Title:
Visualization of the Curvature Monotonicity Regions of Polynomial Bézier Curves and its Application to Curve Design

Authors:
Norimasa Yoshida, yoshida.norimasa@nihon-u.ac.jp, Nihon Univeristy
Seiya Sakurai, cise20009@g.nihon-u.ac.jp, Nihon University
Hikaru Yasuda, cihi22003@g.nihon-u.ac.jp, Nihon University
Taisei Inoue, inoue.taisei@nihon-u.ac.jp, Nihon University
Takafumi Saito, txsaito@cc.tuat.ac.jp, Tokyo University of Agriculture and Technology

Keywords:
curvature monotonicity region, polynomial Bézier curves, curve design

DOI: 10.14733/cadconfP.2023.1-5

Introduction:

Freeform curves, such as Bézier curves or B-splines curves, are widely used in many applications, such as Illustration software and CAD systems. Although Bézier curves and B-spline curves have many nice properties, controlling the curvature variation by manually moving control points is not easy. For quadratic Bézier curves, Sapidis et al. clarified the theoretical region of a control point where the curvature becomes monotonically varying [4]. For cubic polynomial Bézier curve, Dietz et al. proposed a method for generating curves with monotonically varying curvature using precomputed tables [1]. Various methods have been proposed to generate curves with monotonically varying curvature. Some of them are class A Bézier curves [2] and pseudo-log-aesthetic curves [7]. For Bézier curves of degree 3 or higher, the curvature monotonocity region, which is the region of a control point where the curvature becomes monotonically varying, has not been visualized before. By visualizing the curvature mononotonicity region in real time, a user can know where to move the control point to make the curvature monotonically varying.

Recently, Yan et al. proposed κ-curves [6], which are interpolating quadratic Bézier curves having local maxima of curvature only at interpolating points. Miura et al. extended the method to cubic curves so that the method have additional control by α. In both approaches, the curve shape cannot be modified locally. A method that can control the curve shape locally without introducing another curvature maxima or minima is desirable.

In the present work, for polynomial Bézier curves, we present a real-time method for visualizing the region of a specific control point where the curvature becomes monotonically varying. Therefore, when the region is visible, a user can know where to move the specific control point to make the curvature monotonically varying. We also present two applications of the proposed approach.

Checking the monotonicity of curvature:

Let $\mathbf{P}(t)$ be a regular planar parametric curve. If $\mathbf{P}(t)$ is a Bézier of degree n,

$$\mathbf{P}(t) = \sum_{i=0}^{n} B_i^n(t)\mathbf{P}_i \tag{2.1}$$

where $B_i^n(t)$ are Bernstein polynomials and $\mathbf{P}_i$ are control points. Curvature monotonicity can be checked if $\frac{d\kappa}{ds}$ does not change its sign within $t \in [0,1]$. For planar curves, $\frac{d\kappa}{ds}$ can be computed by the following equation [5]:

$$\frac{d\kappa}{ds} = \frac{\det(\dot{\mathbf{P}}, \dddot{\mathbf{P}})\dot{\mathbf{P}} \cdot \dot{\mathbf{P}} - 3\,\det(\dot{\mathbf{P}}, \ddot{\mathbf{P}})\dot{\mathbf{P}} \cdot \ddot{\mathbf{P}}}{|\dot{\mathbf{P}}|^6} \tag{2.2}$$

where $\dot{\mathbf{P}} = \frac{d\mathbf{P}}{dt}$, $\ddot{\mathbf{P}} = \frac{d^2\mathbf{P}}{dt^2}$, and $\dddot{\mathbf{P}} = \frac{d^3\mathbf{P}}{dt^3}$. Assuming the curve is regular, the first derivative does not vanish. In other words, the denominator of Eq. (2.2) is always positive. Therefore, checking if $\frac{d\kappa}{ds}$ changes its sign reduces to check if the numerator of $\frac{d\kappa}{ds}$ changes its sign.

Let the numerator of Eq. (2.2) be $\lambda(t)$:

$$\lambda(t) = \det(\dot{\mathbf{P}}, \dddot{\mathbf{P}})\dot{\mathbf{P}} \cdot \dot{\mathbf{P}} - 3\,\det(\dot{\mathbf{P}}, \ddot{\mathbf{P}})\dot{\mathbf{P}} \cdot \ddot{\mathbf{P}} \tag{2.3}$$

For a polynomial curve of degree n, the degree of $\lambda(t)$ is $4n-7$. Since $\lambda(t)$ is a polynomial, it can be represented in Bernstein basis as

$$\lambda_B(t) = \sum_{i=0}^{4n-7} B_i^{4n-7}(t)\xi_i. \tag{2.4}$$

In [5], for cubic Bézier curves, the sufficient condition of $\xi_i \geq 0 (i = 0, \ldots, 5)$ or $\xi_i \leq 0 (i = 0, \ldots, 5)$ is used to guarantee the monotonicity of curvature. For more strict check of the curvature monotonicity, the following algorithm is used.

Algorithm 1: Curvature Monotonicity Check

(1) If $\xi_i \geq 0 (i = 0, \ldots, 4n-7)$, the part of the curve is judged to be monotonically increasing. If $\xi_i \leq 0 (i = 0, \ldots, 4n-7)$, the part of the curve is judged to be monotonically decreasing.

(2) If $\xi_0 \cdot \xi_{4n-7} < 0$, the curvature is judged to be NOT moronically varying.

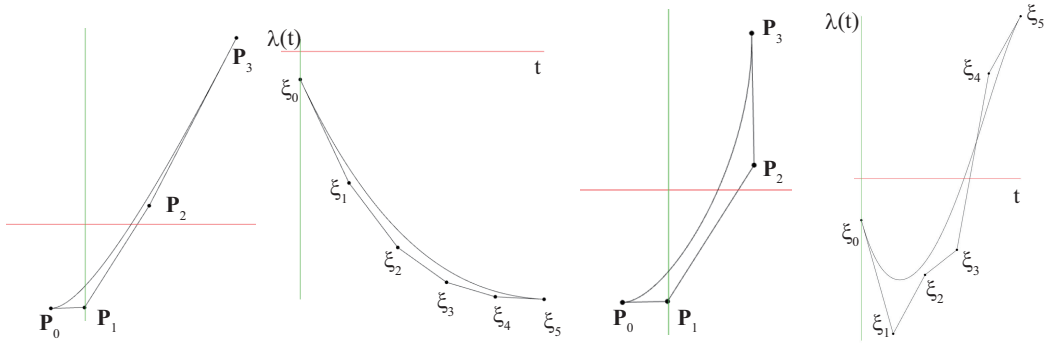
(3) Recursively subdivide the curve at $t = 0.5$, until the condition (1) is satisfied for all subdivided parts of the curve. If the condition (2) is satisfied or the recursion reaches to the user-specified depth, the curvature is judged to be NOT monotonically varying.

Fig 1 (a) shows a cubic Bézier curve with monotonically varying curvature and its $\lambda(t)$. Since ξ_i are all negative, the curve is judged to be monotonically varying (decreasing). Fig 1 (b) shows a cubic Bézier curve with NOT monotonically varying curvature. Since $\xi_0 \cdot \xi_5 < 0$, the curve is immediately judged to be NOT monotonically varying.

Visualization of monotone curvature regions of Bézier curves:

To visualize the curvature monotonicity region of a specific control of a Bézier curve, we compute ξ_i in Eq. (2.4) and checks the curvature monotonicity using Algorithm 1 for all the cases where the specific control point is placed at every pixel in the screen window. For efficiency, the computation is performed using a GPU. In the fragment shader, ξ_i are computed and the curvature monotonicity is checked by Algorithm 1. Depending on the curvature is monotonically increasing, monotonically decreasing, or NOT monotonically varying, the corresponding pixel is painted with user-specified colors.

Fig. 2 show the curvature monotonicity region of a cubic Bézzier curve. In Fig. 2(a) or (b), if $\mathbf{P}_0$ is in the blue region, the curvature become monotonically decreasing. If $\mathbf{P}_0$ is in the red region, the curvature becomes monotonically increasing. Fig. 2(c) or (d) shows the region for $\mathbf{P}_1$. We have implemented the visualization of the curvature monotonicity regions of Bézier curves of degree 4 or higher, but the monotonicity region becomes generally smaller as the degree gets higher.



(a) A curve with monotonically varying curvature and its λ(t)

(b) A curve with NOT monotonically varying curvature and its λ(t)

Fig. 1: Cubic Bézier curves and their $\lambda(t)$

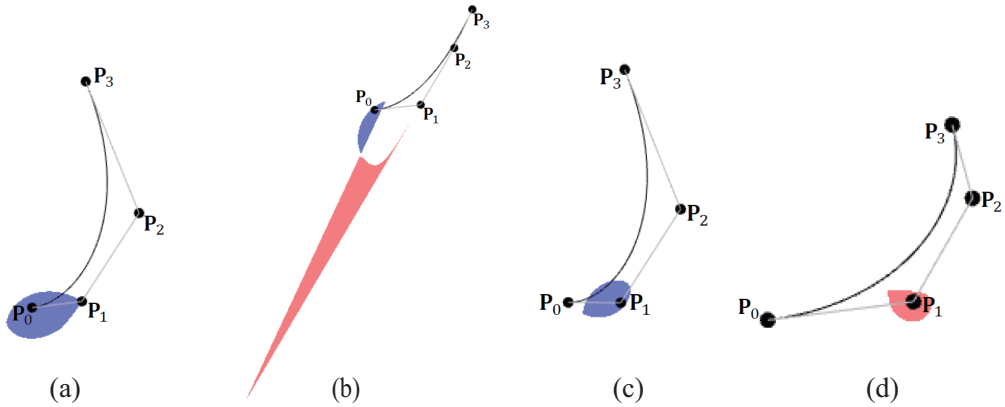


Fig. 2: The curvature monotonicity regions. (a) and (b) are the regions of $\mathbf{P}_0$. (c),(d) are those of $\mathbf{P}_1$.

Application 1: Curve Design:

We have implemented a curve design tool like the one in Adobe Illustrator and added the visualization of the curvature monotonicity regions. Fig. 3 shows an example. The dark red region near Q_3 indicates that if Q_3 is in the region, the curvature of the cubic Bézier curve defined by Q_1, Q_2, Q_3 and Q_4 becomes monotonically varying. The cyan region near Q_5 is the same for the curve defined by Q_4, Q_5, Q_6, Q_7. When Q_4 is moved, both Q_3 and Q_5 are moved accordingly. If Q_4 is placed within the purple region, both the curvature of the curve defined by Q_1, Q_2, Q_3 and Q_4 and the curvature of the curve defined by Q_4, Q_5, Q_6 and Q_7 become monotonically varying. Fig. 4(a) shows an apple designed without visualizing the curvature monotonicity region. Fig. 4(b) to (f) are the process of modifying the curve shape so that the curvature becomes monotonically varying within a use-specified curve segment. Fig. 4(g) shows the final design.

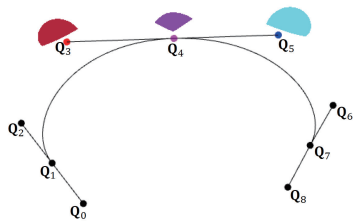


Fig. 3: Visualization of monotone curvature regions in a curve design tool.

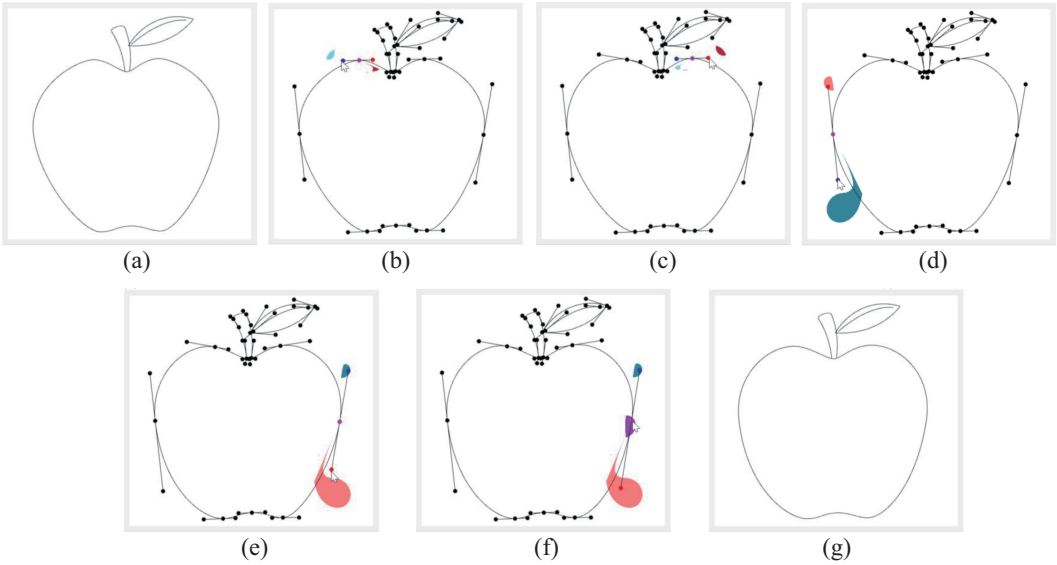


Fig. 4: Design of an apple shape. (a) is the original shape created without visualizing the curvature monotonicity regions. (b) to (f) are the process of modifying the shape so that the curvature becomes monotonically varying within a user-specified curve segment. (g) is the shape of the final design.

Application 2: Modifying the shape of the curve generated by κ-curves:

Yan et al. proposed κ-curves that are interpolatory curves where local maxima of curvature are placed only at control points. Visualization of curvature monotonicity region can be applied to modify the curve shape generated by κ-curves. We first subdivide the curve generated by κ-curves at the curvature maxima using the de Casteljau's algorithm. Since quadratic Bézier curves are used in κ-curves, the parameter value at the curvature maxima can be easily computed. Then we degree elevate the curves to cubic Bézier curves. As shown Fig. 5(f), $\mathbf{P}_{n-1}$ and $\mathbf{P}_{n+1}$ can be modified within the blue or red regions without introducing another curvature extremum. If $\mathbf{P}_{n-1}$, $\mathbf{P}_n$, $\mathbf{P}_{n+1}$ lies in a straight line, G^1 continuity is guaranteed. Note that almost G^2 continuity is guaranteed in κ-curves. Although the continuity becomes G^1 where we modify the curve shape, it may not be a problem in many illustration applications. Our approach provides local modification of curve shape without introducing another curvature extremum.

Conclusions:

In this work, we presented a real time method to visualize the curvature monotonicity regions of poly-

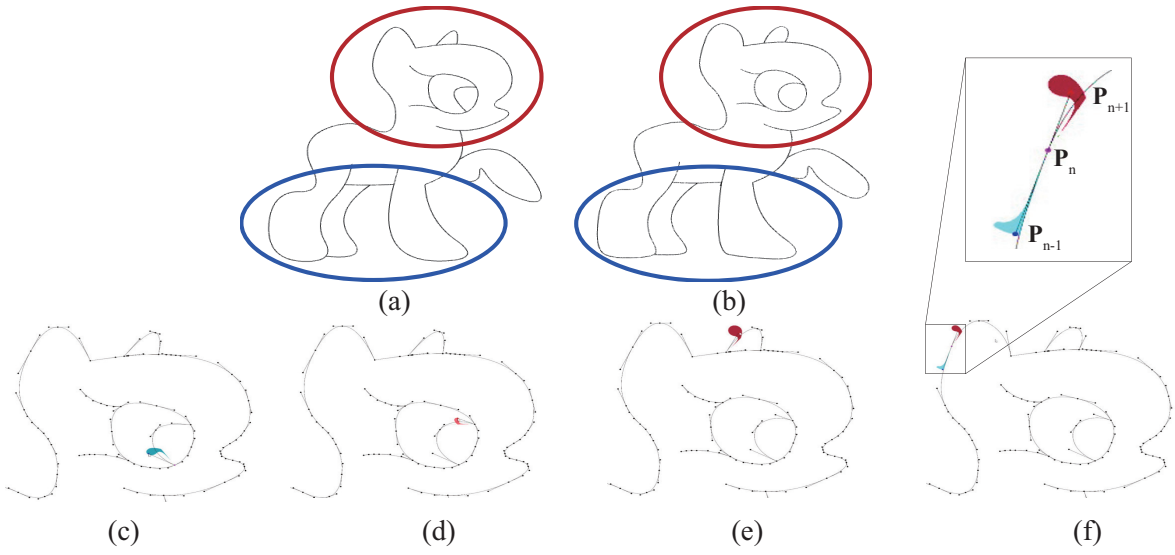


Fig. 5: Locally modifying the shape (a) generated by κ-curves to (b) using the proposed approach without introducing another curvature extremum. (c) to (f) are intermediate processes.

nomial curves. We show that the curve shape generated by κ-curves can be modified locally without introducing another curvature extremum with G^1 continuity. We also show that our approach can be immediately applied to a curve design tool, such as the one in Adobe Illustrator.

The proposed approach can be applied to B-spline curves and rational curves but succinct and efficient representation of $\lambda(t)$ in Bernstein form is required. We are currently working to apply the idea to 3D curves.

References:

[1] Dietz, A. D.; Piper B.: Interpolation with cubic spirals, Computer Aided Geometric Design, 21(2), 2004, 165-180. https://doi.org/10.1016/j.cagd.2003.09.002

[2] Gerald Farin, Class A Bézier curves, Computer Aided Geometric Design, 23(7), 2006, 573-581. https://doi.org/10.1016/j.cagd.2006.03.004

[3] Miura, K. T.; Gobithaasan, R. U.; Salvi, P.; Wang, D.; Sekine, T.; Usuki, S.; Inoguchi J.; Kajiwara, K.: $\epsilon\kappa$-Curves: controlled local curvature extrema, The Visual Computer, 38, 2022, 2723-2738. https://doi.org/10.1007/s00371-021-02149-8

[4] Sapidis, N. S.; Frey, W. H.: Controlling the curvature of a quadratic Bézier curve, Computer Aided Geometric Design, 9, 1992, 85-91. https://doi.org/10.1016/0167-8396(92)90008-D

[5] Wang, Y.; Zhao, B.; Zhang, L.; Xu, J.; Wang, K.; Wang, S.; Designing fair curves using monotone curvature pieces, Computer Aided Geometric Design, 21(5), pp. 515-527 (2004). https://doi.org/10.1016/j.cagd.2004.04.001

[6] Yan, Z.; Schiller, S.; Wilensky, G.; Carr, N.; Schaefer, S.: κ-curves: Interpolation at local maximum curvature, ACM Transaction on Graphics, 36(4), 2017, 1-7. https://doi.org/10.1145/3072959.3073692

[7] Yoshida, N.; Saito, T.: Quasi-Aesthetic Curves in Rational Cubic Bézier Forms, Computer-Aided Design & Applications, 4(1-4), 2007, 477-486. https://doi.org/10.1080/16864360.2007.10738567