EUROGRAPHICS Symposium on Sketch-Based Interfaces and Modeling (2011)
T. Hammond and A. Nealen (Editors)

Neatening sketched strokes using piecewise French Curves

James McCrae and Karan Singh

Dynamic Graphics Project, University of Toronto
{mccrae, karan}@dgp.toronto.edu

Abstract

We apply traditional bimanual curve modeling using French curves to the problem of automatic neatening of sketched strokes. Given a sketched input stroke and a set of template French curves we present an approach that fits the stroke using an optimal number of French curve segments. Our algorithm operates in both curvature and point space, reconstructing the salient curvature profiles of French curve segments, while limiting error accumulation resulting from curvature integration. User-controlled parameters allow the neatened stroke to model G^2 continuous curves, capture G^1 discontinuities, define closed curves and explore the trade-off between fitting error and the number of French curve segments used. We present an interactive sketch stroke neatening implementation to demonstrate the real-time performance of our algorithm and evaluate the quality of its results.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Line and curve generation

1. Introduction

Adapting tools and workflows from traditional design to the digital realm has been a popular and effective approach to leveraging the domain expertise of many designers. A large number of interactive digital modeling tools are strongly inspired by sketch and sculpt metaphors [KS08, Sin06]. In traditional engineering and product design, sweeps and Burmester or French curves or sweeps have been extensively used in the creation and editing of 2D design curves. Physically, these curves or sweeps are constructed out of plastic, wood or steel and used as shape guides along which to draw or manipulate clay [McC71]. Figure 1 shows a complete Burmester set from a German engineering encyclopedia *Lexicon der gesamten Technik*, 1904. Here we use French curve as a term encompassing pre-defined curve templates that influence a design in two ways: they are used to neaten rough design marks to create high-quality design curves and the curves themselves capture a personal signature or corporate standard across a family of designs. A digital equivalent of French curves was introduced in [Sin99] as a technique to sculpt NURBS curves within conventional CAD modeling pipelines.

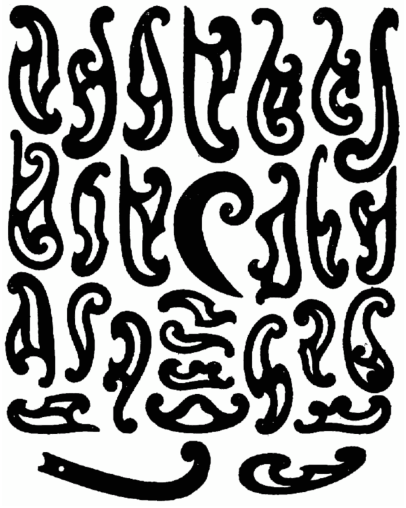

Figure 1: *A complete French curve set from the Lexikon der gesamten Technik, 1904.*

The traditional application of French curves have a bimanual interaction [KFBB97], where one hand selects and positions the desired physical French curve, while the other hand commits a segment of it by sketching along it or dragging the segment profile across a deformable 3D clay model. Singh in [Sin99] sought to preserve this metaphor of interactively

sculpting design curves based on their spatial proximity to a manipulated French curve segment.

With the increasing adoption of pen and touch-based input, however, sketch-based techniques are rapidly gaining in prominence in conceptual design over conventional interfaces that are based on the control point manipulation of high-order spline curves. Sketch curve neatening is thus an important ongoing research problem where various approaches have been proposed. These approaches typically rely on existing families of desirable shapes such as lines, circles, clothoids and cubic splines as shape priors [BLP10, MS09, Far90], to eliminate motor and device noise in sketch curves, while preserving the modeless fluidity of sketching. The term *neatening* here refines the commonly used terms of stroke *smoothing* or *fairing* to include intentional sharp corners and other discontinuities in parts of the sketch.

This paper thus contributes the affordance of French curves, or user customizable shape primitives, to the problem of sketch stroke neatening. In our system, users load a family of pre-authored French curves they wish to use, and then sketch freely. We automatically find an optimal piecewise combination of French curve segments that neatens sketched strokes with user controlled continuity up to G^2 (see Figure 4). Equally important as the creation of fair curves from sketched strokes is the preservation of intended discontinuities (see Figure 3). French curves provide a unique mechanism of filtering discontinuities in tangent and curvature, as it is possible to enforce that only discontinuities encapsulated within the given family of French curves may be permitted in the neatened stroke. We are thus able to neaten sketched strokes modelessly to conform to the aesthetics of a family of French curves without requiring their explicit selection and placement.

From a high-level our approach operates in curvature space in a manner roughly as follows (see Figure 4): The input stroke represented as a polyline is defined in curvature space as a function of discrete curvature at the input points with respect to arc-length along the stroke. The library of French curves is similarly represented. The absolute difference of total curvature between a segment of the input stroke and the French curve is used as an error of fit. A dynamic programming algorithm is used to optimize the trade-off between the number of segments of French curves used and their cumulative error of fit. The selected French curve segments are then individually transformed to match their corresponding input stroke segments. Adjacent segment endpoints that are within a given threshold are blended with G^2 continuity, otherwise the algorithm recursively attempts to improve the fit by introducing an additional segment. This iterative mix of curve fitting in curvature and point space results in an efficient algorithm that both captures the important role of curvature in defining the visual character of curves and bounds the error accumulation in point space as a result of curvature integration. G^1 discontinuities can be captured by either pre-filtering the input stroke into multiple strokes based on curvature discontintuity, or by enforcing that the discontinuity be represented by a French curve segment. A small number of user-controlled parameters allow interactive exploration over continuity and error of fit of the sketched strokes.

The remainder of this paper positions our work in relation to prior art, presents our sketch neatening algorithm and concludes with an evaluation and discussion of the results obtained.

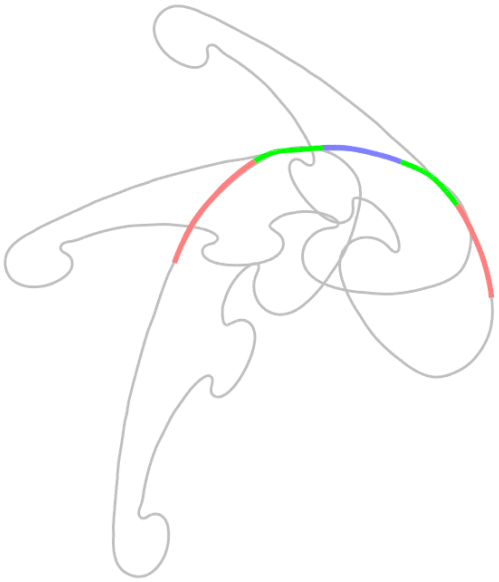

Figure 2: *Showing the alignment of the French curves with each selected piece, in order to illustrate the specific sections of each French curve used.*

2. Related Work

Our work impacts two areas of research in sketch-based geometric modeling: traditional metaphors applied to digital curve design and approaches to the interactive creation, editing and most importantly, neatening of curves.

Sketch and sculpt metaphors are a popular approach to leverage existing domain expertise in the creation of interactive modeling tools. An assortment of such techniques can be found in [Sin06]. The most relevant among these and perhaps the first digital adaptation of French curves is [Sin99], where users interactively manipulate a French curve as a sculpting tool to shape cubic NURBS curves based on their spatial proximity to the French curve. In contrast, we automatically process an entire sketched stroke to be represented by a sequence of optimally-fitting curve segments selected from an entire set of French curves.

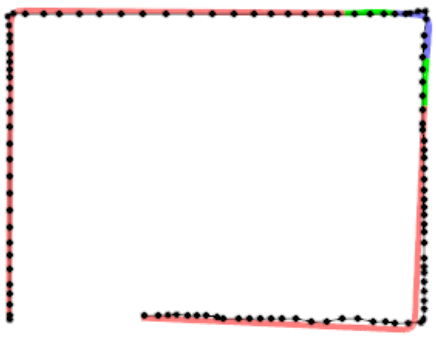

Figure 3: *Fitting an input polyline using a French curve that consists of a single corner piece. Such French curves can be used to neaten deliberate G^1 discontinuities present in the input polyline.*

The research literature on creation, editing, fairing and neatening of curves is extensive and a testament to the importance of the problem. Until a decade ago, virtually all approaches to curve modeling focused on higher-order (typically cubic) splines [Far90]. The control points provide sparse handles for curve manipulation. The curves themselves provided a linear approximation to minimum strain energy point interpolation. Least-squares spline fitting is robust and efficient [Pra87, BBS08] however the curvature plot of the resulting spline can be highly variable. Further, while splines are desirable for representing smooth curves, they are not ideally suited to represent curves that have a mix of smoothness and high-frequency detail. The past decade has seen the advent of alternate curve modeling techniques that operate on densely sampled polyline representations of curves [BLP10, DJBDT10, FLTL08, MS09, GBS03, TBSR04]. These approaches provide different handles and metaphors for curve manipulation and impose smoothess constraints on the polylines by curve primitive fitting or energy minimization. Indeed, minimizing the overall variation of curvature along the curve allows natural shapes like circular arcs, and has been shown to provide better fairness characteristics than cubic splines [MS92]. Rather than attempt to model the aesthetics of the sketched curves using geometric properties of fairness, we allow users to import French curves in order to express their aesthetics explicitly. Philosophically, our approach is similar and complimentary to curve analogies [HOCS02], where curve analogs provided by French curve segments could be used to further stylize our neatened strokes.

From a conceptual standpoint, our work is comparable to [FLTL08] where pre-defined families of shapes are used to neaten sketched strokes. While this approach allows sketched strokes to be subconsciously neatened as the user draws, the templates shown are limited to simple analytical shape families like parallel lines or concentric circles, and must be explicitly activated when sketching strokes based on a given template. In contrast, our approach is modeless matching each sketched stroke to an optimal sequence of segments which are chosen from a potentially large and diverse set of French curves.

Finally, our algorithm is motivated by the curvature space dynamic programming approach of [MS09]. Unlike [MS09], we select segments considering both curvature and position space. We are also not constrained to produce curves with purely linear variation in curvature. This allows us the affordance to individually fit French curve segments to the sketched strokes and blend the transitions between adjacent segments. As a result, curves generated by our algorithm generally conform better to the input stroke.

3. Algorithm

A conceptual overview of our approach is shown in Figure 4. The following subsections detail each of the key components.

3.1. Curvature Estimation

We need to compute curvature profiles for both the given set of French curves as polylines $\mathbf{F}_1 \dots \mathbf{F}_m$ and input polyline $\hat{\mathbf{P}}$.

It is important to first filter the input curve $\hat{\mathbf{P}}$ to remove noise, which will result in a smoother curvature estimation and allow for more aesthetically pleasing results further along. So as a pre-process we iterate through the points $\hat{p}_1 \dots \hat{p}_n$ of the input curve, and for each point $\hat{p}_i$ we consder points within a local neighbourhood (5 units in our implementation) to compute an average point p_i. The next point $\hat{p}_i$ we process will be beyond this neighbourhood, making the spacing between points in the filtered curve more uniform. We ensure that the endpoints of the input curve $\hat{\mathbf{P}}$ are also included in the filtered curve $\mathbf{P}$.

We then compute the curvature profile for $\mathbf{P}$. We calculate the radial curvature value κ for each point of the curve. In the continuous case κ can be expressed using the tangent derivative

$$\frac{d\vec{T}}{ds} = \kappa\vec{N}, \tag{1}$$

where s is the arc-length parameter, and $\vec{T}$ and $\vec{N}$ are the tangent and normal respectively at s.

Since $\mathbf{P}$ is a polyline consisting of an ordered set of discrete points, we rely on an estimation of radial curvature at each of the points. To determine κ at each point p_i, we use

$$\kappa = \frac{2 \cdot \vec{v_1} \times \vec{v_2}}{||\vec{v1}|| + ||\vec{v2}|| + ||\vec{v1} + \vec{v2}||}, \tag{2}$$

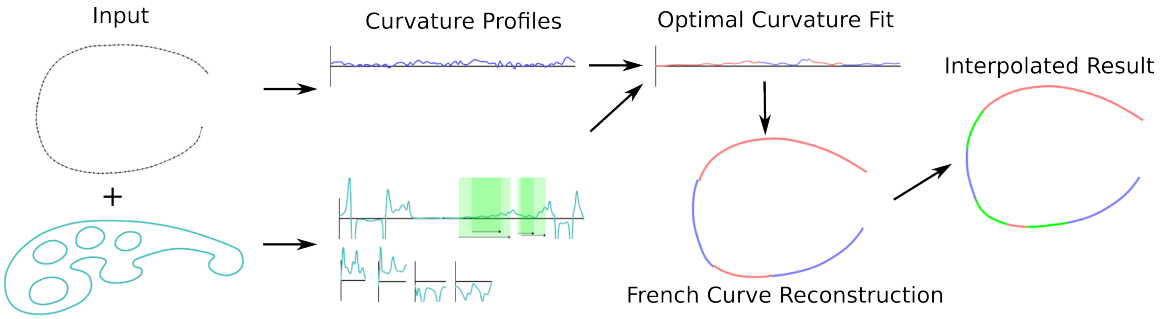

Figure 4: *Overview of the algorithm. Curvature profiles are computed for the input polyline and one or more French curves. A dynamic programming algorithm finds sections of the French curve whose curvature profiles optimally match sections of the input curve, while also trying to minimize the number of sections used. The input curve is reconstructed using sections of the French curve with a piecewise clothoid representation. Finally, disjoint sections of the reconstructed curve are interpolated together to produce a connected curve with G^2 continuity.*

where $\vec{v1} = p_i - p_{i-1}$ and $\vec{v2} = p_{i+1} - p_i$. To handle endpoints p_1 and p_n, we linearly interpolate the curvature from the known interior values.

3.2. Finding Optimal Pieces

Before discussing how we divide the input curve into sections, we first describe how for a given section of the input curve we determine the optimally fitting french curve piece. We present in this order since the division of the input curve depends on how well quantitatively sections of the input curve can be matched with single French curve pieces.

To find the optimal French curve piece for a range of the input curve $p_i \ldots p_j$, of arc length w, for each of the French curves $\mathbf{F}_1 \ldots \mathbf{F}_m$ we move a window of width w by an offset u in discrete steps along each French curve $\mathbf{F}_k$'s curvature profile as shown in Figure 5. At each position u, we calculate a term for *error in fit* $E_{fit}(i,j)$ by integrating the difference in curvature profiles, as expressed by

$$E_{fit}(i,j) = \min_u \int_0^w \mid f(s) - g_k(u+s) \mid ds, \qquad (3)$$

where $f(s)$ expresses the curvature of the input curve $\mathbf{P}$ between i and j, and $g_k(s)$ expresses the curvature for each French curve $\mathbf{F}_k$. In our implementation, we approximate the integral by taking a finite set of samples from f and g_k and sum the absolute value of each pairwise difference.

3.2.1. Flipping

Being physical objects, French curves can be flipped upside down. This is important to note because by doing so, the creator perform a reflection across the tangential axis of any given section of the French curve.

In curvature space, a flip corresponds to a negation of both the curvature values and the arc length direction. This provides a second continuum of possible curves to sample from in reconstructing the input curve. To account for this, we perform a second evaluation of the E_{fit} term at every point u by using the modified equation

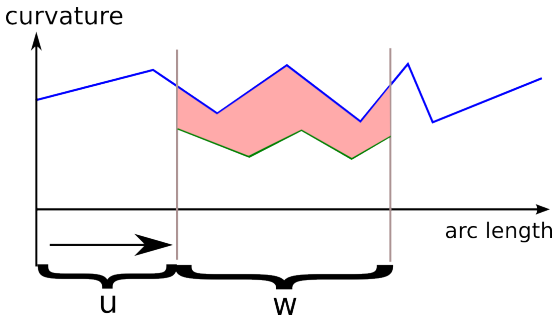

Figure 5: *Given the curvature profile of a section of the input curve (green) of length w, we find the optimal curvature fit by sliding it along each of the French curve curvature profiles (blue) and consider using the section of the French curve where the difference (pink) is minimal.*

$$E_{fit}(i,j) = \min_u \int_0^w \mid f(s) + g_k(u+w-s) \mid ds. \qquad (4)$$

Note that we use addition instead of subtraction within the integral, and we sample from g_k in the opposite direction.

3.2.2. Closed Curves

In the physical world, the French curve must always be closed. It follows that our system should support these curves as well.

Without addressing this, our approach will never choose a

window that contains the first and last points of any French curve (for a closed curve, the selection of first and last point is arbitrary in defining the curve, but there will always necessarily be some ordering of the points).

To adapt our approach of finding the optimal piece to a closed curve, we repeat the curvature profile a second time, and move the window along a length twice the arc length of each French curve $\mathbf{F}_k$, as shown in Figure 6. This ensures that the window passes through sections of the French curve that contain the first and last point.

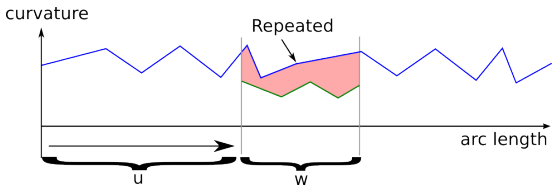

Figure 6: *To support closed French curves, we repeat the French curve curvature profile (blue) a second time and slide the window across the point of repetition in order to evaluate all possible sections of the closed curve.*

3.3. Splitting the Curve

To find the globally optimal segmentation of the curve, we use a dynamic programming algorithm. An important parameter to our algorithm is the scalar E_{cost} term, which defines the cost of using a French curve piece. This parameter allows interactive calibration of the tradeoff between using many French curve pieces (to represent the input curve more faithfully), and using few French curve pieces (to create a simpler, fairer curve that potentially has greater aesthetic appeal) as shown in Figure 7.

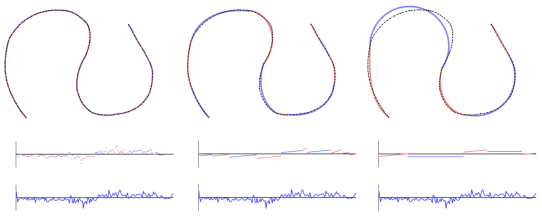

Figure 7: *Changing the value of E_{cost} influences the number of pieces selected. E_{cost} values from left to right:* 0, 0.2, 0.4. *The top row shows the input polyline and generated French curve pieces. The middle row shows curvature profiles for the French curve pieces. The bottom row shows the curvature profile of the input polyline.*

We compute the upper triangle of a matrix $\mathbf{M}$. $\mathbf{M}$ is of size $n \times n$, where the input polyline $\mathbf{P}$ consists of n points. We compute entries $\mathbf{M}(i,j)$, where $1 \leq i < j \leq n$, in a bottom-up fashion (starting at elements closest to the diagonal) using

$$\mathbf{M}(i,j) = \min \left\{ E_{fit}(i,j) + E_{cost}, \min_{i<k<j} \{\mathbf{M}(i,k) + \mathbf{M}(k,j)\} \right\}, \tag{5}$$

which determines whether it is best to use a single French curve piece (cost of $E_{fit}(i,j) + E_{cost}$) for the entire section $p_i \dots p_j$, or to subdivide and use one (or more) pieces for each of the intervals $p_i \dots p_k$ and $p_k \dots p_j$. Information such as where to split the input curve, which of the French curves $\mathbf{F}_1 \dots \mathbf{F}_m$ to sample from and at what arc-length position, are all stored as $\mathbf{M}$ is populated.

When this process is completed, $\mathbf{M}(1,n)$ expresses the cost of the optimal configuration for points $p_1 \dots p_n$ of $\mathbf{P}$. We "walk through" the matrix $\mathbf{M}$ starting at element $(1,n)$, splitting where necessary to reconstruct the solution.

3.4. Reconstruction

Given the curvature profile of each optimal French curve piece, we form each of the pieces in point space as a piecewise clothoid curve. Each linear segment of the curvature profile maps to a unique clothoid segment. These clothoid segments are attached so position and tangent are continuous, resulting in a curve with G^2 (curvature) continuity. Consult the appendix for more details on clothoids, as well as the equations used in our implementation to evaluate clothoid curves by an arc-length parameterization.

We now have an ordered set of French curve pieces represented as piecewise clothoid, but these pieces are not yet aligned in space or connected. We translate and then rotate each French curve piece to match its endpoints with the input polyline points at the start and end of the section (i.e. the piece which covers points $p_i \dots p_j$ will be transformed so its endpoints align as closely as possible with p_i and p_j).

Since the French curve piece is only a local approximation of the input curve, significant deviation in point space is possible. A given French curve piece may not be acceptable, despite being the best selection of those possible. The problem is worsened by the fact that the error in curvature is integrated twice in moving from curvature space to point space.

In order to validate that the piece selected will be adequate, we ensure that the piece endpoints and polyline endpoints have a distance between them which does not exceed some threshold (in our implementation we use the value 4.0, but this threshold can be modified interactively), indicating that the piece conforms well to the input polyline. If the threshold is exceeded, we discard the piece and split the section $p_i \dots p_j$ optimally into two smaller sections that will each be assigned one piece (or more, if those pieces selected are also inadequate). At the end of this process, all selected pieces conform well at the endpoints to their input polyline sections.

The final step is to connect each of these pieces. In order to maintain G^2 continuity, we use the polynomial function

$$f(s) = s^3(6s^2 - 15s + 10), \tag{6}$$

to interpolate between points sampled from adjacent French curve pieces. Having a continuous second derivative, and the following properties: $f(0) = f'(0) = f''(0) = 0$, $f(1) = 1$, $f'(1) = f''(1) = 0$, and $f(\frac{1}{2}) = \frac{1}{2}$, causes adjacent French curve pieces to be joined with G^2 continuity.

The user can control the arc-length distance to extend outward from the French curve piece endpoints for blending, as shown in Figure 8. Interpolation between points of two adjacent pieces occurs over this interval.

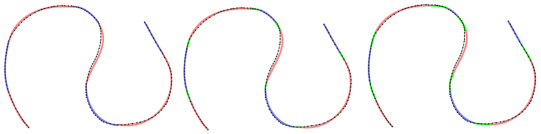

Figure 8: *French curve pieces (red and blue) are locally blended with their neighbours (green) using an interpolating function that results in a G^2 curve. The arc-length distance of the blend can be interactively modified. From left to right: 0 (no blend), 10 and 20 units.*

To close input polylines, we use interpolation on the end pieces of the reconstructed curves, as shown in Figure 9.

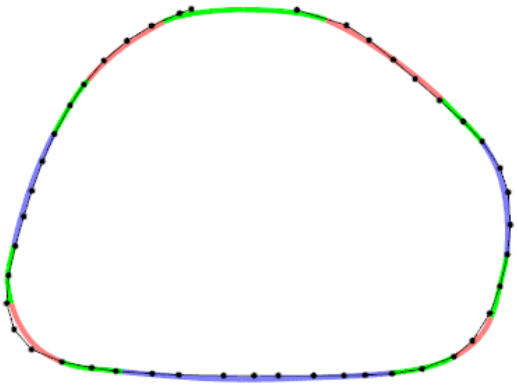

Figure 9: *When the endpoints of the input polyline are in close proximity (top of the curve), we close the curve by smoothly interpolating between the two French curve pieces at the endpoints.*

4. Results and Discussion

In Figure 11, we show some piecewise French curves generated by our algorithm on a variety of input strokes. For all of these results, the E_{cost} parameter was fixed at 0.2. Note that an amount of deviation from the input curve is allowed to take place, where the algorithm balances between using few French curve pieces and precisely matching the input curve. Figure 12 shows how our approach can be used to add artistic effects to input strokes.

One limitation of our approach stems from the specific function we have used to calculate the error in fit term E_{fit} (Equation 4) for a given French curve piece on an input curve piece. A French curve piece with a zig-zag pattern, for example, will have a curvature profile with pronounced alternating positive and negative peaks. If the arc-length spacing between these peaks is not consistent between the French curve and input stroke (which may result from using longer or shorter lines between each discontinuity), shifting in the arc-length direction will cause the two profiles to be mis-aligned with one another, leading to a high fitness error value. (A close analogy would to be consider aligning two sinusoidal curves with differing wavelengths.) An alternate formulation of the error in fit which would not penalize arc-length shifting between points of curvature maxima would improve the selection process for curves with pronounced features (i.e. structured G^1 discontinuities).

Another limitation to our algorithm is the use of dynamic programming to find a globally optimal segmentation of the input curve. Being $O(n^2)$ in performance, the algorithm will take significantly longer to process especially long curves. The issue is alleviated by filtering the input curve, which subsamples it and results in fewer points to process.

Exploring this limitation further, we also implemented a greedy approach, which iterates along the input curve attempting to use a progressively larger piece, at each step determining if the error in fit exceeds a threshold. When the threshold is exceeded, a split point to separate pieces one point back is defined. A new piece is created which starts at the split point and iteration then continues. This approach offers linear time $O(n)$ performance and scales well for longer input polylines. An important downside of the approach is that the solution produced is suboptimal in terms of global curvature fit. In Figure 10 we time the algorithm on inputs of various sizes and compare the results.

5. Conclusion

French curves provide a useful mechanism to convey contextual information for neatening sketched strokes.

Architectural drawings for example, might use a set of French curves with straight lines and angles that will tend to create rectilinear curves from sketches, while an automotive set of French curves may be completely devoid of straight

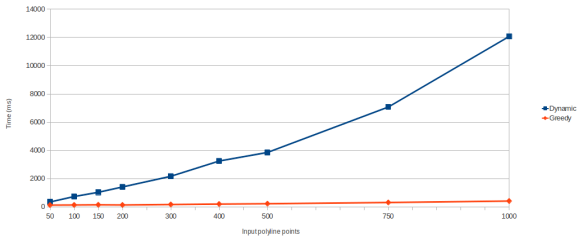

Figure 10: *Processing time for our dynamic programming approach compared to a greedy method, for a variety of input sizes.*

line shapes and instead comprise circles for wheels and other characteristic curve profiles to be used across a fleet of designs. The use of French curves in this paper is also modeless, in that once loaded the set of French curves are invisible to the user, and implicitly neaten sketched strokes, preserving the modeless simplicity of pure sketching. Our interactive implementation shows them to be successful both at neatening sketched strokes and effective in imposing a visual characteristic on the resulting design curves.

One drawback of our sketch neatening algorithm and all other approaches based on curve primitive fitting is that an entire stroke must be completed before the neatened stroke is computed and displayed. This makes our approach unsuitable for sketch applications such as feature tracking or performance animation, where neatened curves must be committed as the user draws. The greedy version of our algorithm is better in this regard as it can continually commit neatened French curve segments as the user draws, tightening the visual loop between the user's sketch and the final output.

French curves also extend conceptually to manifold surfaces and in the future we would like to explore their application to conceptual surface modeling.

Appendix A: Clothoids

The Fresnel integrals provide an arc-length parameterization of the Cartesian coordinates for points along the spiral:

$$C(s) = \int_0^s \cos \frac{\pi}{2} u^2 du, \tag{7}$$

$$S(s) = \int_0^s \sin \frac{\pi}{2} u^2 du. \tag{8}$$

A point on the clothoid curve is defined as

$$\pi B \begin{pmatrix} C(s) \\ S(s) \end{pmatrix}, \tag{9}$$

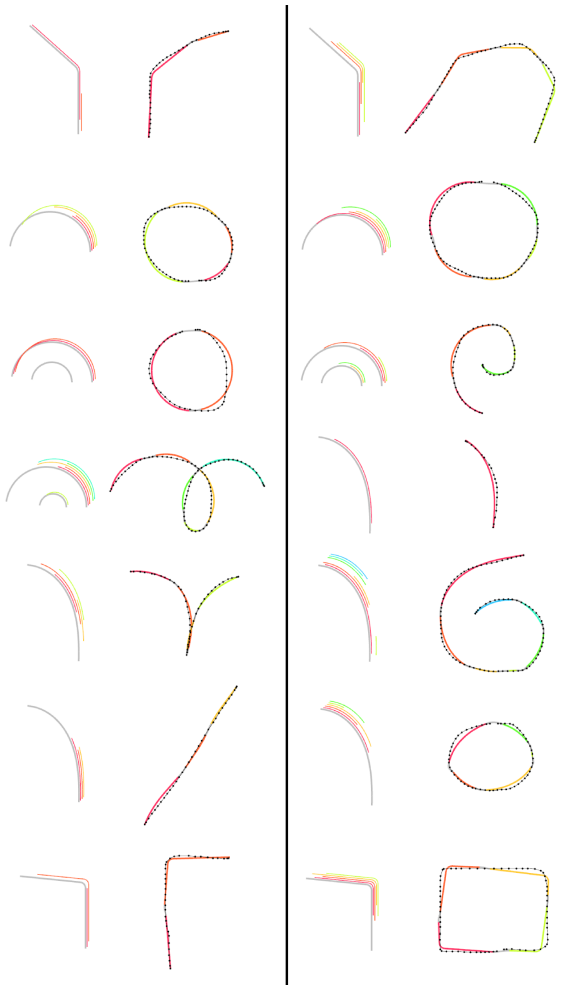

Figure 11: *Results generated using our system, organized into pairs. For each pair, at left the French curve(s) and the optimal pieces selected are shown, and on the right are the input and neatened output curves. The colour correspondence show what pieces of the French curve(s) were used and where they are on the output curve. A translation has been applied to each overlaid French curve piece so that they are all visible.*

where B is a uniform scaling parameter. Used by our implementation, rational low-order approximations for the integrals [Hea85] are given by

$$C(s) \approx \frac{1}{2} - R(s) \sin\left(\frac{\pi}{2}(A(s) - s^2)\right), \tag{10}$$

$$S(s) \approx \frac{1}{2} - R(s) \cos\left(\frac{\pi}{2}(A(s) - s^2)\right), \tag{11}$$

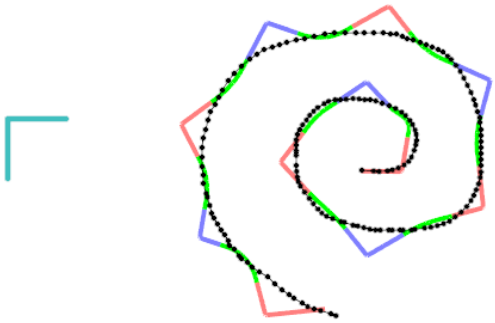

Figure 12: *The single French curve piece on the left is used with artistic effect to reconstruct a spiral-shaped input polyline.*

where

$$R(s) = \frac{0.506s + 1}{1.79s^2 + 2.054s + \sqrt{2}}, \tag{12}$$

$$A(s) = \frac{1}{0.803s^3 + 1.886s^2 + 2.524s + 2}. \tag{13}$$

As their curvature varies linearly with arc-length, clothoids are line segments in curvature space. Given a line segment in curvature space, the horizontal difference between points is the arc-length l, and the vertical positions of the two points specify the start and end curvatures κ_1 and κ_2. This curvature space information is enough to solve for the unique clothoid segment with these properties. In this formulation, we solve for B, s_1 and s_2 (start and end arc-lengths), which together define the section of the clothoid curve to use:

$$B = \sqrt{\frac{l}{\kappa_2 - \kappa_1}}, \tag{14}$$

$$s_1 = \kappa_1 B \tag{15}$$

$$s_2 = \kappa_2 B \tag{16}$$

References

[BBS08] BAE S.-H., BALAKRISHNAN R., SINGH K.: Ilovesketch: as-natural-as-possible sketching system for creating 3d curve models. In *Proceedings of the 21st annual ACM symposium on User interface software and technology* (New York, NY, USA, 2008), UIST '08, ACM, pp. 151–160. 3

[BLP10] BARAN I., LEHTINEN J., POPOVIĂĞ J.: Sketching clothoid splines using shortest paths. *Computer Graphics Forum 29* (2010), 655–664. 2, 3

[DJBDT10] DEROUET-JOURDAN A., BERTAILS-DESCOUBES F., THOLLOT J.: Stable inverse dynamic curves. In *ACM SIGGRAPH Asia 2010 papers* (New York, NY, USA, 2010), SIGGRAPH ASIA '10, ACM, pp. 137:1–137:10. 3

[Far90] FARIN G.: *Curves and surfaces for computer aided geometric design*. Academic Press Professional, Inc., San Diego, CA, USA, 1990. 2, 3

[FLTL08] FUNG R., LANK E., TERRY M., LATULIPE C.: Kinematic templates: end-user tools for content-relative cursor manipulations. In *Proceedings of the 21st annual ACM symposium on User interface software and technology* (New York, NY, USA, 2008), UIST '08, ACM, pp. 47–56. 3

[GBS03] GROSSMAN T., BALAKRISHNAN R., SINGH K.: An interface for creating and manipulating curves using a high degree-of-freedom curve input device. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2003), ACM, pp. 185–192. 3

[Hea85] HEALD M. A.: Rational approximations for the fresnel integrals. *Mathematics of Computation 44*, 170 (1985), 459–461. 7

[HOCS02] HERTZMANN A., OLIVER N., CURLESS B., SEITZ S. M.: Curve analogies. In *Proceedings of the 13th Eurographics workshop on Rendering* (Aire-la-Ville, Switzerland, Switzerland, 2002), EGRW '02, Eurographics Association, pp. 233–246. 3

[KFBB97] KURTENBACH G., FITZMAURICE G., BAUDEL T., BUXTON B.: The design of a gui paradigm based on tablets, two-hands, and transparency. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 1997), CHI '97, ACM, pp. 35–42. 1

[KS08] KARA L. B., SHIMADA K.: Supporting early styling design of automobiles using sketch-based 3d shape construction. *Computer-Aided Design & Applications 5*, 6 (2008), 867–876. 1

[McC71] MCCONALOGUE D. J.: Algorithm 66 - an automatic french-curve procedure for use with an incremental plotter. *Computer Journal 14* (1971), 207–209. 1

[MS92] MORETON H. P., SÉQUIN C. H.: Functional optimization for fair surface design. *SIGGRAPH Comput. Graph. 26*, 2 (1992), 167–176. 3

[MS09] MCCRAE J., SINGH K.: Sketch-based interfaces and modeling (sbim): Sketching piecewise clothoid curves. *Comput. Graph. 33* (August 2009), 452–461. 2, 3

[Pra87] PRATT V.: Direct least-squares fitting of algebraic surfaces. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1987), ACM, pp. 145–152. 3

[Sin99] SINGH K.: Interactive curve design using digital french curves. In *I3D '99: Proceedings of the 1999 symposium on Interactive 3D graphics* (New York, NY, USA, 1999), ACM, pp. 23–30. 1, 2

[Sin06] SINGH K.: Industrial motivation for interactive shape modeling: a case study in conceptual automotive design. In *ACM SIGGRAPH 2006 Courses* (New York, NY, USA, 2006), SIGGRAPH '06, ACM, pp. 3–9. 1, 2

[TBSR04] TSANG S., BALAKRISHNAN R., SINGH K., RANJAN A.: A suggestive interface for image guided 3d sketching. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems* (New York, NY, USA, 2004), ACM, pp. 591–598. 3