Interactive Curve Design using Digital French Curves

Karan Singh

Alias|wavefront,
210 King St. E., Toronto, Canada M5A 1J7
ksingh@aw.sgi.com

Abstract

In the real world designers often use French curves or sweeps to create or edit curves to bring out a personal style or reflect a corporate standard in all their designs. A French curve is a general term for a pre-defined curve template used to create high quality 2D drawings or sculpt 3D models. Research in interactive curve and surface design is continually moving toward direct manipulation of the objects being defined. This paper describes a system in which digital French curves, represented by planar cubic NURBS curves, interactively create and sculpt curves and surfaces that comprise the design of an object. The approach is especially relevant in the early stages of conceptual design to beautify and simplify curves obtained from quick gestural sketches. Algorithmically, the contributions of this paper are twofold. We describe an efficient technique for approximating a planar parametric curve by a small set of elliptic arcs. Proximity computation to the approximating ellipses is simple and efficient, greatly improving the interactivity of the sculpting paradigm. We also describe a simple approach to smoothly replace sections of design curves with sections of the French curve. The results are illustrated within a design system using a puck that provides simultaneous position and orientation information to control the digital sweep, allowing the user the same physical feel and efficiency of motion of a real sweep.

1 Introduction

Finding effective techniques for interactive curve and surface design continues to be a challenging and fertile area of research [3, 6, 7, 9, 10, 11, 20, 22, 23]. While a number of industrial design systems are in widespread use today, traditional design approaches of sculpting and sketching continue to be used. Many industrial designers prefer to build prototypes in a real workshop as part of the initial conceptual design process, because it is important to quickly resolve shape and form in 3D. A real world model shop allows the designer to quickly understand form, shape and size. Blue foam, card or clay models are thus popular just to understand the 3D form and the problems designing with real world constraints. As a result a number of interactive digital approaches that capture the essence of these traditional techniques have been investigated [2, 5, 12, 16, 19, 22, 23, 24].

Designers often use French curves or sweeps to create or edit curves. French curves are typically constructed from 1/8" clear perspex, in a shape, pleasing to the designer [15] (See Figure 1). A similar tool called a steel is widely used by modelers in the automotive industry to sculpt sections on clay models. A set of French curves or steels thus defines character for an entire era of designs.

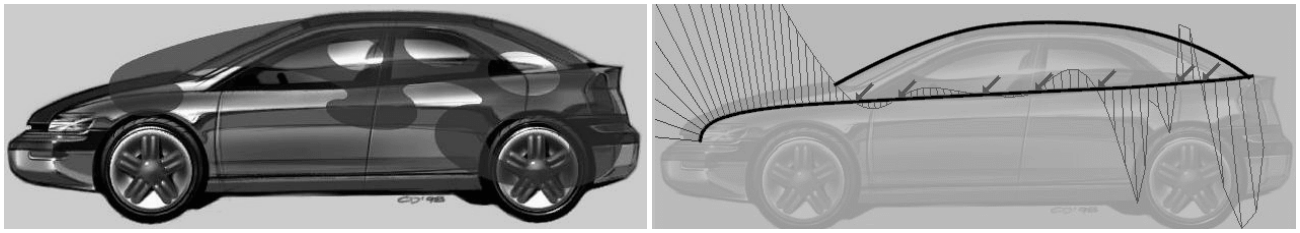
Existing industrial design systems use a number of techniques to create and edit digital curves. These range from the basic manipulation of control vertices, to more sophisticated direct manipulation methods using tangency or curvature constraints, snapping, sculpting and sketching. Illustrator's pencil/pen tools [1], provide an excellent example of current curve editing paradigms. French curves differ signficantly from these techniques in three ways. Firstly, the edited curves precisely fit predefined templates, allowing design curves to accurately reflect a personal or corporate style. Secondly, French curves are carefully crafted to have a minimal curve complexity and desirable curvature properties, which it imparts to edited design curves. Finally, digital French curves emulate a traditional design paradigm, allowing many designers to utilise already acquired skills towards digital curve design.

Figure 1: Physical French Curves

1.1 Problem description

Our model for digital French curves is catered to the initial stages of conceptual object design (See Figure 2). A typical workflow would involve using a French curve to clean up and simplify sections of sketched curves by a creating a replacement curve or sculpting the sketched design curves (See Figure 3,4). We thus make the following assumptions of our design model.

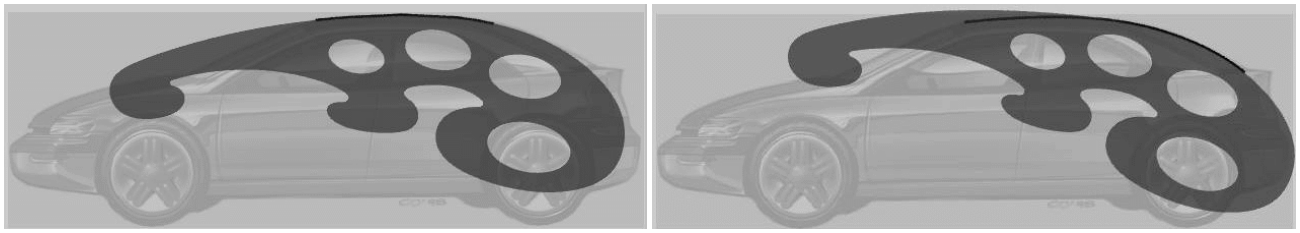
- The digital French curve is planar. When editing design curves that have depth information outside of the plane of the French curve, it is treated like an extruded object (See Figure 5).
- The design curves to be edited are either planar or almost planar. Curves like the sinusoidal wave in Figure 5 will be discussed in the conclusion, but for the most part the design curves will be assumed to be planar.
- We use cubic NURBS curves to model our French curves due to their generality and minimality of representation. Since French curves are also used to enhance the quality of the object design curves, design curves of a different representation are converted to cubic NURBS curves [10, 14] before being edited.



(a) Scanned sketch

(b) Noisy design curve

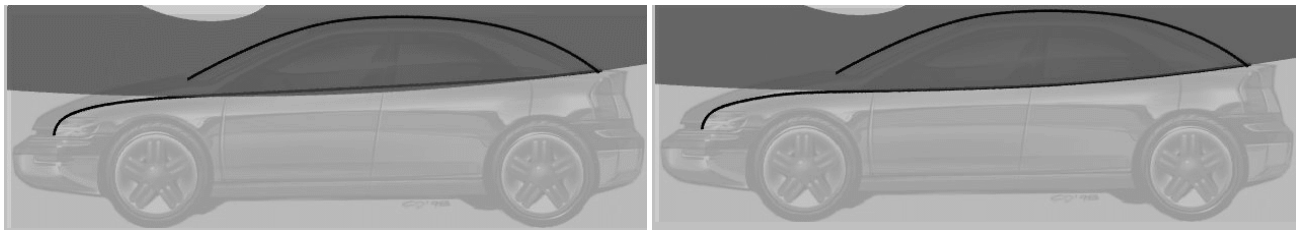
Figure 2: Conceptual Design Workflow



(a) Initial curve creation

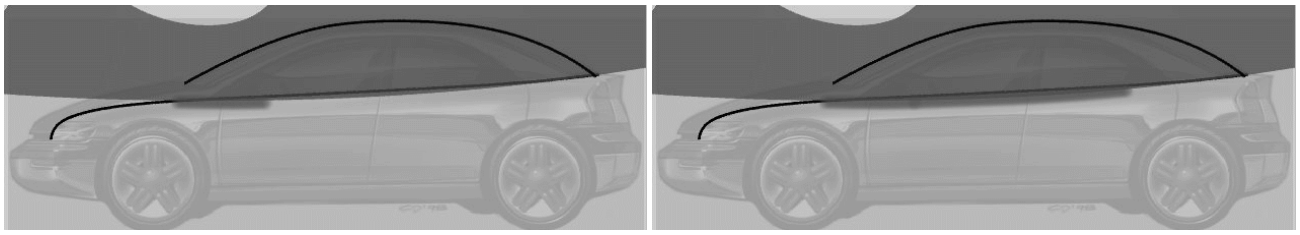
(b) Curve Extension

Figure 3: Curve creation using a French curve



(a) French curve interaction

(b) Edited design curve



(c) Brushing along a French curve

(d) Brushing along a French curve

Figure 4: Curve editing using a French curve

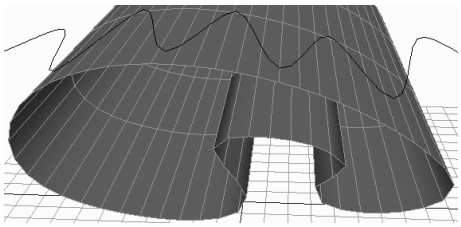


Figure 5: Extruded French curve sculpting a 3D design curve

There are two aspects to design with French curves.

- **Creation**: A segment of the French curve is specified that gets converted into a design curve (See Figure 3).
- **Editing**: Here a French curve interacts with an already existing design curve to alter or extend it (See Figure 4a,b). Editing is a two step process:
  Correspondence: Determining which part of the design curve is being edited by which part of the French curve.
  Replacement: Replacing a section of the design curve with a section of the French curve.

1.2 Overview

The remainder of this paper is organized as follows. Section 2 proposes approaches to the correspondence problem, based on proximity calculations between points on the design and French curves. We continue our discussion of the correspondence problem in Section 3 with an efficient solution to curve proximity calculation, achieved by approximating the curve by set of elliptic arcs. Section 4 describes the algorithm for generating a smooth curve by replacing a section of a design curve with a section from a digital French curve. Section 5 provides implementation details and Section 6 concludes with a discussion of the results obtained.

2 Curve correspondence

This is the first step during interaction with a digital French curve, where a section of the French curve that creates or edits a section of a design curve must be specified. We discuss two approaches to calculating this correspondence: brushstrokes and interactive sculpting.

2.1 Direct specification using brushstrokes

Brushstrokes are the basic technique for the creation of design curves using digital French curves [13]. The user sketches/brushes over a section of the French curve (See Figure 4c,d), that forms a new design curve (See Figure 3a). When editing an existing design curve, brushing along a French curve precisely specifies the section of the French curve to be used as a replacement for part of the design curve.

Creating a new design curve from brushing over a French curve is simply a matter of generating a sub-curve [10] from the formulation of the digital French curve. Editing design curves using French curves is based on the premise that the section of the design curve being edited has the same general shape and position in space as the section of the French curve it will be replaced with. We therefore assign a user controlled thickness so that the brushstroke covers an area around the specified section of the French curve. The part of the design curve that lies within this area is inferred as the section to be replaced.

2.2 Specification based on interactive sculpting

Reiterating the premise that the section of the design curve being edited and the section of the French curve used for the editing are spatially proximal, one may attempt to infer both sections implicitly from the spatial relationship of the design and French curves. What results is an interactive approach where sections of the French curve continually sculpt the design curve by replacing sections of the design curve that are proximal to it. This approach has greater interactivity than using brushstrokes for editing since both manipulation and editing is accomplished in a single operation.

2.3 Calculating proximity between curves

A simple approach to establishing correspondence between proximal sections of the design curve and French curve would be to find the Euclidean closest point to one curve from a sampling of points along another. We sample points along the design curve. A sequence of sample points for which the Euclidean closest distance to the French curve is within a given tolerance (the thickness of a brushstroke), define a section of the design curve. The closest points on the French curve correspondingly provide a section of the French curve with which to sculpt the design curve. Closest point to curve calculations lie at the heart of virtually any correspondence algorithm such as the one outlined above.

The precise calculation of the closest point to a curve is a computationally expensive operation. Techniques based on the deCasteljau construction of parametric splines [8, 21] provide a closest point by recursively subdividing the convex hull of the curve in the local proximity of the estimated closest point. The algebraic distance to an implicitly defined curve or surface is usually easier to compute than Euclidean distance and may be used as a good proximity metric instead of the shortest Euclidean distance [17]. We only require approximate proximity calculation since fine tuning the correspondence and the transition from the unedited design section to the French curve section is ultimately under user control. We thus propose a mechanism where the French curve is first approximated by a small number of elliptic arcs and straight lines. We then use algebraic distance to the arcs instead of Euclidean distance to efficiently determine the proximity of points to the French curve.

Given a point in the plane and an elliptic arc, we transform the point to the local reference frame of a circle, which when non-uniformly scaled results in the given ellipse. If the point lies within the angular range of the arc, the algebraic distance provides the proximity metric, else the square of the Euclidean distance to the closer of the two arc end-points is used. For a point in the plane we thus iterate through the sequence of elliptic arcs until the point lies within the proximity requirements of some arc.

The next section describes the approximation of planar curves using elliptic arcs. Note that there are a number of alternate approaches to calculating proximity of a point to a curve [8, 21, 4]. The elliptic arc approximation, however, is particularly well suited to our application, since physical French curves have been traditionally crafted from conic sections.

3 Curve fitting using elliptic arcs

The algorithm first breaks the curve into a number of potential curve segments based on inflection points and points of extreme curvature. At this point each segment could be finely sampled and Pratt's least squares approach [17, 18] used to fit an ellipse through the points. Alternatively, since we have a continuous curve, we can use the ability to evaluate points and their derivatives at arbitrary parameter value to good advantage. We thus calculate elliptic arcs analytically using the position, tangent and curvature of each curve segment at the break points. Finally we try and combine the arcs

representing adjacent segments to reduce the number of elliptic arcs used.

3.1 Generation of break points

This step is based on the simple observation that the curvature of an ellipse is a well behaved function that attains extreme values at the intersection with its principal axes. The candidate break points on the curve are thus points of local extrema of curvature along the curve (See Figure 6). In our implementation we simply sample parametric curves using an adaptive step, to approximate an arc length parametrization. A low-pass filter is applied to the curvature along the curve, so that local extrema caused due to curvature changes at frequencies higher than the filter cutoff do not generate break points. Lower cutoffs thus result in fewer break points or curve segments and subsequently a fewer number of elliptic arcs, at the cost of accurately representing the high frquency curvature changes in the curve.

We also use break points to demarkate intervals of near constant curvature along the curve which can be simply represented by circular arcs, or straight line segments in the case of near zero curvature.

Finally we add break points when the normal to the curve rotates through more than a right angle without the generation of a break point. This is to deal with curves like spirals (See Figure 11) where curvature changes monotonically.

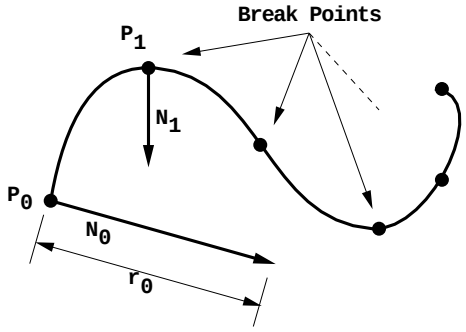


Figure 6: Break points on a curve

3.2 Fitting a curve segment with an elliptic arc

A solution to this problem using existing fitting methods would be to sample the the curve segment to generate a cloud of points on the curve to which an elliptic arc may be fit using a least squares approach.

What we propose below is an efficient analytic approach. The algorithm is based on the premise that the input curve segment is itself a monotonic polynomial function, typically of low degree, such as a cubic and therefore will be well approximated by the ellipse in the interior of the segment if the curve and its approximating ellipse match up well at the break points in terms of position, tangent and curvature.

Let point P_0, P_1 be two adjacent break points. Let the normals to the curve at these points be N_0, N_1 and the respective radii of curvature r_0, r_1 (See Figure 6). Since the points are local extrema on the curve we attempt to make the points lie on the principal axes of the elliptic arc. We will construct an ellipse passing through point P_0 whose principal axis A lies along N_0. The parametric equation of such an ellipse in a reference frame defined by N_0 with the origin at P_0 is

$$x - a = a\cos t . y = b\sin t. \tag{1}$$

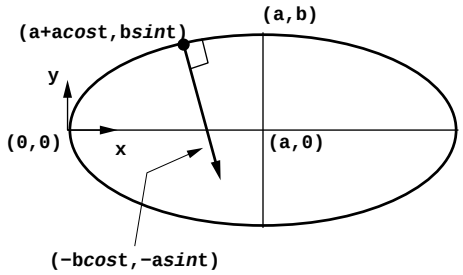


Figure 7: Definition of an ellipse

The radius of curvature at a point on the ellipse is given by

$$r_{ell} = \frac{(a^2\sin^2 t + b^2\cos^2 t)^{3/2}}{ab}. \tag{2}$$

We would like to match the curvature of the ellipse and the curve at P_0.

$$r_0 \approx r_{ell_0} = b^2/a. \tag{3}$$

Additionally the normal vector to the ellipse at a point is

$$N_{ell} = (-b\cos t, -a\sin t). \tag{4}$$

Let $(i, j) = P_1^{local}$ and $(n_i, n_j) = N_1^{local}$ be $P_1 and N_1$ transformed into the reference frame of the ellipse (See Figure 8).

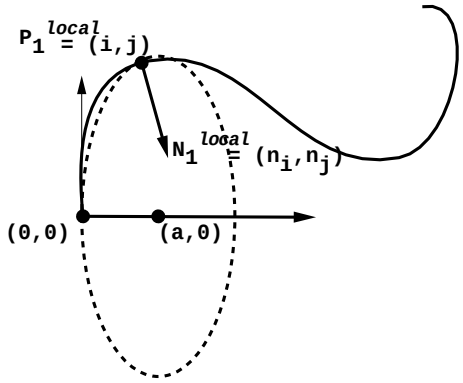


Figure 8: Fitting an elliptic arc to a curve segment

Once again we would like the normal vector for the ellipse and the curve at P_1 to be the same. Using Eq. (1) and Eq. (4) this is equivalent to

$$\frac{(i-a)b}{ja} \approx \frac{n_i a}{n_j b}. \tag{5}$$

Eq. (3) and Eq. (5) may be rewitten in the form

$$\frac{1}{b^2} = \frac{1}{a^2}(m_1 + m_2 a). \tag{6}$$

where $m_1 \approx 0, m_2 \approx 1/r_0$ for Eq. (3) and $m_1 \approx \frac{in_j}{jn_i}, m_2 \approx \frac{-n_j}{jn_i}$ for Eq. (5).

Substituting Eq. (6) into the implicit form of Eq. (1) at point (i, j) we get

$$a = \frac{i^2 + m_1 j^2}{2i - m_2 j^2}. \tag{7}$$

Based on the continuity properties of the curve and the criteria for generating break points we can assume that $i > 0$, otherwise we have a segment of zero curvature best represented by a straight line. Also observe that $n_i \geq 0, jn_j < 0$

A valid solution must have $a \geq i$. Therefore, according to Eq. (7) the curvature of the ellipse at P_0, $r_{ell_0} \leq \frac{j^2}{i}$. Also from the equation, for a to be positive $2ir_{ell_0} - j_2 > 0$, or $r_{ell_0} > \frac{j^2}{2i}$.

Suppose $m_1 = \frac{in_j}{jn_i}, m_2 = \frac{-n_j}{jn_i}$. Substituting this into Eq. (7) we get $a = \frac{i(in_i+jn_j)}{2in_i+jn_j}$. The curvature of the ellipse at P_0 would be, $r_{ell_0} = \frac{b^2}{a} = \frac{j^2}{i}(1 + \frac{in_i}{jn_j})$.

We pick a value for the radius of curvature that tries to match both the normal vector to the curve at P_1 and the radius of curvature of the curve at P_0.

The radius of curvature at P_0 for the ellipse is thus $r_{ell_0} = (r_0 + \frac{j^2}{i}(1+\frac{in_i}{jn_j}))/2$. We constrain r_{ell_0} to lie between $(\frac{j^2}{2i}, \frac{j^2}{i}]$ and use Eq. (7) to set $a = \frac{i^2 r_{ell_0}}{2ir_{ell_0} - j^2}, b = \sqrt{ar_{ell_0}}$.

We can now calculate the parameter t for P_1 on the ellipse as $t = sin^{-1}(j/b)$ from Eq. (1) and hence normal N_{ell_1}, and radius of curvature r_{ell_1} to the ellipse at P_1 using Eq. (4) and Eq. (2) respectively.

We use $(r_0 - r_{ell_0})^2 + (r_1 - r_{ell_1})^2$ as a measure of fitness for the curve segment; the smaller the fitness value, the better the fit.

Since the resulting ellipse depends on the order of the adjacent break points, we use the above algorithm to generate two elliptic arcs, one for each ordering of the two break points. The arcs that lie within an acceptable fitness threshold are kept. Each approximating ellipse is defined by a center $C = P_0 + aN_0$, an orientation N_0 and the two principal axes a and b.

In the circumstance that neither arc is considered fit enough, we subdivide the curve segment in the middle and repeat the process on the two smaller curve segments. Observe that in the limiting case small curve segments will eventually be approximated by line segments.

3.3 Combining elliptic arcs

Once we have generated a sequence of elliptic arcs we repeatedly iterate over the sequence in an attempt to coalesce adjacent arcs into a larger composite arc. If any two adjacent approximating ellipses match up within a given tolerance in terms of the values of their center, orientation and principal axes we generate a composite approximating ellipse, defined by the average of these values. The fitness of this ellipse is the sum of the fiteness of the component ellipses. We also combine and reduce the number of curve segments. A comprehensive solution to the elliptic arc combination problem would maintain a heap of possible combinations prioritized by fitness at every iteration step. Our implementation, however, is a sequential, greedy approach, that works well in practice.

When approximating ellipses can be combined no more we iterate through the existing sequence of curve segments and simply pick the fittest approximating elliptic arc, circular arc or line segment to represent that part of the input curve.

3.4 Results

The results of our approach can be seen on the test cases below.

Figure 9a shows a French curve represented by a cubic NURBS curve with 17 control vertices, with a plot of curvature variation along the curve. Figure 9b shows the set of 14 approximating elliptic arcs of better fitness. Figure 9c shows the set of 14 approximating elliptic arcs of lower fitness and Figure 9d shows the result of combining the 14 elliptic segments into 7 composite segments. A whole ellipse is used to depict the arcs that get combined.

Figure 10a shows a noisy free sketched curve. Figure 10b shows the filtered curve with its calculated break points. Figure 10c shows the set of 16 approximating elliptic arcs of better fitness. Figure 10d shows the result of combining the elliptic segments into 10 composite segments.

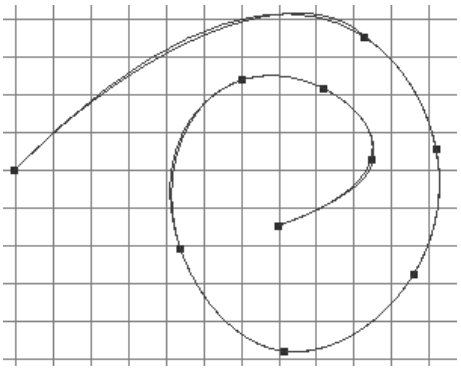
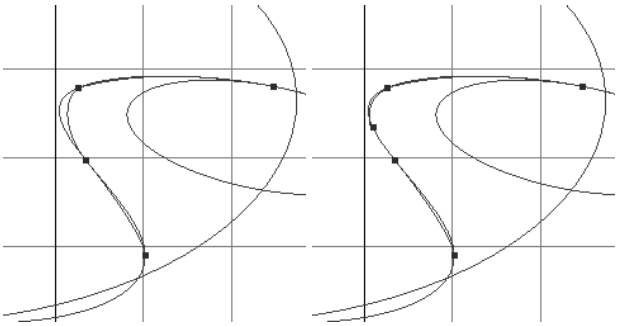


Figure 11: Approximating a spiral with elliptic arcs

Figure 11 shows a spiral being fitted by elliptic segments. The increasing curvature in this case prevents any of the arc segments from being combined.



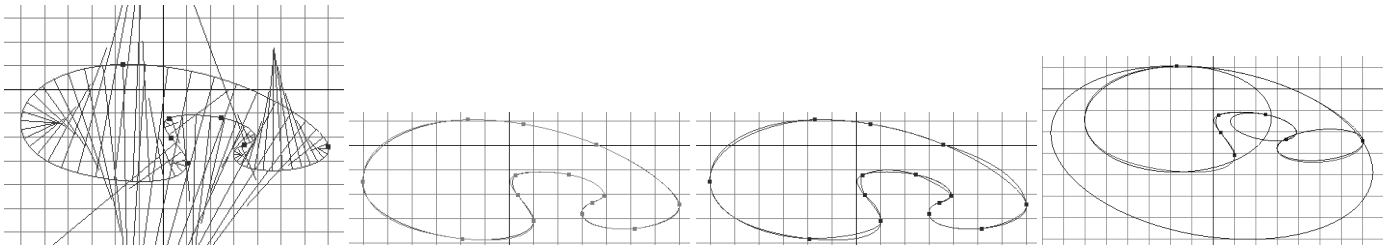
(a) Higher error tolerance (b) Lower error tolerance

Figure 12: Varying tolerance on the fitness of elliptic arcs

As can be seen from the three examples above the fit of arcs obtained from the initially computed break points itself provides an effective approximation to the curve. If we lower the threshold of acceptable fitness in the case of Figure 9b we see that an additional break point is generated in the concave part of the curve on the left to obtain a better set of approximating arcs (See Figure 12a,b).

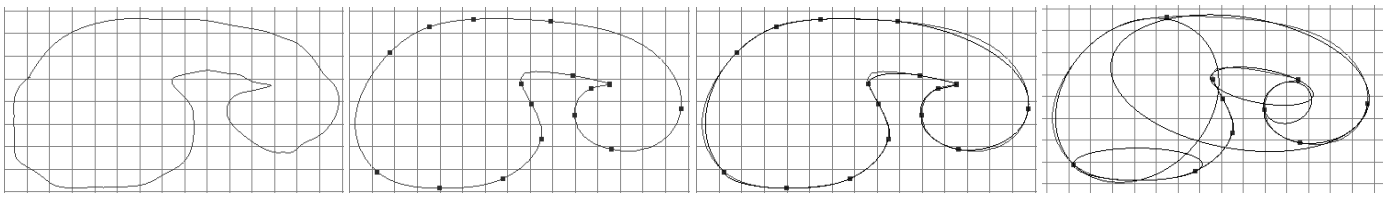
4 Curve editing using a French curve section

Once a correspondence has been established between sections of the design curve and French curve, the section of the design curve needs to be altered to track the section of the French curve. The initial design curves, which are typically scanned or gesturally sketched (See Figure 2) tend to have a large number of control vertices. To reemphasize, the motivation behind using digital French curves, is not only to beautify the design curves to conform to the aesthetically pleasing French curve sections but to reduce the complexity of the design curves in terms of the number of control vertices used to represent them.



(a) Curvature variation (b) Primary fit (c) Secondary fit (d) Combined fit

Figure 9: Approximating a French curve with elliptic arcs



(a) Unfiltered curve (b) Postfilter break points (c) Primary fit (d) Combined fit

Figure 10: Approximating a noisy curve with elliptic arcs

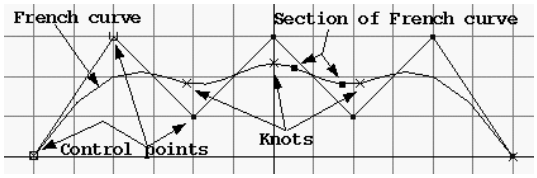
We thus propose that the sculpting operation be accomplished, not by deforming the control points of the existing design curve but by appropriately replacing the control points and knot sequence pertaining to the section of the design curve with the control points and knot sequence from the section of the French curve. We insert knots at the transition boundaries to control the blend from the unedited section of the design curve to the newly inserted section of the French curve.

Let a design curve be represented as a cubic NURBS curve with knot sequence $< d_0, .., d_{(m+2)} >$ with a control vertex sequence $< v_0^d, .., v_m^d >$. The relationship between the number of knots and control vertices results from specifying every knot to be a simple knot except for triple knots at the end points [10]. Let the French curve similarly be represented by knots $< f_0, .., f_{(n+2)} >$ and control vertices $< v_0^f, .., v_n^f >$.

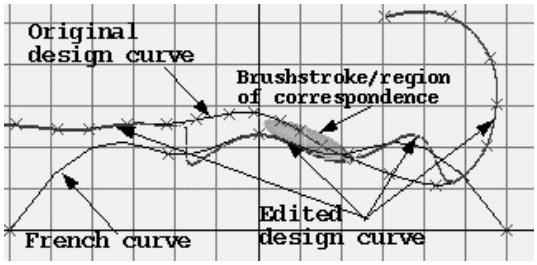
The corresponding sections on the design and French curves are indicated using a parameter range on each curve. Given a parameter range on a NURBS curve we first extract the section of the knot sequence that contains the parameter range. The pertintent control vertices for the knot subsequence are easily calculated based on the degree of the curve. For cubics, the i^{th} knot influences control vertices $i-1, i, i+1$ [10]. Figure 13a, show a cubic curve with control vertices $0..6$ and a knot sequence with multiple end knots $0, 0, 0, 1, 2, 3, 4, 4, 4$. The specified parameter range $[2.1, 2.8]$ is thus defined by knot subsequence $2, 3$ and control vertices $2..5$.

We replace the knot subsequence $d_i, .., d_j$ for a section of the design curve with the knot subsequence for a section of the French curve $f_r, .., f_s$ by reparametrizing every knot f_p, where $p \in [r, s]$, to a knot value $d_i + \frac{(f_p - f_r) * (d_j - d_i)}{(f_s - f_r)}$. The control vertex subsequence corresponding to the knot subsequence $f_r, .., f_s$ is simply projected onto the design plane and inserted in place of the control vertex subsequence corresponding to the knot subsequence $d_i, .., d_j$ to define the new design curve.

This is shown in the Figure 13b using the curve in Figure 13a as the French curve. Notice that though the resulting curve conforms before the French curve in the parameter range specified there is undesirable behavior as the curve makes a transition from the original design curve to the French curve section. Better control results by inserting simple knots into the curves at the parameter range extrema performing replacement of the design curve section. This is shown by the added knots on the French curve and the almost acceptable result in Figure 14a. We find the addition of two proximal simple knots at extrema of sections of the design curve and French curve to give good results in most practical cases Figure 14b.



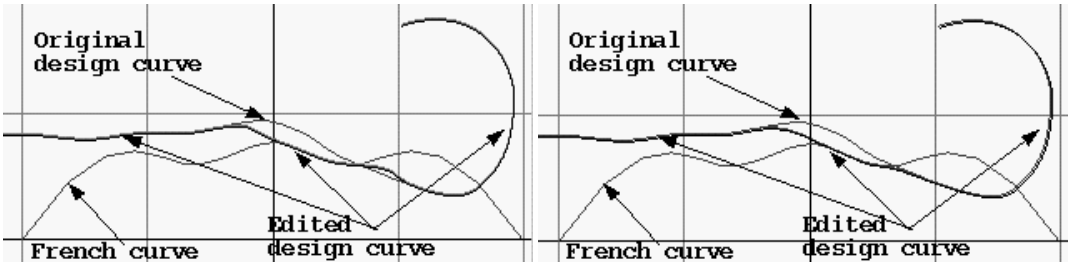
(a) Extracting a curve section



(b) Resulting design curve

Figure 13: Replacement of design curve section with French curve section

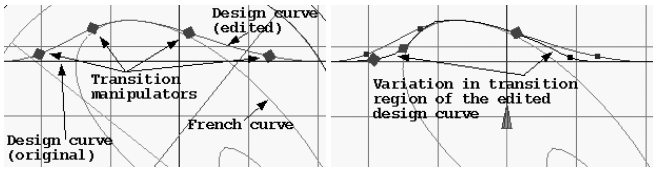
Manipulation of the position of these inserted knots provides the user with control over the transition from the unedited design curve to the French curve section (See Figure 15).



(a) One inserted knot

(b) Two inserted knots

Figure 14: Replacement after knot insertion



(a) Default

(b) Moving three manipulators inward

Figure 15: Manipulators controlling the transition between design and French curve

5 Implementation

Curve design using digital French curves has been implemented as a plug-in in our modeling and animation system Maya.

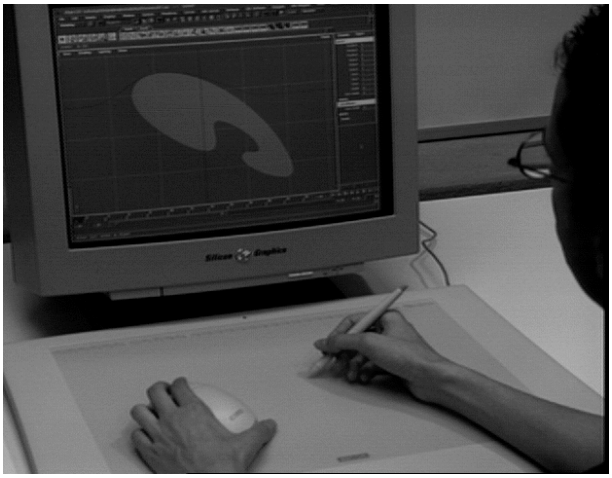


Figure 16: User interaction using a puck, pen and tablet

A typical workflow involves interactively sculpting design curves by affine transformations of the French curve (See Figure 16). Once the sculpted section of the design curve has the general desired shape, the behavior of the curve in the region of transition between the unedited design curve and the French curve can be edited using the point on curve manipulators shown in Figure 15. The resulting curve in Figure 15a is shown ghosted in Figure 15b to indicate the change in behavior. Finally the edit operation is committed and the user can continue to sculpt the edited design curve.

We have prototyped an interaction style for digital French curves which is based on the two-handed input paradigm proposed by Kurtenbach [13]. This paradigm emulates the common method of working with physical French curves on paper where the non-dominant hand positions and orients the French curve, while the dominant hand is free to draw along the French curve. Our prototype implements this style of interaction by using the Wacom Intuos digitizer tablet which allows simultaneous use of the puck and pen (see Figure 16). The puck, which senses its position and orientation, is used to control the digital French curve while the pen can be used for brushstrokes along the French curve. In the case of sculpting with a French curve, where drawing isn't required, we envision that the pen could be used to trigger the acceptance of curve edits and to adjust the point on curve manipulators for the inserted knots (See Figure 15).

We do not, however, dictate a strict assignment of functionality of hands. For example, by adding graphical manipulators to the digital French curves, both control of the French curve and drawing (or adjusting manipulators while sculpting) could be accomplished with one input device by serially adjusting each manipulator. In comparison to the two-handed interaction style, though, serial interaction may be less efficient. Also, some positioning and sculpting tasks may require precise positioning of the French curve and the user may switch to using the puck in the dominant hand for more control.

Currently, our interaction paradigm has had limited testing. Our initial impressions are that, in the sculpting case, simultaneous control of position and orientation of a French curve allow quick exploration of how a French curve affects a target curve, aiding a designer in quickly determining the correct position of the French curve for a desired effect.

We compared the efficiency of our implementation using approximating elliptic arcs for calculating proximity between the design and French curve against a fixed depth subdivision approach [8]. Interaction rates were about 20 Hz using elliptic arcs and about 14 Hz using subdivided control polygons for results of comparable quality. The speedup is largely due to the much smaller number of elliptic arcs than subdivided line segments.

6 Conclusion

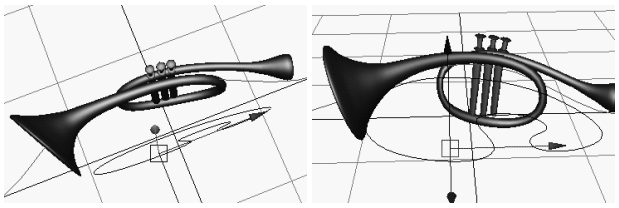
The digital French curve paradigm is of great value in the initial stages of conceptual object design. To begin with they provide an efficient mechnism for neatening and simplifying sketched or scanned design curves and endowing them with a minimal representation that has good continuity and curvature properties. Equally important is the stylistic uniformity and character they provide to an entire era of object designs.

The algorithm described for approximating a parametric curve

using elliptic arcs efficiently solves the proximity to a 2D curve problem that is essential to the interactivity of the sculpting approach. The comparison to another approach in Section 5 illustrates its efficiency but alternate algorithms [21, 8] can provide acceptable results. Additionally the elliptic arc algorithm is of utility to other applications which require the computation of other curve attributes such as arc length along the curve, ray-curve intersection or an inside/outside function.

It may be possible to extend the approach to approximate 3D curves. For space curves the algorithm should generate break points whenever the plane of the curve as determined by a Frenet frame moving along the curve deviates by more than a given tolerance. Each segment can then be approximated by an elliptic arc in the plane of an average Frenet frame running over the curve segment. Note that Frenet frames are ill-defined at points of inflection on curves but these already generate break points on the curve.

Since our digital French curves to some extent emulate physical French curves, we are interested in enhancing the emulation by using digital French curves on digitizer tablets, where the display surface and input surface are combined. Systems of these type eliminate the displacement of the hands from the artwork which is present in current systems where the display and tablet are seperated. We believe that integrated display/tablet system would further bridge the interaction gap between a digital and a physical French curve.



(a) Curve with depth (b) Sculpted curve

Figure 17: Editing a 3D design curve

Throughout this paper, we have addressed the application of French curves to editing planar design curves. It is clear that a fundamentally different approach is needed to apply French curves to sculpting curves such as the sine-wave in Figure 5, since the minimal knot sequence of the French curve section is not enough to represent the curve variation normal to the design plane. For curves with minimal variation outside of the design plane such as in Figure 17a, we simply constrain edit points of the sculpted curve, to snap to the original design curve, in the normal direction to the design plane. This largely preserves the shape of the design curve orthogonal to the design plane without affecting the curve projection in the design plane.

Summarising, we have investigated the use of a digital equivalent of French curves or French curves for use in conceptual object design, and found them to be an appealing interactive technique for curve and surface design.

Acknowledgements

Many thanks to Steve Spenceley, Severin Wille and the A|W Design division for posing the problem and its user-view requirements. We would also like to thank Eugene Fiume and Gord Kurtenbach for their guidance and assistance with the writing of this paper and Jerome Maillot, George Fitzmaurice, Ravin Balakrishnan and the anonymous reviewers for their comments.

References

[1] Adobe Illustrator 8.0, User Guide. *Adobe Systems*, 65–80, 1998.

[2] T. Baudel. A mark-based interaction paradigm for free-hand drawing. *UIST '94 Proceedings*, 185–192, Nov. 1994.

[3] W. Boehm, G. Farin and J. Kahmann. A survey of curve and surface methods in CAGD *Computer Aided Geometric Design*, 1–60, 1984.

[4] C. de Boor, K. Hollig and M. Sabin High accuracy geometric Hermite interpolation. *Computer Aided Geometric Design*, 4:4,269–278, 1987.

[5] V. Branco, A. Costa, and F.N. Ferriera. Sketching 3D models with 2D interaction devices. *Eurographics '94 Proceedings*, 13(3):489-502, 1994.

[6] K. Brodlie A Review of Methods for curve and function drawing *Mathematical Methods in Computer Graphics and Design*, 1–38, 1980.

[7] G. Celniker and D. Gossard Deformable curve and surface finite elements for free-form shape design. *Computer Graphics*, 25:257–266, 1991.

[8] B. Crespin, C. Blanc and C. Schlick. Implicit Sweep Objects. *Proc. Eurographics*, 26–30, 1996.

[9] John A. Eisenman. Graphical editing of composite bezier curves. *Master's thesis*, EECS, MIT, 1998.

[10] G. Farin Curves and surfaces for computer aided geometric design. *Academic Press*

[11] A. Finkelstein and D. H. Salesin Multiresolution Curves *Computer Graphics*, 261–268, 1994.

[12] T. Galyean and J. Hughes. Sculpting: An interactive volumetric modeling technique. *Computer Graphics*, 25(4):267-274, July 1991.

[13] G. Kurtenbach, G. Fitzmaurice, T. Baudel and B. Buxton. The Design of a GUI Paradigm based on Tablets, Two-hands, and Transparency. *ACM CHI*, 35–42, 1997.

[14] O. Lozover and K. Preiss Automatic generation of a cubic B-spline representation for a general digitalized curve. *Eurographics*, 119–126, 1981

[15] D. J. McConalogue Algorithm 66 - an Automatic French-Curve Procedure for Use with an Incremental Plotter. *Computer Journal*, v14, 207–209, 1971.

[16] T. A. Ommundsen, C. Kud, K.J. MacCallum and T.J. Williams Tolerance-Dependent Modelling Approach for Curve Manipulation *North-Holland*, 295–301, 1979.

[17] V. Pratt. Direct Least-Squares Fitting of Algebraic Surfaces. *Computer Graphics*, 21(4), 145–152, 1987.

[18] Theo Pavlidis Curve Fitting with Conic Splines *ACM Trans. on Graphics*, 2(1), 1–31, 1983.

[19] D. Pugh. Designing solid objects using interactive sketch interpretation. *Computer Graphics (1992 Symposium on Interactive 3D Graphics)*, 25(2):117-126, Mar. 1992.

[20] R. F. Riesenfeld. A Film on Schemes for Interactive Curve Design. *Computer Graphics*, 10:66, 1976.

[21] P. Schneider. Solving the Nearest-Point-on-Curve Problem. *Graphics Gems*, Academic Press, vol.1:607–612, 1990.

[22] P. Schneider. Phoenix: An interactive curve design system based on the automatic fitting of hand-sketched curves. *Master's thesis*, Department of Computer Science, U. of Washington, 1988.

[23] K. Singh and E. Fiume Wires: A geometric deformation technique. *SIGGRAPH 98*, 405–414.

[24] R. Zeleznik, K. Herndon and J. Hughes SKETCH: An Interface for Sketching 3D Scenes *SIGGRAPH 96*, 163–170