]. Both letters are quadratic Bspline curves with 24 (for the E) and 20 (for the F) control points. The letter E was interactively derived from the letter F by modifying the control polygon of the F, hence changing the correspondence between the

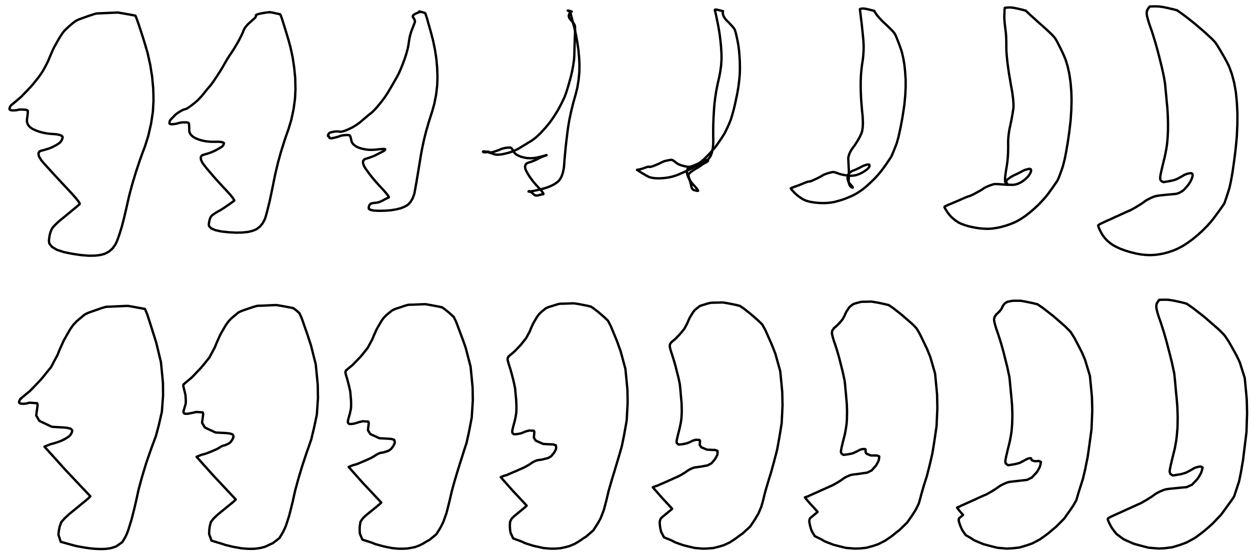


Figure 8: The matching algorithm can be employed when the two blended shape are quite different geometrically. The naive metamorphosis of (a) does quite poorly compared with the results after the matching computation in (b). Notice the preservation of the mouth feature in (b) using discrete matching $i(j)$ with m = 100 reduced to a piecewise linear $v(u)$ with 20 coefficients.

two characters for the entire domain of the letters, as seen in Figure 9 (a). The matching algorithm that is presented is able to completely compensate for this mismatch, resulting in a much better metamorphosis that is seen in Figure 9 (b). Figure 10 also shows the two three dimensional ruled surfaces describing the metamorphosis process of the letters, before and after the matching process.

Consider two linear curves, one is horizontal and one is vertical. While the curves are identical up to rigid motion, tangent fields are invariant to translation only. Hence, one can rotate the curve $C_1(u)$, so that T_1^i is aligned with a tangent of curve $C_2(v)$, T_2^j, where (i, j) is assumed to be in the output sequence of the discrete matching algorithm. A second additional optimization technique can be applied to closed curves. While for open curves the end points of the source curve must match the end points of the target curve, this constraint must be relaxed for closed curves. Therefore, given the two sequences of m tangent vectors sampled from two closed curves, we introduce an arbitrary *k-shift* of the tangents, $T_1^i \rightarrow T_1^{i+k}$, in a cyclic mode where $T_1^{m-k} \rightarrow T_1^0$. While this extension requires one to test for an optimal solution over all $0 \leq k < m$ and hence increases the expected computational complexity to $O(m^3)$, we eliminate by this generalization the algorithmic dependency on the artificial location of the end points of the specific parametric representation of the closed curve. Moreover, combined together, the rotation and the shifting generalizations allow one to find the best relative orientation of the two curves, $C_i(t)$, making the matching almost independent to rigid motion, with accuracy that is up to the discrete nature of the approximation. It should be recalled that complete rigid motion independence can be achieved only for a complete match, and as the quality of the match is decaying, the rotational independence is lost. Figure 6 (b) shows an example of the constructed

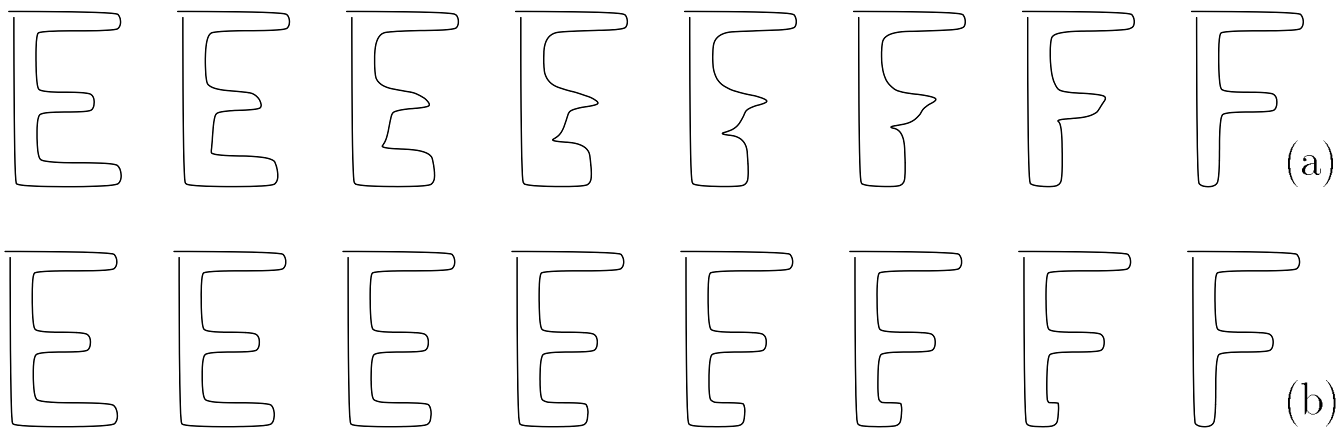


Figure 9: A matching of the letter E to the letter F, both represented as quadratic Bspline curves. In (a), naive metamorphosis is shown, while in (b) the metamorphosis is computed after the matching computation and reparameterization. See also Figure 10.

reparameterization after shifting is applied.

The proposed matching algorithm is not only capable of eliminating self intersection in the local for metamorphosis applications but is also parameterization independent due to the shift optimization and is translational invariant since only first order differential properties participate in the match. Furthermore, in practice, the matching algorithm can be made rotationally invariant for similar shapes, because of the rotational optimization that can be applied. Finally, features are clearly detected and preserved using the matching of the tangent fields, in the provided examples.

In Figure 11, two different parameters of the proposed algorithm are compared. The number of tangent vectors sampled along the curve, m, and the degree of the least squares reduced reparameterization curve $v(u)$. The lowest degree reparameterization curve $v(u)$ (Figure 11 (b)) performs better than the higher order $v(u)$ (Figure 11 (c)), possibly because the higher order approximation filters out necessary high resolution details in the speed. Further, insufficient degrees of freedom manifested as too low value of m (Figure 11 (d)) can also lead to unsatisfactory result. Significant deficiencies in degrees of freedom can be detected by the algorithm when it fails to find a valid match between two curves, when one exists. Nevertheless, in Figure 11 (d), a valid match has successfully been computed while the result which is obviously better than Figure 11 (a) is not as appealing as one might desire.

The presented inter-fairness matching algorithm, matches the tangent fields and hence it preserves features with similar tangent fields of parametric curves. As such, the algorithm can also be employed in the detection or recognition of features. By ranking the match of a give shape, $\mathcal{F}$, against a finite set of recognizable shapes according to the optimal $Cost(m-1, m-1)$ computed using Equation (7), one is able to

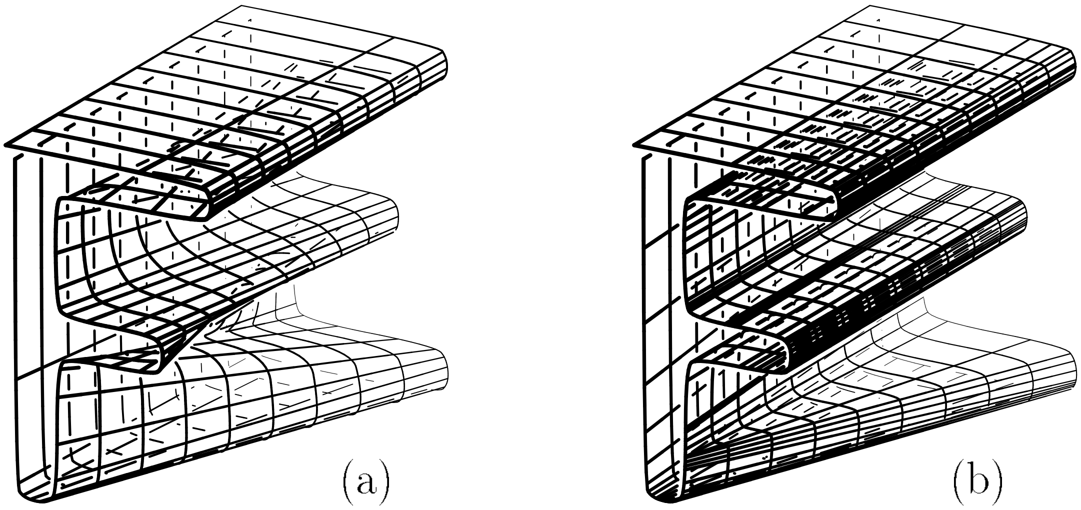


Figure 10: A ruled surface between the letter E and the letter F. In (a) the ruling is computed naively, while in (b) the matching algorithm was first applied. See also Figure 9.

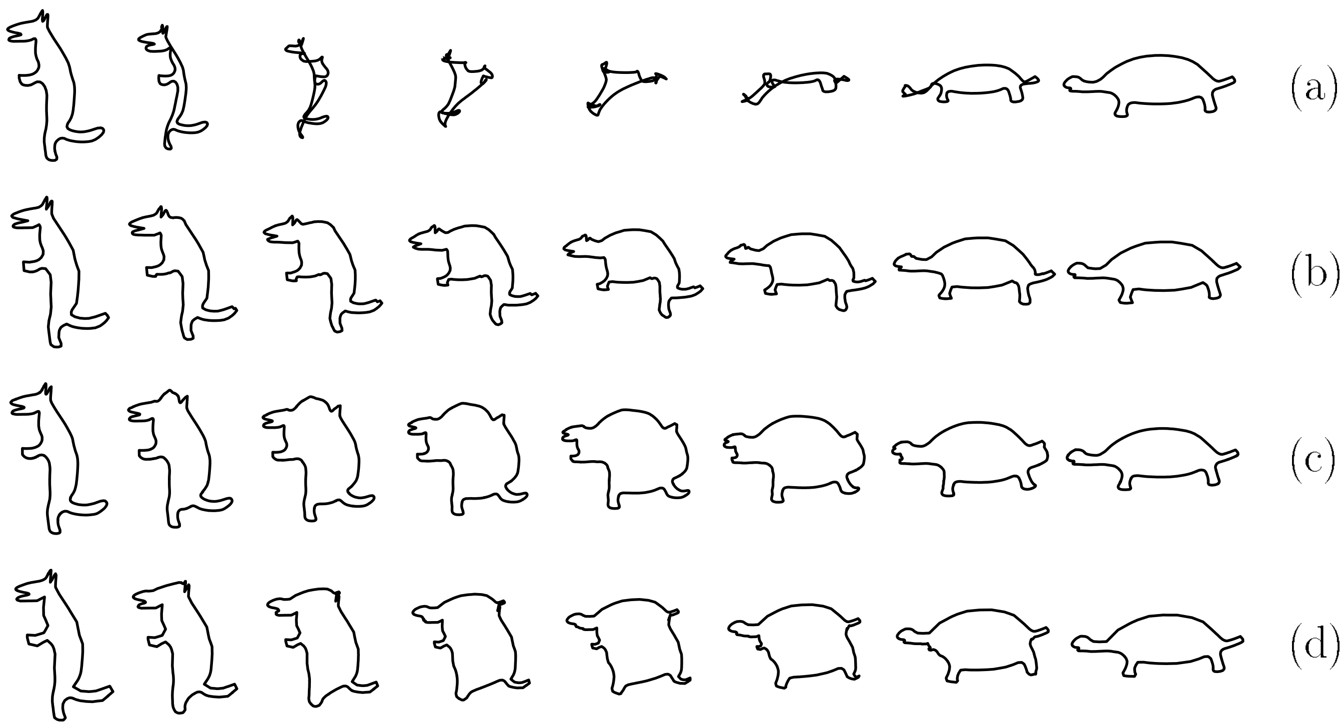


Figure 11: A metamorphosis between a wolf and a turtle. In (a), a metamorphosis with no inter-fairness matching is shown using a simple affine combination. In (b) through (d), matching enhanced by shifting is employed. In (b), matching is established with $m = 100$ and $v(u)$ being piecewise linear reduced to 20 control points. In (c), matching is established with $m = 300$ and $v(u)$ being piecewise quadratic reduced to 50 control points. In (d), matching is established with $m = 20$ and $v(u)$ being piecewise linear reduced to 10 control points.

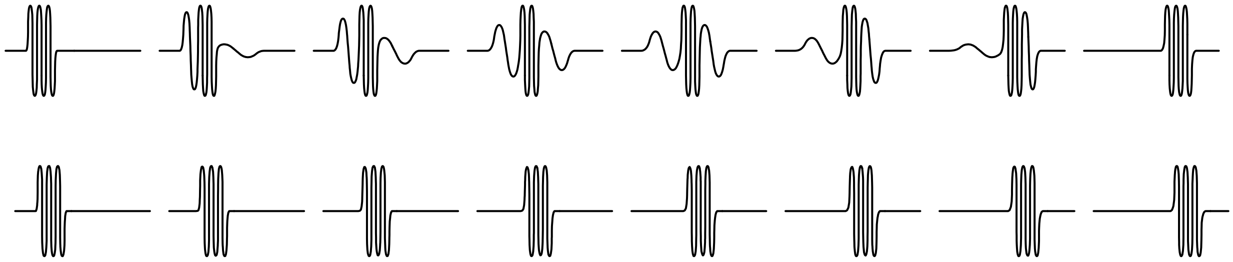


Figure 12: The matching algorithm can also be seen as a feature detector or a feature recognizer. In this metamorphosis sequence, a wave feature is simply translated (compare extreme drawing). The naive metamorphosis of (a) does not preserve the wave feature, while after the matching computation in (b) the wave is completely preserved throughout the sequence, resulting in the expected translational metamorphosis.

recognize $\mathcal{F}$ as the shape with the best $Cost(m-1, m-1)$ from the set of recognizable shapes. In Figure 12, a sine wave feature is completely matched and hence detected in the source and target curves, converting the metamorphosis into a translational process.

Another application that is closely related to morphing is key-frame interpolation for animation. Two outlines of an animal in two different time frames can be inbetweened in time if a proper correspondence can be established. In Figure 13, two outlines of an oryx are shown. The left most sketch was derived from the right most sketch, using a freeform curve editing tool. While the shapes are similar and with similar starting and end locations (near the mouth), the shrinkage of the horn has a devastating influence on the quality of the interpolation as seen in Figure 13 (a). In Figure 13 (b), the established matching clearly eliminates these artifact and vastly improves the quality of the result.

While metamorphosis of freeform curves is an obvious application of the proposed matching algorithm, we have also shown applications in freeform surface modeling. Figure 10, in addition to conveying the proper matching between the letter E and F, also shows a naive ruled surface construction as well as a proper one using matching between the two curves of the E and the F. An additional modeling application for more than two curves is seen in Figure 14. An approximation surface is fitted or skinned through a set of curves, each is mostly linear with a single bell shape feature at a different parameter value. In Figure 14 (a), a naive surface approximation is generated, ignoring the question of inter-fairness relative parameterization, resulting in a surface with several uncorrelated and disjoint features. In Figure 14 (b), using the matching algorithm to match all curves in a sequence, a surface that preserves the feature of the bell shape throughout its parametric domain is automatically created. Given n curves, $C_i(t)$, $0 \leq i < n$, a matching is computed between $C_i(t)$ and $C_{i+1}(t)$, reparametrizing $C_{i+1}(t)$, for i goes from 0 to $n-2$.

We summarize this section with a hint on the computational costs. A matching with a piecewise linear

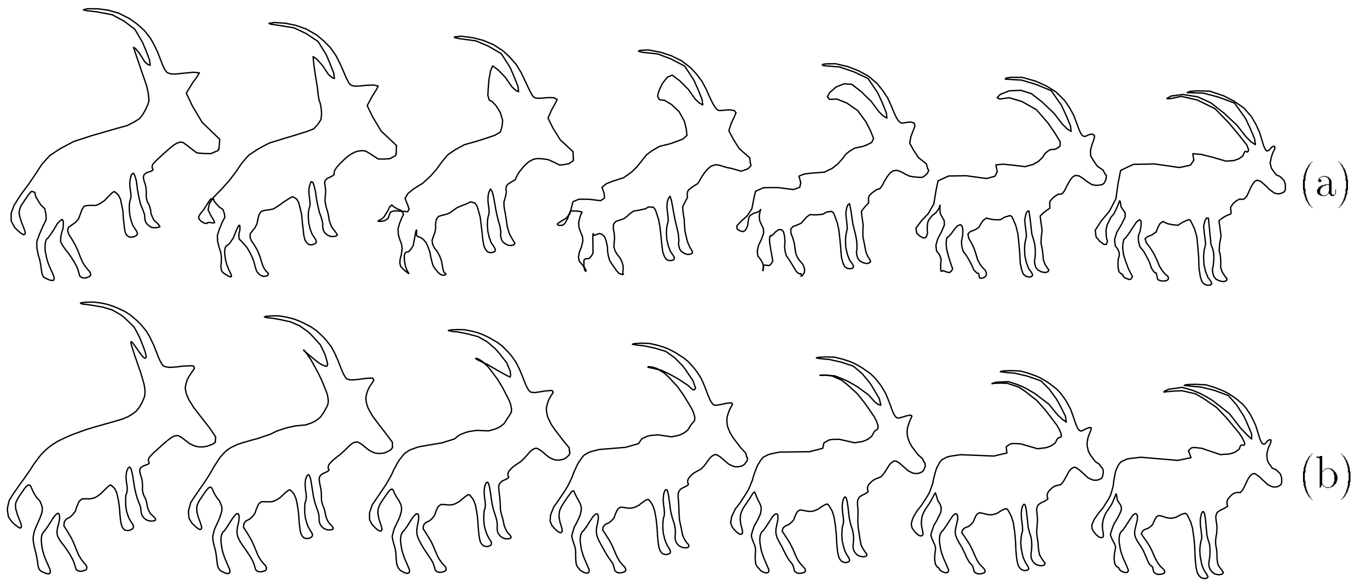


Figure 13: The matching algorithm is applied here to two outlines of an oryx. The left oryx was edited from the original right one and hence these are quite close. Yet the modifications, most noticeably in one of the horns, can have grieving results as seen in (a). In (b), the matching with m = 100 reduced to a piecewise linear $v(u)$ with 20 coefficients significantly improves this result.

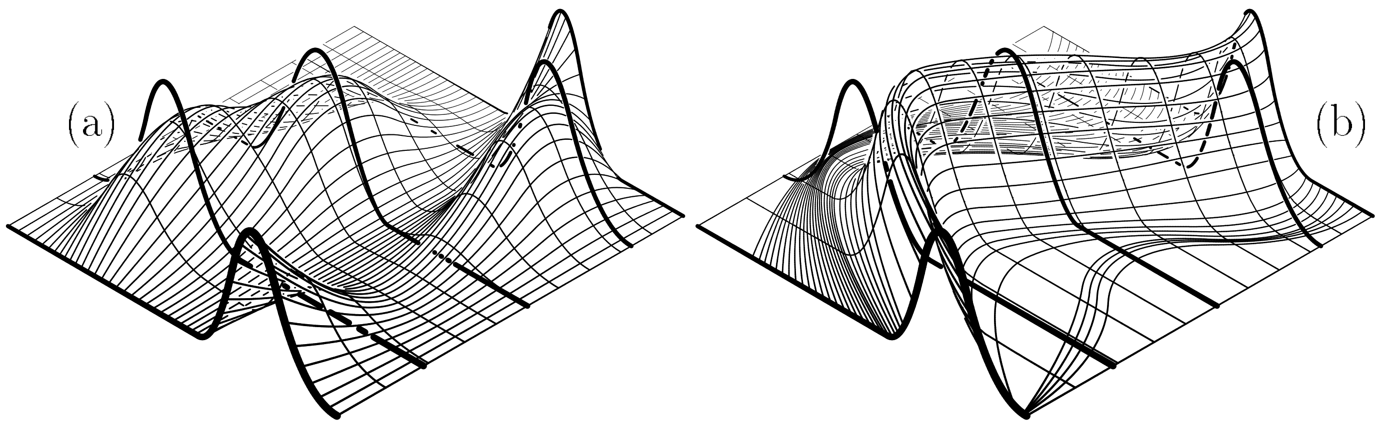


Figure 14: A Bspline surface *approximation* through five curves (thick lines) with each curve with one bell shape feature, at a different location. In (a), the naive fit is shown to clearly ignore the bell shape feature in-between the five cross sections. In (b), after proper relative matching and reparameterization with $m = 50$ and with $v(u)$ least squares reduced to 20 control points, the surface preserves the feature of the bell shape throughout, while approximating the curves.

$v(u)$ function and reparameterization with $m \leq 100$ can be completed in about a second on an SGI 150MHz R4400. Once completed, animating a metamorphosis sequence between the two curves (Equation (1)) can be accomplished in an interactive speed of several frames a second. Going to higher orders or resorting to a larger m will obviously have its impact on the computation time. In practice, we found that a piecewise linear reparameterization $v(u)$ and $m \leq 100$ successfully covers all the tested applications demonstrated.

5 Conclusions

We have presented an inter-fairness method to match the relative parameterization of two or more freeform parametric curves, using first order differential properties. We have shown that the proposed method can be useful in a variety of applications from computer graphics' metamorphosis of freeform curves to better modeling constructors of freeform geometry.

While it is clear that the matching algorithm can benefit other areas such as feature recognition, there are several open questions to be resolved, whose answers will result in a better algorithm. In the approach taken in this work, the m unit tangent vectors were sampled uniformly in the parametric space of the two curves. While it is clear that better sampling can result in a better matching, some sort of curve interrogation techniques must be employed, hindering the efficiency of the algorithm, and possibly eliminating its almost interactive capabilities. Either arc length sampling or curvature based adaptive sampling can be suggested while both necessitate significant overhead of analysis of freeform curves. It is questionable and probably application dependent whether such a preprocessing analysis is desired or necessary.

The presented algorithm can also be extended in various domains. The algorithm guarantees not only to find a match if one exists, but also to compute the best match out of the set of valid matchings. Unfortunately, the algorithm can provide less aid if no valid match can be established. The latter can occur in cases of different topology, where, for example, one closed loop should be matched against several disjoint closed loops. It is a plausible to assume that while the algorithm proposed herein cannot resolve these topological differences, it can serve as a low level geometrical matching tool for a higher level topological matching paradigm.

This work has exploited first order derivatives in the form of unit tangent fields. However, it is clear that other differential forms can be exploited as part of the general framework presented, with equal success. Matching second or third order differential properties can find correspondence between inflection points or points of extreme curvature.

In this work, we matched the two given curves with no filtering. It is expected that in some applications a prefiltering process might be beneficial. If one is attempting to match an outline of a fish to a curve representing a bird, high resolution details in both curves might disrupt the matching process that only attempts to match the tail of the bird to that of the fish. In this case, a low pass filter can alleviate these difficulties. Taking this idea a step forward, it might be useful to consider the matching of two shapes at some prescribed resolution level, once a multi-resolution decomposition of the two Bspline curves have been computed.

Throughout this paper, we have employed an affine combination (Equation (1)) between the source and target curves. Clearly this approach is improper in many instances. While one can establish a matching between the two curves that is guaranteed not to self intersect in the local, this is not the case globally [8]. Furthermore, if the matching cannot be established, affine combinations can fail miserably and self intersect. Lacking more information, the affine combination is a reasonable method of choice. However, better, higher order, blending schemes must be investigated. A similar pursuit for proper interpolation methods for n curves should also be conducted.

Being able to employ dynamic programming opened the way to the efficient computation of the globally optimal inter-fairness match of the discrete problem. Extending this notion to bivariate surfaces or even higher dimensional varieties is difficult. The first order derivatives can be expressed via the Gauss or normal map of the surface as suggested in the discussion on the engraving on a stone. Nevertheless, there is no simple remedy in the form of dynamic programming, anymore. While the global optimum cannot be found in an efficient way, greedy methods that seek local extrema of the prescribed function can be employed, yielding heuristic answers to the matching problem that may be sufficient. These extensions are under current investigation.

6 Acknowledgment

We are thankful to In Kwon Lee and Eli Goldstein for some of the Bspline curves employed in this work. We are also thankful for the reviewers of this paper whose comments greatly improved the quality of this paper.

References

[1] Alpha_1. Alpha_1 User's Manual, University of Utah, Computer Science, 1992.

[2] H. Alt, B. Behrends, and J. Blömer. Approximate matching of Polygonal Shapes. 7th ACM Symposium on Computational Geometry, pp 186-193, 1991.

[3] H. Alt and M. Godau. Measuring the Resemblance of Polynomial Curves. 8th Annual Computational Geometry, pp 102-109, Berlin, Germany, June 1992.

[4] D. H. Ballard and C. M. Brown. Computer Vision. Prentice-Hall, New Jersey, 1982.

[5] M. D. Carmo. Differential Geometry of Curves and Surfaces. Prentice-Hall, 1976.

[6] G. Elber. Free Form Surface Analysis using a Hybrid of Symbolic and Numeric Computation. Ph.D. thesis, University of Utah, Computer Science Department, 1992.

[7] G. Elber. Symbolic and Numeric Computation in Curve Interrogation. Computer Graphics *forum*, Vol 14, No 1, pp 25-34, March 1995.

[8] G. Elber. Metamorphosis of Freeform Curves and Surfaces. Computer Graphics International 1995 (CGI 95), Leeds, June 1995.

[9] D. J. Fillip. Blending Parametric Surfaces. ACM Transaction on Graphics, Vol. 8, No. 3, pp 165-173, July 1989.

[10] H. Fuchs, Z. M. Kedem, and S. P. Uselton. Optimal Surface Reconstruction from Planar Contours. Communication of the ACM, Vol. 20, No. 10, pp 693-702, October 1977.

[11] E. Goldstein and C. Gotsman. Polygon Morphing Using a Multiresolution Representation. Proceedings of Graphics Interface 95, pp 247-254, Quebec City, May 1995.

[12] J. Hoschek and D. Lasser. Fundamentals of Computer Aided geometric Design. A K Peters, Wellesley, Massachusetts, 1993.

[13] IRIT solid modeller. IRIT 6.0 User's Manual, Technion, March 1996.

[14] K. Kim. Blending Parametric Surfaces. M.Sc. thesis, University of Utah, Computer Science Department, 1992.

[15] L. Piegl. Fundamental Developments of Computer Aided geometric Modeling. Academic Press, 1993.

[16] D. Sankoff and J. B. Kruskal. Time Warps, String Edits, and Macromolecules: the Theory and Practice of Sequence Comparison. Addison-Wesley, 1983.

[17] T. W. Sederberg and E. Greenwood. A Physically Based Approach to 2D Shape Blending. *Computer Graphics 26*, Vol. 26, No. 2, pp 25-34, July 1992.

[18] T. W. Sederberg, P. Gao, G. Wang, and H. Mu. 2D Shape Blending: An Intrinsic Solution to the Vertex Path Problem. *Computer Graphics 27*, pp 15-18, August 1993.

[19] B. Serra and M. Berthod. Subpixel Contour Matching Using Continuous Dynamic Programming. IEEE Conference on Computer Vision and Pattern Recognition, pp 202-207, 1994.

[20] M. Shapira and A. Rappoport. Shape blending using the star-skeleton representation. *IEEE Computer Graphics and Application*, Vol. 15, No. 2, pp 44-51, March 1995.

[21] Y. Shinagawa and T. L. Kunii. The Homotopy Model: a generalized Model for Smooth Surface Generation from Cross Sectional Data. The Visual Computer, Vol. 7, pp 72-86, 1991.

[22] M. S. Waterman. Introduction to Computational Biology, Maps, Sequences and Genomes. Chapman & Hall, London, 1995.

[23] A. Witkin and Z. Popovic. Motion Warping. *Computer Graphics Proceedings*, pp 105-108, August 1995.

A Composition of Bézier Curves

Let $v(u)$ be a scalar Bézier curve such that $v(u) \in [0, \ldots, 1], \forall u$. Let $C(v)$ be a Bézier curve. Then,

$$C(v(u)) = \sum_{i=0}^{n} P_i B_i^n(v(u)), \tag{8}$$

where $B_i^n(v)$ is the i'th Bézier basis function of degree n, $B_i^n(v) = \binom{n}{i} v^i (1-v)^{n-i}$.

The composition is now narrowed to the problem of computing the composition of $B_i^n(v(u))$. Assuming one can compute and represent the composition $B_i^n(v(u))$, the curve $C(v(u))$ is also representable as a polynomial because it involves the scaling, addition and multiplication of only polynomial varieties of the form of $B_i^n(v(u))$.

$$B_i^n(v(u)) = \binom{n}{i} (v(u))^i (1.0 - v(u))^{n-i}. \tag{9}$$

Here again, Equation (9) contains the product and difference of polynomial varieties, that can only result in a polynomial form, and therefore representable as a scalar Bézier curve.

If either $C(v)$ or $v(u)$ is rational, the result of the composition would be rational. If $C(v)$ is rational, the P_i control points in Equation (8) would simply be treated as in projective space. If $v(u)$ is rational, Equation (9) now becomes

$$\begin{aligned} B_i^n\left(\frac{v(u)}{w(u)}\right) &= \binom{n}{i} \left(1.0 - \frac{v(u)}{w(t)}\right)^{n-i} \left(\frac{v(u)}{w(t)}\right)^i \\ &= \binom{n}{i} \frac{(w(t) - v(u))^{n-i} (v(u))^i}{(w(t))^n}. \end{aligned} \tag{10}$$

Equation (10) should then be substituted into Equation (8) in a similar way to Equation (9). If now $C(v)$ is also rational, the denominator term in Equation (10), $(w(t))^n$, is canceled out because it appears in both the numerator and the denominator.