κ-Curves: Interpolation at Local Maximum Curvature

ZHIPEI YAN, Texas A&M University
STEPHEN SCHILLER, Adobe Research
GREGG WILENSKY, Adobe
NATHAN CARR, Adobe Research
SCOTT SCHAEFER, Texas A&M University

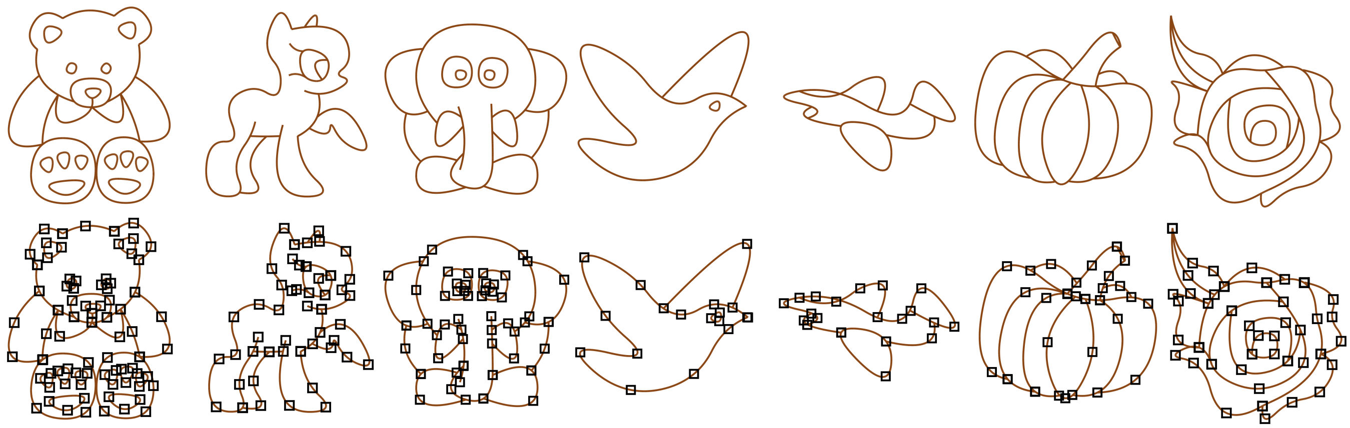

Fig. 1. Top row shows example shapes made from the control points below. In all cases, local maxima of curvature only appear at the control points, and the curves are G^2 almost everywhere.

We present a method for constructing almost-everywhere curvature-continuous, piecewise-quadratic curves that interpolate a list of control points and have local maxima of curvature only at the control points. Our premise is that salient features of the curve should occur only at control points to avoid the creation of features unintended by the artist. While many artists prefer to use interpolated control points, the creation of artifacts, such as loops and cusps, away from control points has limited the use of these types of curves. By enforcing the maximum curvature property, loops and cusps cannot be created unless the artist intends for them to be.

To create such curves, we focus on piecewise quadratic curves, which can have only one maximum curvature point. We provide a simple, iterative optimization that creates quadratic curves, one per interior control point, that meet with G^2 continuity everywhere except at inflection points of the curve where the curves are G^1. Despite the nonlinear nature of curvature, our curves only obtain local maxima of the absolute value of curvature only at interpolated control points.

CCS Concepts: •**Computing methodologies → Parametric curve and surface models;**

Additional Key Words and Phrases: interpolatory curves, monotonic curvature, curvature continuity

This work was supported by NSF Career award IIS 1148976.

DOI: http://dx.doi.org/10.1145/3072959.3073692

ACM Reference format:
Zhipei Yan, Stephen Schiller, Gregg Wilensky, Nathan Carr, and Scott Schaefer. 2017. κ-Curves: Interpolation at Local Maximum Curvature. *ACM Trans. Graph.* 36, 4, Article 129 (July 2017), 7 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073692

1 INTRODUCTION

Curve modeling has a long history in computer graphics, finding use in drawing, sketching, data fitting, interpolation, as well as animation. This rich application space has led to decades of research for both representing and modifying curves. The goal of such curve representations is to provide the user with control over the shape of the curve while building a curve that has certain geometric properties. These properties may include smoothness, interpolation of various points, and locality.

In this paper we focus on interpolatory curves; that is, curves that interpolate their control points. While much research has concentrated on approximating curves, many users prefer direct control over salient geometric features of the curve such as the position of the curve. Yet interpolatory curves have a maligned past as they can often generate geometric features such as cusps and loops away from control points that the user has a hard time controlling (see Figure 2).

Our premise is that salient geometric features should appear only at control points for interpolatory curves. Position is one such example of a feature that is automatically enforced in interpolatory curve constructions. However, the question is then: what other features should appear only at control points? Levien et al. [Levien

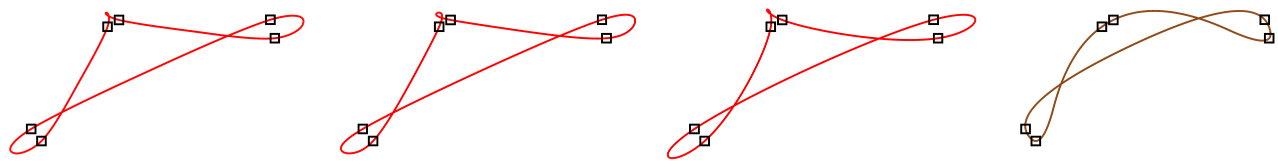

Fig. 2. Comparison of our result with other C^2 curves. From left to right: 6-point interpolatory subdivision curve [Deslauriers and Dubuc 1989], C^2 Catmull-Rom spline [Catmull and Rom 1974], C^2 interpolating cubic B-spline [Farin 2002], our curve.

and Séquin 2009] argue that points of maximal curvature are also salient features. Indeed, we can see that lack of control of points of maximal curvature has led to many of the historical problems with interpolatory curve constructions. For example, the propensity to produce cusps is due to a local maximum of curvature (in this case, infinite curvature) being produced away from the control points.

We propose to create interpolatory curves where the local maximum of the absolute value of the curvature of the curve only appears at control points, which we call κ-curves. In addition, we build such curves using piecewise quadratic curves that meet with G^2 continuity everywhere except at inflection points, where the join is G^1. We should point out that the primary application envisioned for these splines is more for artistic design, as opposed to CAD. Thus the occasional lack of G^2 continuity is not an issue. Also, because we envision these curves to be controlled by artists, the curve should change continuously under continuous motion of the control points. This last property is trivially satisfied by many curve constructions, but not all. For example, Figure 3 shows an example of continuous movement of control points for a clothoid curve [Havemann et al. 2013] that produces a discontinuous change of the resulting curve.

2 RELATED WORK

There are large numbers of interpolatory curve constructions that have been developed, and we cannot provide an exhaustive list but refer to [Hoschek and Lasser 1993] for many such methods. Catmull-Rom splines [Barry and Goldman 1988; Catmull and Rom 1974] are one of the more common interpolatory curve representations and are combinations of Lagrange interpolation with B-spline basis functions. Subdivision curves [Deslauriers and Dubuc 1989; Dyn et al. 1987] can also be used to model interpolatory splines. Cubic splines, formed from approximating B-splines, interpolate points with C^2 continuity through the solution to a tridiagonal system of equations [Farin 2002]. Different parameterizations, such as centripetal or chordal, can be used to control the shape of these curves as well. In the case of C^1 Catmull-Rom splines, such a choice can guarantee that no cusps appear except at control points [Yuksel et al. 2011]. However, these results do not extend to C^2 Catmull-Rom splines. While all of these constructions build interpolatory curves, even with curvature continuity, none allow control over curvature. Figure 2 shows a comparison of many of these methods versus our construction. Note that all of these curves create cusps or have local maximum curvature points away from control points except for our curve.

More related to our method are classes of curves that, not only interpolate control points, but control curvature in some way. Higashi et al. [Higashi et al. 1988] restricted the locations of control points of Bézier curve to obtain monotonic curvature and create a C^2 spline. “Class A” Bézier curves [Farin 2006; Mineur et al. 1998] have monotonic curvature, but few degrees of freedom and can be difficult to control. Clothoids, also known as Euler spirals, [Havemann et al. 2013; McCrae and Singh 2009; Schneider and Kobbelt 2000] are perhaps the best known such curve. These curves have the property that the curvature of the curve changes linearly with respect to arc length. Hence, piecewise clothoid curves have local maximum curvature magnitude at the interpolated points. Such curves would be ideal for our purposes except that continuous motion of the control points does not always create a continuous deformation of the curve as is illustrated in Figure 3. Log-aesthetic curves [Miura and Gobithaasan 2014; Miura et al. 2013; Yoshida et al. 2009; Yoshida and Saito 2017] are similar to clothoids (indeed clothoids are a special case) and have curvature plots that increase exponentially with respect to arc length. Levein et al. [Levien and Séquin 2009] also describe a two parameter spline family modulo conformal transformations.

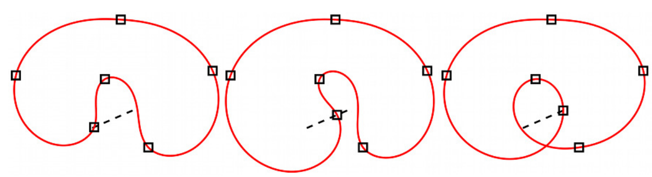

Fig. 3. Moving one control point continuously for a piecewise clothoid curve can result in a discontinuous change in the curve as shown on the right where the curve suddenly flips over.

In addition to controlling curvature, our method uses piecewise quadratic curves that meet with G^2 continuity. Most curve constructions require cubic curves to generate C^2 or G^2 curves, but G^2 quadratic curves have appeared in the past. Schaback [Schaback 1989] created a piecewise quadratic G^2 Bézier curve to interpolate a list of non-inflecting points. This approach creates a quadratic Bézier curve between interpolated points but tends to produce flat curves at the interpolated points. Feng et al. [Feng and Kozak 1996] modified this approach and build a G^2 quadratic curve to interpolate a list of points with associated tangent directions where the end points of each quadratic appear between interpolated points. Gu et al. [Gu et al. 2009] used quadratic Bézier curves to interpolate a list of points with arbitrary tangent directions with G^1 continuity.

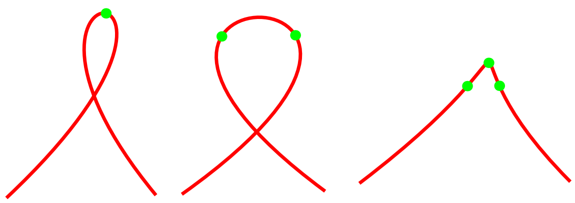

Fig. 4. Cubic curves with different number of local maximum curvature points. From left to right: a cubic curve with one, two, and three local maximum curvature points highlighted in green.

Our approach to creating G^2 quadratic curves differs from all these approaches, and we develop an explicit solution of the join point between two quadratics to enforce G^2 continuity. In addition, we consider the added condition that control points are interpolated at maximal curvature magnitude locations.

3 GEOMETRIC CONSTRAINTS

We begin by considering the geometric properties that we require of these curves. Specifically, given an ordered set of points $p_1 \dots p_n \in \mathbb{R}^2$, we would like to construct a curvature continuous curve (G^2 curve) such that the curve interpolates the p_i; that is, there exists parameters t_i such that $p(t_i) = p_i$. We make the additional assumption that these points form a closed curve and should be treated cyclically (we relax this assumption in Section 4).

Furthermore, we would like any local maximums of the curvature magnitude (the absolute value of curvature) to exist only at the p_i. This last criterion is based on the assumption that points at which the curve bends the most, at least locally, are salient features of the curve and should be under direct control of the user. Notice that this last property does not imply that the derivative of the curvature magnitude must be zero at every p_i. Hence, curvature may be locally increasing or decreasing at a interpolated point p_i. However, if a maximum of the curvature magnitude exists, it appears at an interpolated point.

3.1 Interpolation at Local Maximums of Curvature

Like many curve methods, we focus on piecewise polynomial curves as our curve representation. We represent our curves in Bézier form. Given a set of control points $c_{i,j}$, the i^{th} Bézier curve of degree d is given by

$$c_i(t) = \sum_{j=0}^{d} \frac{d!}{(d-j)!j!}(1-t)^{d-j}t^j c_{i,j}$$

where $t \in [0, 1]$.

However, controlling curvature for polynomial curves is difficult at best. Cubic parametric curves can have three local maxima of the curvature magnitude in any given segment. Figure 4 shows several parametric cubic Bézier curves with the local maxima of curvature magnitude highlighted. While "class A" Bézier curves exist [Farin 2006] and have monotonic curvature, they are extremely restrictive and have few degrees of freedom.

Given that maximal curvature is so difficult to control for polynomial curves, we opt for an extremely simple representation for our curves, which we construct out of piecewise quadratic Bézier curves; one curve ($c_i(t)$) for each interpolated point p_i. Individual quadratic curves have the property that they possess at most one point of maximum curvature, which is crucial for our application.

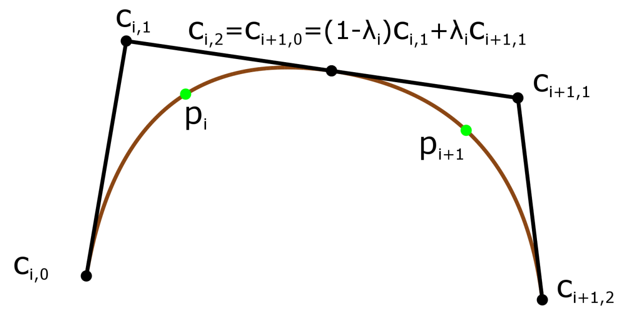

Fig. 5. The notation for our control points $c_{i,0\dots2}$, interpolated input points p_i and the G^2 join condition

Let $c_{i,0}, c_{i,1}, c_{i,2} \in \mathbb{R}^2$ be the three control points for the quadratic curve $c_i(t)$. The curvature of this curve is given by

$$\begin{aligned} \kappa_i(t) &= \frac{\det(\frac{\partial c_i(t)}{\partial t}, \frac{\partial^2 c_i(t)}{\partial t^2})}{||\frac{\partial c_i(t)}{\partial t}||^3} \\ &= \frac{\Delta(c_{i,0}, c_{i,1}, c_{i,2})}{||(1-t)(c_{i,1}-c_{i,0}) + t(c_{i,2}-c_{i,1})||^3} \end{aligned} \tag{1}$$

where Δ gives the area of the triangle specified by its arguments. Taking the derivative of $\kappa_i(t)$ and setting the equation equal to zero gives the parameter t_i for the point of maximal curvature in terms of the Bézier coefficients for the i^{th} Bézier curve

$$t_i = \frac{(c_{i,0}-c_{i,1}).(c_{i,0}-2c_{i,1}+c_{i,2})}{||c_{i,0}-2c_{i,1}+c_{i,2}||^2}. \tag{2}$$

Now to construct a curve $c_i(t)$ such that $c_i(t_i) = p_i$, we will show that, for any $c_{i,0}$, $c_{i,2}$, and p_i, there exists a choice of $c_{i,1}$ such that p_i is interpolated at the point of maximal curvature. Starting with the condition

$$c_i(t_i) = p_i,$$

we solve for the point $c_{i,1}$ and obtain

$$c_{i,1} = \frac{p_i - (1-t_i)^2 c_{i,0} - t_i^2 c_{i,2}}{2t_i(1-t_i)}. \tag{3}$$

Substituting Equation 3 into Equation 2 results in a cubic equation in terms of t_i.

$$\begin{aligned} &||c_{i,2}-c_{i,0}||^2 t_i^3 + 3(c_{i,2}-c_{i,0}).(c_{i,0}-p_i)t_i^2 \\ &+(3c_{i,0}-2p_i-c_{i,2}).(c_{i,0}-p_i)t_i - ||c_{i,0}-p_i||^2 = 0. \end{aligned} \tag{4}$$

While this equation could have three real roots, which would mean the solution is not unique, we show in Appendix A that this equation has exactly one real root in $[0, 1]$ for any choice of $c_{i,0}, p_i, c_{i,2}$. Given that there is exactly one root of this cubic, finding the root is simple, and there are many ways to do so. We use the exact formula for roots of a cubic in our implementation. Even in the degenerate case where all three points form a straight line (i.e; $p_i = (1-\alpha)c_{i,0} +$

$\alpha c_{i,2}$), the cubic trivially has one root of $t_i = \alpha$. Once we have t_i, substituting this value into Equation 3 completes the quadratic curve that interpolates p_i at the point of local maximum curvature magnitude.

3.2 Smoothness

While we have discussed a local construction for interpolating points where the curve has maximum curvature magnitude, we aim to piece these curves together to form a curvature continuous (i.e; G^2 curve). Note that it is not possible to create a G^2 everywhere curve using piecewise quadratics if the sign of curvature changes along the curve. The reason is that quadratic curves cannot possess zero curvature unless the curve is trivially a straight line. Hence, unless our curves are strictly convex, we cannot hope to build piecewise quadratic curves that are G^2 everywhere.

Our compromise is to build piecewise quadratic curves that are G^2 *almost* everywhere. The only place where our curves will lose continuity of curvature will be at points where the curve changes from convex to concave or vice versa. Hence, unlike standard spline constructions, the geometric smoothness of our piecewise construction changes dependent on the *geometry* of the curve instead of how the curve is decomposed into polynomial pieces. Conditions for joining quadratic curves with G^2 smoothness have been discussed before [Schaback 1989]. However, given that it is not well known that building curvature continuous quadratic curves is even possible, we derive the conditions here and provide a closed-form solution, which will then be part of our optimization in Section 4.

Our curves consist of one quadratic curve, with control points $c_{i,0}, c_{i,1}, c_{i,2}$, per interpolated point p_i. The C^0 continuity conditions between curves are trivial and simply require that $c_{i,2} = c_{i+1,0}$. G^1 continuity is simple as well and requires that

$$c_{i,2} = (1 - \lambda_i)c_{i,1} + \lambda_i c_{i+1,1} \tag{5}$$

where $\lambda_i \in (0, 1)$.

For G^2 continuity, we consider the convex curve in Figure 5. Since $c_{i,2}$ is a linear combination of $c_{i,1}$ and $c_{i+1,1}$, the question is if there is a choice of $\lambda_i \in [0, 1]$ that leads to curvature continuity. G^2 continuity requires $\kappa_i(1) = \kappa_{i+1}(0)$. Using Equation 1 and writing the G^2 condition in terms of the control points and λ_i yields

$$\frac{\Delta(c_{i,0}, c_{i,1}, c_{i+1,1})}{|c_{i,1} - c_{i+1,1}|^3 \lambda_i^2} = \frac{\Delta(c_{i,1}, c_{i+1,1}, c_{i+1,2})}{|c_{i,1} - c_{i+1,1}|^3 (1 - \lambda_i)^2}. \tag{6}$$

This condition creates a quadratic equation in terms of λ_i that has exactly one root in $(0, 1)$, which is

$$\lambda_i = \frac{\sqrt{\Delta(c_{i,0}, c_{i,1}, c_{i+1,1})}}{\sqrt{\Delta(c_{i,0}, c_{i,1}, c_{i+1,1})} + \sqrt{\Delta(c_{i,1}, c_{i+1,1}, c_{i+1,2})}}.$$

When the convexity of the curve changes, the curve cannot be G^2 since quadratic curves cannot have zero curvature unless they degenerate to a line. In this case, we choose λ_i by minimizing the difference of the curvature magnitude squared

$$\min_{\lambda_i} (|\kappa_i(1)| - |\kappa_{i+1}(0)|)^2.$$

Since the curvatures of are opposite signs, such a minimization is equivalent to solving $\kappa_i(1) + \kappa_{i+1}(0) = 0$ for λ_i. Using the expressions

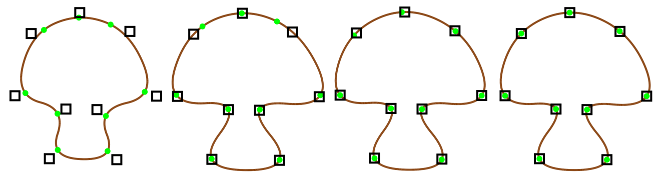

Fig. 6. Iterations of our optimization showing convergence with control points (black boxes) and maximum curvature positions (green dots). From left to right: our initial guess, after 1 iteration, after 2 iterations, and final convergence after 30 iterations.

for curvature in Equation 6, we again find one root for the quadratic equation in $(0, 1)$ given by

$$\lambda_i = \frac{\sqrt{|\Delta(c_{i,0}, c_{i,1}, c_{i+1,1})|}}{\sqrt{|\Delta(c_{i,0}, c_{i,1}, c_{i+1,1})|} + \sqrt{|\Delta(c_{i,1}, c_{i+1,1}, c_{i+1,2})|}}. \tag{7}$$

Note that this equation is identical to the previous expression for λ_i except for the absolute value. Hence, this expression unifies both cases. In the purely convex case, this choice of λ_i yields a G^2 curve whereas the curve is G^1 when the convexity of the curve changes. However, in this case, the absolute value of curvature (but not the sign of curvature) matches at the join between curves.

The above equations for for λ_i will be undefined if both triangle areas in the denominators are zero, which can happen when enough of the $c_{i,j}$ are co-linear or coincident. This case can be robustly handled by adding a small constant, $\epsilon = 10^{-10}$, to the square roots of each such area.

4 OPTIMIZATION

Using the geometric conditions in Section 3, we combine these conditions together to generate an optimization to find a curve that satisfies these properties. To do so, we adopt a local/global approach [Liu et al. 2008; Sorkine and Alexa 2007]. Our degrees of freedom in the optimization are the off-the-curve points $c_{i,1}$ since $c_{i,2} = c_{i+1,0}$ by the C^0 condition and $c_{i,2} = (1 - \lambda_i)c_{i,1} + \lambda_i c_{i+1,1}$ by the G^1 and G^2 conditions.

Given the current solution for $c_{i,0,\ldots,2}$, we perform a local step and estimate the λ_i using Equation 7 and then update the $c_{i,0}$ and $c_{i,2}$ using Equation 5. Next we compute the maximum parameters t_i from Equation 4. For the first iteration, we use the initial guess that $\lambda_i = \frac{1}{2}$ and $c_{i,1} = p_i$. Using these locally computed values, we assume the t_i and λ_i are constant and solve a global, linear system for the $c_{i,1}$ such that (1) $c_i(t_i) = p_i$ and (2) the constraints from Equation 5 hold. These constraints lead to linear equations for each p_i in the unknowns $c_{i-1,1}, c_{i,1}, c_{i+1,1}$ of the form:

$$\begin{aligned} p_i = \quad & (1 - \lambda_{i-1})(1 - t_i)^2 c_{i-1,1} + \lambda_i t_i^2 c_{i+1,1} + \\ & (\lambda_{i-1}(1 - t_i)^2 + (2 - (1 + \lambda_i)t_i)t_i)c_{i,1}. \end{aligned}$$

Solving this circulant, tridiagonal system of linear equations leads to updated positions for the $c_{i,1}$. We then repeat this solving process until convergence.

This optimization converges quickly and each iteration is fast to compute since we only solve a small, sparse linear system of equations. Figure 6 shows the progress of our optimization. After

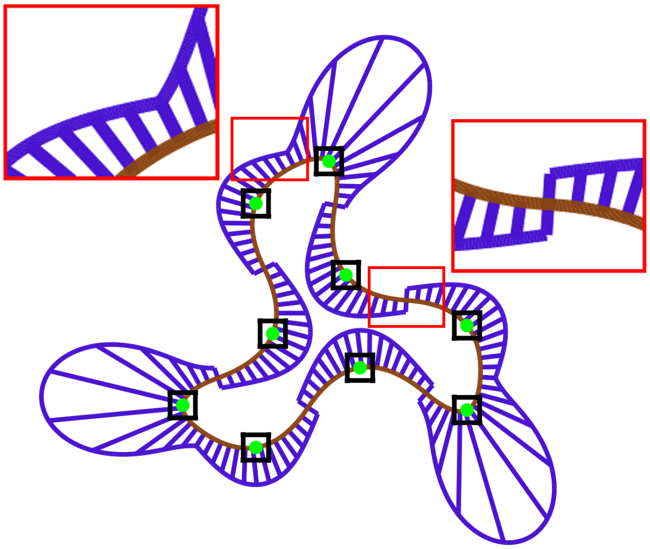

Fig. 7. Our curve (brown) shown with control points as block boxes. Green points are positions of local maximal curvature magnitude. We also draw the curvature normal for the curve (purple). Our curve is G^2 everywhere as shown in the highlighted region except at inflection points where the curve is G^1 and the sign, but not magnitude, of the curvature changes.

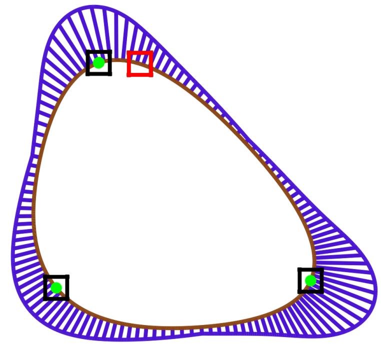

Fig. 8. The red control point is not at a critical point of curvature, which is decreasing. However, all local maxima of curvature magnitude appear at control points.

just one iteration, our result is very close to the final solution, but some maximum curvature points do not coincide with control points yet. After two iterations, the result is nearly indistinguishable from our final result. Even for large curves with many control points, our optimization yields results beyond interactive speeds due to its simplicity. Furthermore, it is possible to make the optimization even faster by starting with the results from the user's previous control point configuration, although we have found such improvements to be unnecessary.

4.1 Curves with Boundaries

Our discussion has concentrated on closed curves. Handling curves with end-points is a simple modification to our current approach. Let $p_{1\ldots n}$ be interior control points and p_0, p_{n+1} be the end-points of the curve. For the curve $c_1(t)$, which interpolates p_1, we simply add the requirement that $c_{1,0} = p_0$. Likewise, we constrain $c_{n,2} = p_{n+1}$. Such a change makes sure that p_0 and p_{n+1} do *not* appear at local maximum curvature points, but instead are points of local minimal curvature. Figure 9 shows an example of a curve with end-points using this method.

5 RESULTS

Our curves are designed to have maximal curvature magnitude at the control points p_i. Figure 7 shows the control points as hollow boxes and displays all local maxima of curvature magnitude as green points on the curve. Hence, green points should appear within each control point box for our curve. These are the points where the curve, locally, bends the most. Despite its complex shape, no cusps or loops exist in the curve. Figure 1 demonstrates shapes composed of many curves. In all cases, local maxima of curvature magnitude only appear at control points.

Figure 7 also displays the continuity of the curve through the curvature normal (the normal whose length is proportional to curvature of the curve at that point). For a curvature continuous (G^2) curve, the magnitude of the curvature normal should change continuously over the curve. This is true for our curve everywhere except at the inflection points of the curve. At these points the curvature normal flips orientation but maintains the same magnitude.

Note that it is possible that a control point does not exist at a maximum curvature point as demonstrated in Figure 8. In this case, the curvature is decreasing at the highlighted point. However, all points of maximum curvature magnitude appear at control points.

Unlike curves such as clothoids, continuous motion of the control points results in continuous deformation of the curve. Clothoids use estimates of curvature that change continuously with motion of the control points. When moving from positive to negative curvature, the curve *must* pass through a point of infinite positive curvature to a point of infinite negative curvature (a cusp) in order for the geometry of the curve to change continuously. Clothoids vary curvature piecewise linearly between control points and cannot possess such behavior. In contrast, our curves can generate infinite curvature, though only at control points. Having unbounded curvature is not a unwanted artifact but precisely the property that creates continuous motion of the curve. However, the cost of this continuous motion is a curvature profile that is not as "fair" as clothoids. Figure 9 depicts the creation of a cusp at a control point as the user manipulates the curve with our method. This figure, and many of the shapes in Figure 1, also demonstrate open curves with end points.

Since our optimization produces a global solution, the influence of one control point is technically global. That is, moving one point changes the shape of the entire curve. However, practically, the influence of one point is quite bounded. Figure 10 shows an example curve where we move a single control point and draw the original curve (blue) behind the new curve (brown). The figure illustrates that very little movement of the curve occurs outside of just a few control points away from the modified shape.

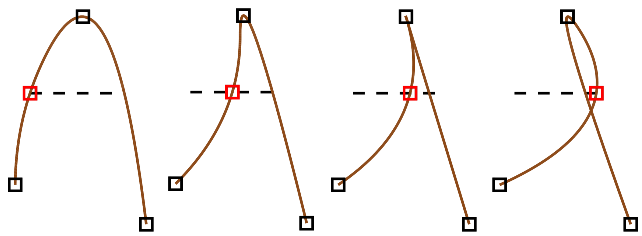

Fig. 9. The creation of a cusp, which can only happen at control points with our method.

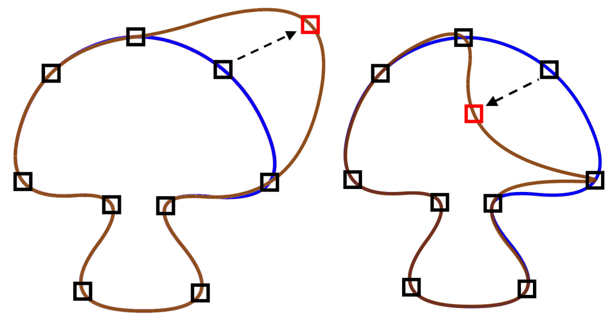

Fig. 10. The original curve in blue with the new curve in brown drawn on top. While control points have a global influence on the curve, that influence drops dramatically with distance from the control point, which creates the effect of local influence.

6 IMPLEMENTATION

The κ-curve system was implemented as a tool in a commercial illustration package, Adobe Illustrator[1]. Adobe Illustrator is instrumented to report the amount of time users spend using each of the available tools. The new tool, called the "curvature" tool, is a direct competitor for the much older "pen" tool that uses cardinal splines. Even though there are experienced and exacting artists with many years invested in the the pen tool, six months after the release of the curvature tool 35% of the combined use of the two tools was with the newer curvature tool on desktop devices. On devices with touch screens, the curvature tool captured 66% of the combined use of the two tools. While more studies are needed to fully understand artists preferences, this usage data strongly suggests that our model is a welcome addition to professional work–flows.

7 CONCLUSIONS AND FUTURE WORK

Interpolatory curves typically suffer from shape artifacts and use high degree polynomials with large support for even low continuity curves. Our curves change this paradigm. We use low degree curves, yet achieve higher-order continuity. Despite their global nature, we have demonstrated that the influence of an input point is local for practical applications. Moreover, the input points coincide with features of the curve; namely, local maxima of curvature magnitude. Even though the user has no direct control over the tangent angle or curvature at the input points, the system automatically chooses natural values for these parameters. In fact one may see this work a way of automatically choosing natural and pleasing tangent angles and curvatures based only the input points. Finally, our curves change continuously under continuous motion of the control points. These two combined properties make them ideal for a number of creative tasks including vector design and motion path keyframing. In summary, we believe that κ-curves solve many of the problems that have plagued interpolatory curves, rendering them much more suitable for modern design applications.

While we have focused on non-rational quadratic Bézier curves here, this work could be extended rational quadratic curves. The extra degrees of freedom could conceivably allow for an independent "tension" control at each interpolated point, allowing more artistic control of the shape of the curve. However, rational curves would not solve the problem of lack of G^2 continuity at inflection points.

[1] The actual implementation in Adobe Illustrator differs slightly from what was presented in this paper due to compatibility and other issues, but the underlying technology is essentially the same.

REFERENCES

Phillip J. Barry and Ronald N. Goldman. 1988. A Recursive Evaluation Algorithm for a Class of Catmull-Rom Splines. In *Proceedings of SIGGRAPH*. 199–204.
E. Catmull and R. Rom. 1974. A class of local interpolating splines. *Computer aided geometric design* (1974), 317–326.
Gilles Deslauriers and Serge Dubuc. 1989. Symmetric iterative interpolation processes. *Constructive Approximation* 5, 1 (1989), 49–68.
Nira Dyn, David Levin, and John A. Gregory. 1987. A 4-point Interpolatory Subdivision Scheme for Curve Design. *Computer Aided Geometric Design* 4, 4 (1987), 257–268.
Gerald Farin. 2002. *Curves and Surfaces for CAGD: A Practical Guide* (5th ed.). Morgan Kaufmann Publishers Inc.
Gerald Farin. 2006. Class A Bézier curves. *Computer Aided Geometric Design* 23, 7 (2006), 573–581.
Yu Yu Feng and Jernej Kozak. 1996. On G^2 continuous interpolatory composite quadratic Bézier curves. *J. Comput. Appl. Math.* 72, 1 (1996), 141–159.
He-Jin Gu, Jun-Hai Yong, Jean-Claude Paul, and Fuhua Frank Cheng. 2009. Constructing G^1 quadratic Bézier curves with arbitrary endpoint tangent vectors. *International Journal of CAD/CAM* 9, 1 (2009).
Sven Havemann, Johannes Edelsbrunner, Philipp Wagner, and Dieter Fellner. 2013. Curvature-controlled curve editing using piecewise clothoid curves. *Computers & Graphics* 37, 6 (2013), 764–773.
Masatake Higashi, Kohji Kaneko, and Mamoru Hosaka. 1988. Generation of high-quality curve and surface with smoothly varying curvature. In *EG Technical Papers*. Eurographics Association.
Josef Hoschek and Dieter Lasser. 1993. *Fundamentals of Computer Aided Geometric Design*. A. K. Peters, Ltd.
Jeffrey M. Lane and R. F. Riesenfeld. 1981. Bounds on a polynomial. *BIT Numerical Mathematics* 21, 1 (1981), 112–117.
Raph Levien and Carlo H Séquin. 2009. Interpolating Splines: Which is the fairest of them all? *Computer-Aided Design and Applications* 6, 1 (2009), 91–102.
Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. 2008. A Local/Global Approach to Mesh Parameterization. In *Proceedings of the Symposium on Geometry Processing*. 1495–1504.
James McCrae and Karan Singh. 2009. Sketching piecewise clothoid curves. *Computers & Graphics* 33, 4 (2009), 452–461.
Yves Mineur, Tony Lichah, Jean Marie Castelain, and Henri Giaume. 1998. A shape controled fitting method for Bézier curves. *Computer Aided Geometric Design* 15, 9 (1998), 879–891.
Kenjiro T Miura and RU Gobithaasan. 2014. Aesthetic curves and surfaces in computer aided geometric design. *International Journal of Automation Technology* 8, 3 (2014), 304–316.
Kenjiro T. Miura, Dai Shibuya, R. U. Gobithaasan, and Shin Usuki. 2013. Designing Log-aesthetic Splines with G^2 Continuity. *Computer-Aided Design and Applications* 10, 6 (2013), 1021–1032.
Robert Schaback. 1989. Interpolation with piecewise quadratic visually C^2 Bézier polynomials. *Computer Aided Geometric Design* 6, 3 (1989), 219–233.
Robert Schneider and Leif Kobbelt. 2000. *Discrete fairing of curves and surfaces based on linear curvature distribution*. Technical Report. DTIC Document.
Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible Surface Modeling. In *Proceedings of the Symposium on Geometry Processing*. 109–116.
Norimasa Yoshida, Ryo Fukuda, and Takafumi Saito. 2009. Log-aesthetic Space Curve Segments. In *SIAM/ACM Conference on Geometric and Physical Modeling*. 35–46.

Norimasa Yoshida and Takafumi Saito. 2017. Quadratic log-aesthetic curves. *Computer-Aided Design and Applications* 14, 2 (2017), 219–226.
Cem Yuksel, Scott Schaefer, and John Keyser. 2011. Parameterization and applications of Catmull–Rom curves. *Computer-Aided Design* 43, 7 (2011), 747–755.

A UNIQUENESS OF MAX CURVATURE SOLUTION

PROOF. To show Equation 4 has exactly one real root in $[0, 1]$, we first write its coefficients in Bézier form, which leads to an extremely simple expression. Let $v_0 = c_{i,0} - p_i$ and $v_2 = c_{i,2} - p_i$. Then the Bézier coefficients are

$$(-|v_0|^2, \frac{-v_0 \cdot v_2}{3}, \frac{v_0 \cdot v_2}{3}, |v_2|^2)$$

Given that we are interested in roots of this polynomial, we divide by the positive constant $|v_0||v_2|$ to obtain an even simpler expression for the coefficients

$$(-r, -\frac{\cos(\theta)}{3}, \frac{\cos(\theta)}{3}, \frac{1}{r}) \tag{8}$$

where $r = \frac{|v_0|}{|v_2|}$ and θ is the angle between the vectors v_0 and v_2. Note that polynomials in Bézier form follow Descartes' rule of signs for bounding the number of roots of a polynomial [Lane and Riesenfeld 1981]. Since the first coefficient is negative while the last coefficient is positive, there exists at least one root on the interval $[0, 1]$. If $\cos(\theta) \geq 0$, then Descartes' rule of signs indicates exactly one root in $[0, 1]$. Therefore, we consider the case where $\cos(\theta) < 0$.

However, under this assumption, the polynomial with Bézier coefficients from Equation 8 is monotonically increasing over the interval $[0, 1]$. We verify this fact by computing the derivative of the polynomial, which yields the quadratic Bézier coefficients

$$(3r - \cos(\theta), 2\cos(\theta), \frac{3}{r} - \cos(\theta)).$$

Both the first and the last coefficient are positive since $\cos(\theta) < 0$ and $r > 0$. Furthermore, we can compute the minimum value of this quadratic, which is

$$\frac{3r - \cos(\theta)(1 + r^2 + r\cos(\theta))}{1 + r^2 - 2r\cos(\theta)}.$$

This value is also strictly positive because $1 + r^2 + r\cos(\theta) > 0$. Since the derivative of the quadratic at 0 is $6(\cos(\theta) - r)$, which is negative, the cubic function with Bézier coefficients in Equation 8 is monotonically increasing and has exactly one real root in $[0, 1]$. ◻