www.cad-journal.net Computer-Aided Design and Applications

Curvature Monotonicity Regions of 2D Polynomial and Rational Bézier Curves as the Intersection of Implicit Regions

Norimasa Yoshida[1], Takafumi Saito[2]

[1]Nihon University, yoshida.norimasa@nihon-u.ac.jp
[2]Tokyo University of Agriculture and Technology, txsaito@cc.tuat.ac.jp

Corresponding author: Norimasa Yoshida, yoshida.norimasa@nihon-uac.jp

Abstract. In this study, we visualize the curvature monotonicity regions for 2D polynomial and rational Bézier curves based on the sufficient condition. These regions are represented as the intersection of half-spaces defined by implicit curves. We verify that the degrees of implicit curves are up to cubic, for polynomial and rational Bézier curves up to degree 10. Using GPU technology, we can visualize regions, including implicit curves, in real time. Visualization of the regions, including implicit curves, partly elucidates how the curvature monotonicity regions are constructed.

Keywords: curvature monotonicity region, Bézier curves, visualization of implicit curves, GPU
DOI: https://doi.org/10.14733/cadaps.2025.68-80

1 INTRODUCTION

Freeform curves, such as Bézier curves and B-spline curves, exhibit numerous desirable properties and find wide-ranging applications. However, determining the placement of control points to achieve monotonically varying curvature poses a challenge. Although CAD systems offer tools to visualize the curvature comb and observe changes resulting from control point adjustment, identifying the specific region of a control point that ensures monotonically varying curvature remained unclear. In our previous work [21, 10], we introduced a real-time method to visualize the monotonicity regions of the curvature of polynomial and rational curves. Using the method, users can know the region of a control point to achieve monotonically varying curvature. However, investigations into these regions have remained unexplored.

In our current study, we visualize the regions of curvature monotonicity of 2D polynomial and rational Bézier curves, relying on a sufficient condition to check curvature monotonicity. Hereafter, we refer to the curvature monotonicity region based on the sufficient condition as a *sufficient region*. Using GPU technology, we propose a real-time approach for visualizing sufficient regions, including the implicit curves that bound the sufficient region. Implicit curves are the coefficients of the curvature monotonicity evaluation function, denoted ξ_i in Eq. (4).

This paper makes the following contributions:

1. We propose a real-time visualization method of sufficient regions as the intersection of half-spaces defined by related implicit curves.
2. We propose a GPU-based real-time method for visualizing the half spaces defined by implicit curves.
3. We elucidate that the sufficient region is composed as the intersection of half-spaces defined by implicit curves up to degree 3. We verified this fact for polynomial and rational Bézier curves up to degree 10, using Mathematica.
4. Visualization of the sufficient region partially elucidates how the curvature monotonicity regions are constructed.

The structure of this paper is as follows: Section 2 provides a review of related work. Section 3 reviews the curvature monotonicity evaluation functions [10]. Section 4 introduces a method for real-time visualization of regions defined by implicit curves using a GPU. This method is used to visualize curvature monotonicity regions on the basis of a sufficient condition. Section 5 proposes a method to visualize sufficient regions, including the implicit curve. We show results for cubic and quartic polynomial, as well as rational cubic Bézier curves. Section 6 presents the conclusions.

2 RELATED WORK

Numerous studies have tackled the generation of freeform curves with monotonically varying curvature. The theoretical foundation for curvature monotonicity regions has been established for quadratic polynomial Bézier curves by Sapidis et al. [12] and quadratic rational Bézier curves by Frey et al. [4], elucidating both the necessary and sufficient conditions. For cubic or higher-degree curves, various methods have been proposed, most of which are related to a sufficient condition for curvature monotonicity. Walton et al. introduced Pythagorean hodograph quintic spirals [15]. Wang et al. proposed a fair curve design method based on the sufficient condition for curvature monotonicity [17]. Minueur et al. introduced typical curves [8], and Farin proposed class A Bézier curves [3]. Yoshida et al. proposed an interactive control method of class A Bézier curves and demonstrated that typical class A Bézier curves approaches logarithmic spirals under the same G^1 Hermite interpolation condition as the degree gets higher [19]. For matrix M that has two real eigenvalues, Romani et al. showed the necessary and sufficient conditions for the curves to be class A [9]. Yoshida et al. [20] approximated log-aesthetic curves in terms of rational cubic Bézier curves, while Lu et al. [7] employed quintic polynomials for the same purpose. 3D class A Bézier curves are investigated by Yoshida et al. [18], Tong et al. [14], and Wang et al. [16].

Unlike other approaches that investigate a sufficient condition for curvature monotonicity, Yoshida and Saito et al. proposed methods for real-time visualization of curvature monotonicity regions in 2D polynomial Bézier curves [21], and for 2D rational Bézier curves [10]. Our approach of visualizing a sufficient region is based on the visualization of regions bounded by implicit curves. Previous works for visualizing implicit curves focus mainly on the topological aspects of implicit curves, such as isolating singular points [1, 5, 6]. In this study, we employ a straightforward yet efficient method for visualizing regions bounded by implicit curves, leveraging the computational power of a GPU.

3 CURVATURE MONOTONICITY EVALUATION FUNCTION

This section reviews the curvature monotonicity evaluation functions for 2D polynomial and rational Bézier curves proposed by Saito et al. [10]. A 2D Bézier curve $\mathbf{P}(t)$ of degree n with $n{+}1$ control points $\mathbf{P}_j = [x_j\ y_j]^{\mathrm{T}}$ and weights w_j $(0 \le j \le n)$ is defined by

$$\mathbf{P}(t) = \frac{\mathbf{Q}(t)}{W(t)}, \tag{1}$$

where

$$\mathbf{Q}(t)=\sum_{j=0}^{n}B_j^n(t)w_j\mathbf{P}_j,\quad W(t)=\sum_{j=0}^{n}B_j^n(t)w_j, \tag{2}$$

and $B_j^n(t)$ is a Bernstein polynomial. We assume that the curve is regular. If all the weights w_j are 1, Eq. (1) represents a polynomial curve. Otherwise, Eq. (1) represents a rational curve.

The curvature monotonicity of a Bézier can be verified by checking whether $\frac{d\kappa}{ds}\leq 0$ or $\frac{d\kappa}{ds}\geq 0$ within $t\in[0\ 1]$. $\frac{d\kappa}{ds}$ for 2D curves is given by [2] as

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

where $\dot{\mathbf{P}}$, $\ddot{\mathbf{P}}$, or $\dddot{\mathbf{P}}$ represents the first, second or third derivative of $\mathbf{P}$ with respect to t. Since the denominator is always positive, curvature monotonicity can be verified by the numerator of $\frac{d\kappa}{ds}$, which we denote $\lambda(t)$. The numerator of $\frac{d\kappa}{ds}$ can be represented as a Bernstein polynomial of degree n_c:

$$\lambda(t)=\sum_{i=0}^{n_c}B_i^{n_c}(t)\xi_i. \tag{4}$$

For polynomial curves, $n_c=4n-7$, whereas for rational curves, $n_c=8n-12$ [10]. We refer to $\lambda(t)$ as the curvature monotonicity evaluation function. We consider ξ_i as implicit functions with respect to the control point coordinates for which we want to visualize the curvature monotonicity regions. The intersection of these functions will provide us with the sufficient regions.

We introduce internal division points and weights of the de Casteljau algorithm. For $0\leq k\leq m\leq n$, k-th internal division point $\boldsymbol{Q}_{m,k}(t)$ and weight $W_{m,k}(t)$ at $(n-m)$-th step are:

$$\boldsymbol{Q}_{m,k}(t)=\sum_{i=0}^{n-m}B_i^{n-m}w_{k+i}(t)\mathbf{P}_{k+i}, \tag{5}$$

$$W_{m,k}(t)=\sum_{i=0}^{n-m}B_i^{n-m}w_{k+i}(t). \tag{6}$$

If both m and k are single digit integers, the comma between them is omitted, as in $\boldsymbol{Q}_{01}$. Note that for polynomial curves, all weights w_i are 1 and Eq. (6) is not used.

$\lambda(t)$ of polynomial Bézier curves with degree n $(n\geq 3)$, which we denote $K_n(t)$, is

$$K_n(t)=S_4(\boldsymbol{V}_1\cdot\boldsymbol{V}_1)-3S_3(\boldsymbol{V}_1\cdot\boldsymbol{V}_2), \tag{7}$$

where

$$\begin{align}
\boldsymbol{V}_1 &= n\ (\boldsymbol{Q}_{11}-\boldsymbol{Q}_{10}), \nonumber\\
\boldsymbol{V}_2 &= n(n-1)\ (\boldsymbol{Q}_{22}-2\boldsymbol{Q}_{21}+\boldsymbol{Q}_{20}), \nonumber\\
S_3 &= n^2(n-1)\ (\boldsymbol{Q}_{20}\wedge\boldsymbol{Q}_{21}+\boldsymbol{Q}_{22}\wedge\boldsymbol{Q}_{20}+\boldsymbol{Q}_{21}\wedge\boldsymbol{Q}_{22}), \nonumber\\
S_4 &= n^2(n-1)(n-2) \tag{8}\\
&\qquad ((1-t)(\boldsymbol{Q}_{31}-\boldsymbol{Q}_{30})\wedge(2\boldsymbol{Q}_{31}-3\boldsymbol{Q}_{32}+\boldsymbol{Q}_{33}) \tag{9}\\
&\qquad +\quad t\quad (\boldsymbol{Q}_{30}-3\boldsymbol{Q}_{31}+2\boldsymbol{Q}_{32})\wedge(\boldsymbol{Q}_{33}-\boldsymbol{Q}_{32})) \nonumber
\end{align}$$

and the curvature monotonicity can be evaluated using the degree $4n-7$ function $K_n(t)$.

$\lambda(t)$ of rational Bézier curves with degree n $(n \geq 3)$, which we denote $k_n(t)$, is

$$\begin{align}
k_n(t) &= ((\boldsymbol{V}_5 \wedge \boldsymbol{V}_6)\boldsymbol{V}_5 + 3S_7\boldsymbol{V}_8) \cdot \boldsymbol{V}_5 \notag\\
&= (\boldsymbol{V}_5 \wedge \boldsymbol{V}_6)(\boldsymbol{V}_5 \cdot \boldsymbol{V}_5) + 3S_7(\boldsymbol{V}_8 \cdot \boldsymbol{V}_5), \tag{10}\\
\boldsymbol{V}_5 &= n\ (W_{10}\boldsymbol{Q}_{11} - W_{11}\boldsymbol{Q}_{10}), \notag\\
\boldsymbol{V}_6 &= n(n-1)(n-2) \tag{11}\\
&\quad (W_{30}\boldsymbol{Q}_{33} - 3W_{31}\boldsymbol{Q}_{32} + 3W_{32}\boldsymbol{Q}_{31} - W_{33}\boldsymbol{Q}_{30}) \tag{12}\\
S_7 &= n(n-1) \notag\\
&\quad\quad (W_{22}(\boldsymbol{Q}_{20} \wedge \boldsymbol{Q}_{21}) + W_{21}(\boldsymbol{Q}_{22} \wedge \boldsymbol{Q}_{20}) + W_{20}(\boldsymbol{Q}_{21} \wedge \boldsymbol{Q}_{22})), \notag\\
\boldsymbol{V}_8 &= n^2(n-1)\ ((1-t)(2W_{11}(W_{20}\boldsymbol{Q}_{21} - W_{21}\boldsymbol{Q}_{20}) \notag\\
&\qquad\qquad\qquad\qquad - W_{10}(W_{20}\boldsymbol{Q}_{22} - W_{22}\boldsymbol{Q}_{20})) \notag\\
&\qquad\qquad + \quad t \quad (W_{11}(W_{20}\boldsymbol{Q}_{22} - W_{22}\boldsymbol{Q}_{20}) \notag\\
&\qquad\qquad\qquad\qquad - 2W_{10}(W_{21}\boldsymbol{Q}_{22} - W_{22}\boldsymbol{Q}_{21})), \notag
\end{align}$$

and the curvature monotonicity can be evaluated using the degree $8n-12$ function $k_n(t)$. See [10] for the derivation of $K_n(t)$ and $k_n(t)$.

In [10], to visualize the curvature monotonicity regions, different fragment shader codes were used depending on the degree of Bézier curves. Using Eq. (7) or Eq. (10) in Bernstein form, fragment shader codes for different degree Bézier curves can be consolidated to a single fragment shader. In this approach, fragment shader code is stored as text within the application program. Degree-specific components are then dynamically replaced, and the fragment shader is subsequently recompiled. Using this approach, we only need one code for polynomial curves and one for rational curves, regardless of the degree, and we can visualize sufficient regions as long as the fragment shader fits into GPU memory.

4 REAL TIME VISUALIZATION OF THE REGIONS DEFINED BY IMPLICIT CURVES

To visualize the curvature monotonicity regions based on the sufficient condition in Sec. 5, we need to visualize the regions defined by implicit curves. This section introduces a real-time method for visualizing the regions defined by implicit curves using a GPU.

Our approach to visualize the regions defined by implicit curves using a GPU is similar to the idea of [21, 10]. In this method, computations are performed in the fragment shader for each pixel in a window, with the pixel being painted accordingly. Suppose that an implicit function is $f(x, y)$, and we need to visualize the region where $f(x, y) \leq 0$. If we need to visualize the region where $f(x, y) \geq 0$, $f(x, y)$ is negated. In the fragment shader of a GPU, $f(x, y)$ is computed in parallel for every pixel (x, y) in a window by drawing a rectangle of the window's size. If $f(x, y) \leq 0$ for a pixel, the pixel is colored with a user-specified color, or the color of the pixel is synthesized when drawing more than one implicit curves. Otherwise, the pixel color remains unchanged. In our implementation, $f(x, y)$s are computed for a 3×3 pixel grid to enable antialiasing. The color of the center pixel is computed as the weighted average of the colors of the surrounding 3×3 pixels.

Our approach can visualize regions defined by implicit curves much faster than Mathematica. Fig. 1 shows an example of visualizing the regions defined by six implicit curves using Mathematica. These curves consist of two cubic, two quadratic, and two linear curves. The colored regions indicate where at least one implicit function is positive. The white region near the origin indicates that all the implicit functions have negative values. The visualization presented in Fig. 1 was generated using Mathematica on a computer equipped with a Ryzen 9 5950X CPU with 64GB of memory and a GeForce RTX 4080 GPU with 16GB of memory. It took approximately 150 seconds to generate this visualization. The window size of Fig. 1 is 1300×1300 pixels. Fig.

1 corresponds to Fig. 3(a1). Computing Fig. 3(a1) with our approach on the same computer takes about 4 ms for a window of 1536×1536 pixels. In our implementation, we utilize C++ and OpenGL Shaders. The computation time depends on the number of implicit functions, their degrees, and the window size.

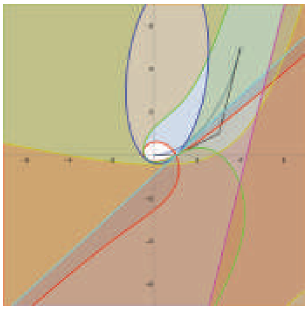


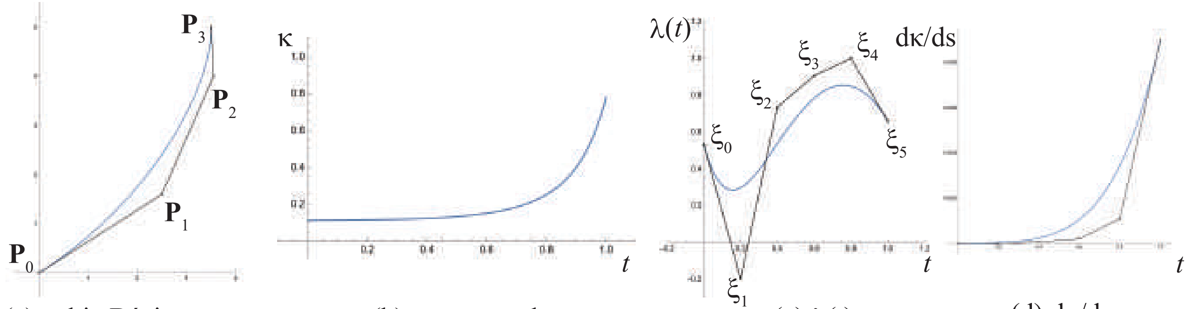
Figure 1: Visualization of the regions defined by implicit curves using Mathematica

5 CURVATURE MONOTONICITY REGION AS THE INTERSECTION OF IMPLICIT REGIONS

In this section, we visualize the sufficient regions as the intersection of implicit curves. We visualize sufficient regions of cubic polynomial, quartic polynomial, and rational cubic Bézier curves.

For a Bézier curve, if $\lambda(t) \geq 0$ or $\lambda(t) \leq 0$ within $t \in [0\ 1]$, the curvature is monotonically varying. As shown in Eq. (4), $\lambda(t)$ is an explicit Bézier curve [13] with control points ξ_i. We refer to the curvature monotonicity regions computed using the condition $\lambda(t) \geq 0$ or $\lambda(t) \leq 0$ within the interval $t \in [0, 1]$ as the *exact curvature monotonicity regions*, or simply as the *exact regions*. The curvature of a curve can be monotonically varying even if ξ_is have different signs. Fig. 2 shows a cubic Bézier curve, its curvature plot, $\lambda(t)$, and $\frac{d\kappa}{ds}$, where the signs of ξ_i are different, but $\lambda(t) \geq 0$ within $t \in [0\ 1]$ as shown in Fig. 2(c). Note that ξ_is are scaled so that $|\xi_i| \leq 1$. $\frac{d\kappa}{ds}$ is $\lambda(t)$ divided by $|\dot{\mathbf{P}}|^6$, which is shown in Fig. 2(d). Although the curve in Fig. 2(c) and the curve in Fig. 2(d) are quite different, both curves are positive within $t \in [0\ 1]$.

When the curvature is monotonically varying, ξ_is may have different signs. To simplify the situation, we investigate the curvature monotonicity regions based on a sufficient condition. Our sufficient region is defined by $\xi_i \leq 0$ for curves with monotonically decreasing curvature or $\xi_i \geq 0$ for curves with monotonically increasing curvature, where $0 \leq i \leq 4n-7$ for polynomial curves or $0 \leq i \leq 8n-12$ for rational curves. Concerning the sufficient region of a control point $\mathbf{P}_j$ $(0 \leq j \leq n)$, it is the intersection of all $\xi_i \geq 0$ (for curves



(a) cubic Bézier curve (b) curvaure plot (c) λ(t) (d) dκ/ds

Figure 2: Cubic Bézier curve, curvature plot, and $\lambda(t)$

with monotonically increasing curvature) or $\xi_i \leq 0$ (for curves with monotonically decreasing curvature) with $\mathbf{P}_j = [x_j \ y_j]^{\mathrm{T}}$ representing a variable associated with ξ_i. Note that a sufficient region may be disconnected.

For a control point $\mathbf{P}_j$ for which we want to visualize the sufficient region, and for each ξ_i, we will find the region in the (x, y) plane where $\mathbf{P}_j$ can be replaced, ensuring that $\xi_i \leq 0$ (or $\xi_i \geq 0$) still holds. To visualize the region of $\xi_i \leq 0$ (or $\xi_i \geq 0$) for $\mathbf{P}_j$, we compute the value of ξ_i in the fragment shader by replacing the coordinate of $\mathbf{P}_j$ with the coordinates corresponding to each pixel in a window. When we visualize the region for decreasing curvature, the corresponding pixel is painted with a user-specified color if $\xi_i < 0$. Otherwise, the pixel remains white. When we visualize the region for increasing curvature, the corresponding pixel is painted if $\xi_i > 0$. Repeating the computation of ξ_i by $4n - 7$ times (or $8n - 12$ times for rational curves), and appropriately synthesizing the colors, allows us to visualize all regions defined by ξ_is simultaneously.

To compute ξ_i, Eq. (7) is used for polynomial curves, whereas Eq. (10) is used for rational curves. For Bernstein multiplications, employing scaled Bernstein [11] significantly simplifies the process, as binomial coefficients are only required in the first and last steps of multiplication. In our implementation, our visualization program works for polynomial curves up to degree 13 and rational curves up to degree 9 using a GeForce RTX 4080 GPU with 16GB of memory. Visualizing sufficient regions enables us to offer a partial explanation for the exact curvature monotonicity regions, particularly when the sizes of the exact regions and the sufficient regions are similar.

5.1 Sufficient Regions of a Cubic Bézier Curve

We visualize the sufficient regions of a cubic polynomial Bézier curve with $\mathbf{P}_0 = [0 \ 0]^{\mathrm{T}}$, $\mathbf{P}_1 = [1 \ 0]^{\mathrm{T}}$, $\mathbf{P}_2 = [3 \ 1]^{\mathrm{T}}$, and $\mathbf{P}_3 = [4 \ 5]^{\mathrm{T}}$. Fig. 3 illustrates the curvature monotonicity region for each control point, along with the control polygon and the curve. In sufficient regions, the regions with $\xi_i \geq 0$ are colored, while the regions $\xi_i < 0$ remain white. Therefore, the sufficient regions are colored white. Note that for visualizing sufficient regions with monotonically increasing curvature, regions with $\xi_i \leq 0$ are colored. For each control point $\mathbf{P}_j$, $\xi_i = 0$ is an implicit curve with respect to x_j and y_j. In Fig. 3, the exact regions are computed using the method proposed in [21, 10].

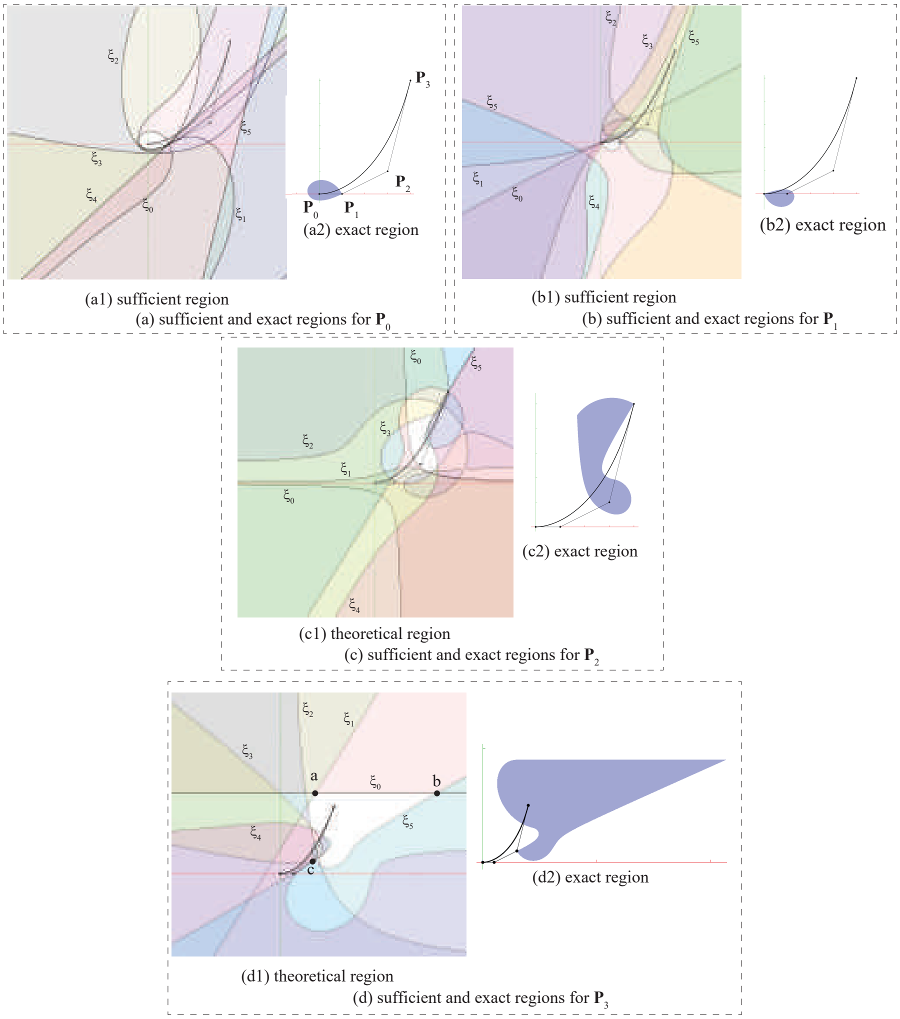
In the sufficient region of $\mathbf{P}_0$ in Fig. 3(a1), $\xi_0 = 0$ and $\xi_1 = 0$ are implicit cubic curves. $\xi_2 = 0$ and $\xi_3 = 0$ are implicit quadratic curves. In this case, $\xi_2 = 0$ forms an ellipse and $\xi_3 = 0$ is a hyperbola. $\xi_4 = 0$ and $\xi_5 = 0$ are both lines. $\xi_0 = 0$, $\xi_1 = 0$, and $\xi_2 = 0$ go through $\mathbf{P}_1$. We verified that these three curves pass through $\mathbf{P}_1$ in the general case using Mathematica. Note that $\xi_4 = 0$ intersects with $\mathbf{P}_1$ in this specific case, but not necessarily in a general context. Upon careful examination of the theoretical region, it becomes evident that the boundary is defined by $\xi_0 = 0$ and $\xi_1 = 0$ in this context.

In the sufficient region of $\mathbf{P}_1$ in Fig. 3(b1), $\xi_i = 0$ $(0 \leq i \leq 4)$ are implicit cubic curves. $\xi_5 = 0$ is an implicit quadratic curve, which takes the form of a hyperbola in this context. $\xi_0 = 0$, $\xi_1 = 0$ and $\xi_2 = 0$ go through $\mathbf{P}_0$. The boundary of the sufficient region is defined by $\xi_0 = 0$ and $\xi_4 = 0$ in this context.

In the sufficient region of $\mathbf{P}_2$ in Fig. 3(c1), $\xi_0 = 0$ is an implicit quadratic curve, which takes the form of a hyperbola in this context. ξ_i $(1 \leq i \leq 5)$ are implicit cubic curves. $\xi_3 = 0$, $\xi_4 = 0$ and $\xi_5 = 0$ go through $\mathbf{P}_3$. The boundary of the sufficient region is defined by $\xi_0 = 0$, $\xi_2 = 0$,$\xi_3 = 0$, $\xi_4 = 0$ and $\xi_5 = 0$ in this context.

In the sufficient region of $\mathbf{P}_3$ in Fig. 3(d1), $\xi_0 = 0$ and $\xi_1 = 0$ are lines. $\xi_2 = 0$ and $\xi_3 = 0$ are quadratic implicit curves, taking the form of hyperbolas in this case. $\xi_4 = 0$ and $\xi_5 = 0$ are implicit cubic curves. $\xi_3 = 0$, $\xi_4 = 0$ and $\xi_5 = 0$ go through P_2. The boundary of the sufficient region is defined by all ξ_is in this context.

If $\xi_0 = 0$ or $\xi_5 = 0$ serves as a boundary for the sufficient region, ξ_0 or ξ_5 also forms the boundary of the exact region. This is because $\lambda(t)$ is in the Bernstein basis. Since $\lambda(t)$ is in the Bernstein basis, it always goes through the first and last control points. In Fig. 3(a1), $\xi_0 = 0$ occupies most of the boundary of the exact region. In Fig. 3(d1), $\xi_0 = 0$ from point a to point b, and $\xi_5 = 0$ from point b to point c, coincide with the boundary of the exact region.



(a1) sufficient region
(a2) exact region
(a) sufficient and exact regions for $\mathbf{P}_0$

(b1) sufficient region
(b2) exact region
(b) sufficient and exact regions for $\mathbf{P}_1$

(c1) theoretical region
(c2) exact region
(c) sufficient and exact regions for $\mathbf{P}_2$

(d1) theoretical region
(d2) exact region
(d) sufficient and exact regions for $\mathbf{P}_3$

Figure 3: Regions of a cubic Bézier curve for each control point

5.2 Sufficient Regions of a Quartic Bézier Curve

We visualize sufficient regions of a quartic polynomial Bézier curve with $\mathbf{P}_0 = [0\ 0]^{\mathrm{T}}$, $\mathbf{P}_1 = [0.2\ 0]^{\mathrm{T}}$, $\mathbf{P}_2 = [0.6\ 0.2]^{\mathrm{T}}$, $\mathbf{P}_3 = [0.8\ 0.6]^{\mathrm{T}}$, and $\mathbf{P}_4 = [1.2\ 1.4]^{\mathrm{T}}$.

Fig. 4 shows the sufficient and exact regions for each control point.

In Fig. 4(a1), $\xi_i = 0 (0 \leq i \leq 2)$ are implicit cubic curves. $\xi_i = 0 (3 \leq i \leq 5)$ are implicit quadratic curves. $\xi_3 = 0$ is an ellipse, and $\xi_4 = 0$ and $\xi_5 = 0$ are hyperbolas in this context. $\xi_i = 0 (6 \leq i \leq 8)$ are lines. ξ_9 is a constant, which takes a negative value in this context. $\xi_0 = 0$, $\xi_1 = 0$, $\xi_2 = 0$ go through $\mathbf{P}_1$.

In Fig. 4(b1), $\xi_i = 0 (0 \leq i \leq 5)$ are implicit cubic curves. $\xi_6 = 0$ and $\xi_7 = 0$ are implicit quadratic curves, which are hyperbolas in this context. $\xi_8 = 0$ and $\xi_9 = 0$ are lines. $\xi_0 = 0$, $\xi_1 = 0$, $\xi_2 = 0$ go through $\mathbf{P}_0$.

In Fig. 4(c1), $\xi_0 = 0$ and $\xi_9 = 0$ are implicit quadratic curves, which are hyperbolas in this context. $\xi_i = 0 (0 \leq i \leq 8)$ are implicit cubic curves.

In Fig. 4(d1), $\xi_0 = 0$ and $\xi_1 = 0$ are lines. $\xi_2 = 0$ and $\xi_3 = 0$ are implicit quadratic curves, which are hyperbolas in this context. $\xi_i = 0 (4 \leq i \leq 9)$ are implicit cubic curves. $\xi_7 = 0$, $\xi_8 = 0$, $\xi_9 = 0$ go through $\mathbf{P}_4$.

In Fig. 4(e1), ξ_0 is a constant, which takes a negative value in this context. $\xi_i = 0 (1 \leq i \leq 3)$ are lines. $\xi_i = 0 (4 \leq i \leq 6)$ are implicit quadratic curves, which are hyperbolas in this context. $\xi_i = 0 (7 \leq i \leq 9)$ are implicit cubic curves. $\xi_7 = 0$, $\xi_8 = 0$, and $\xi_9 = 0$ intersect at $\mathbf{P}_3$.

If $\xi_0 = 0$ or $\xi_9 = 0$ serves as the boundary of a sufficient region, it also forms the boundary of the exact region.

5.3 Sufficient Regions of a Rational Cubic Bézier Curve

Fig. 5 shows the sufficient and exact regions of a rational cubic Bézier curve with $\mathbf{P}_0 = [0\ 0]^{\mathrm{T}}$, $\mathbf{P}_1 = [1\ 0]^{\mathrm{T}}$, $\mathbf{P}_2 = [3\ 1]^{\mathrm{T}}$, $\mathbf{P}_3 = [4\ 5]^{\mathrm{T}}$, $w_0 = 1$, $w_1 = 0.9$, $w_2 = 1.1$, and $w_3 = 1$. Compared to the curve in Fig. 3, only the weights are different.

In Fig. 5(a1), $\xi_i = 0 (0 \leq i \leq 8)$ are implicit cubic curves. $\xi_9 = 0$ and $\xi_{10} = 0$ are implicit quadratic curves, which are hyperbolas in this context. $\xi_{11} = 0$ and $\xi_{12} = 0$ are lines. $\xi_0 = 0$, $\xi_1 = 0$, and $\xi_2 = 0$ go through $\mathbf{P}_1$.

In Fig. 5(b1), $\xi_i = 0 (0 \leq i \leq 11)$ are implicit cubic curves. $\xi_{12} = 0$ is an implicit quadratic curve, which is a hyperbola in this context. $\xi_0 = 0$, $\xi_1 = 0$, and $\xi_2 = 0$ go through $\mathbf{P}_0$.

In Fig. 5(c1), $\xi_0 = 0$ is an implicit quadratic curve, which is a hyperbola in this context. $\xi_i = 0 (1 \leq i \leq 12)$ are implicit cubic curves. $\xi_{10} = 0$, $\xi_{11} = 0$, and $\xi_{12} = 0$ go through $\mathbf{P}_3$.

In Fig. 5(d1), $\xi_0 = 0$ and $\xi_1 = 0$ are lines. $\xi_2 = 0$ and $\xi_3 = 0$ are implicit quadratic curves, which are hyperbolas in this context. $\xi_i = 0 (4 \leq i \leq 12)$ are implicit cubic curves. $\xi_{10} = 0$, $\xi_{11} = 0$, and $\xi_{12} = 0$ go through $\mathbf{P}_2$.

If $\xi_0 = 0$ or $\xi_{12} = 0$ serves as the boundary of a sufficient region, it also forms the boundary of the exact region.

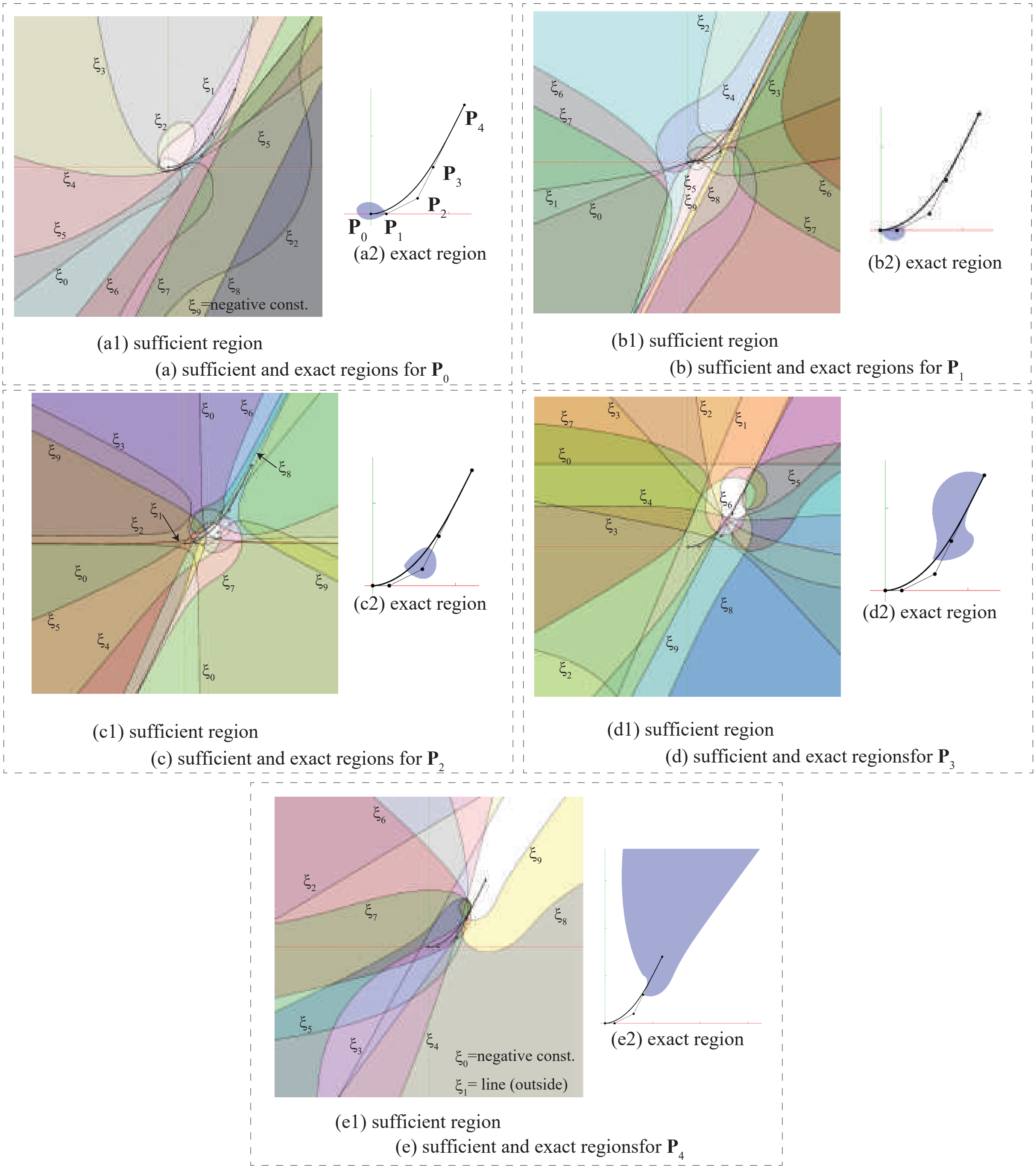
5.4 Degrees of ξ_i

Table 1 shows the degrees of ξ_is of polynomial and rational Bézier curves up to degree five. For a control point $\mathbf{P}_j = [x_j\ y_j]^{\mathrm{T}}$, the degree is related to both x_j and y_j. For example, for a polynomial cubic Bézier curve, concerning $\mathbf{P}_0$, ξ_0 and ξ_1 are implicit cubics with respect to both x_0 and y_0. The degrees of ξ_is are verified using Mathematica. In the Mathematica code for computing the degree, the degrees of x_i and y_i are checked for each term of ξ_i, and the degree of x_i and the degree of y_i are added. The maximum degree among each term of ξ_i is selected as the degree with respect to x_i and y_i.

From Table 1 and considering the computation of Eq. (7) and Eq. (10), we can hypothesize that the maximum degree of ξ_is is 3 for both polynomial and rational Bézier curves with any degree $(n \geq 3)$. If this hypothesis holds, the sufficient region is the intersection of half-spaces defined by implicit curves up to degree 3. We verified using Mathamatica that this hypothesis is true for polynomial and rational Bézier curves up to degree 10.

5.5 Limitations

In many cases, the shape of a sufficient region closely resembles that of the exact region. However, instances arise where the sufficient region is significantly smaller than the exact region or does not exist at all. Fig. 6 presents such a case involving a cubic Bézier curve with $\mathbf{P}_0 = [0.1\ \ -0.4]^{\mathrm{T}}$, $\mathbf{P}_1 = [0.2\ 0]^{\mathrm{T}}$, $\mathbf{P}_2 = [-0.5\ 0.6]^{\mathrm{T}}$, and $\mathbf{P}_3 = [-1.4\ \ -0.3]^{\mathrm{T}}$. Fig. 6(a) illustrates the exact region of $\mathbf{P}_2$. Fig. 6(b) shows $\xi_i (0 \leq i \leq 5)$ for $\mathbf{P}_2$. The absence of a white region indicates that the sufficient region does not exist. This phenomenon of vanishing sufficient regions similarly occurs for higher-degree curves. Consequently, our analysis, which relies on investigating the sufficient region, is ineffective in this case.



(a1) sufficient region

(a2) exact region

(a) sufficient and exact regions for $\mathbf{P}_0$

(b1) sufficient region

(b2) exact region

(b) sufficient and exact regions for $\mathbf{P}_1$

(c1) sufficient region

(c2) exact region

(c) sufficient and exact regions for $\mathbf{P}_2$

(d1) sufficient region

(d2) exact region

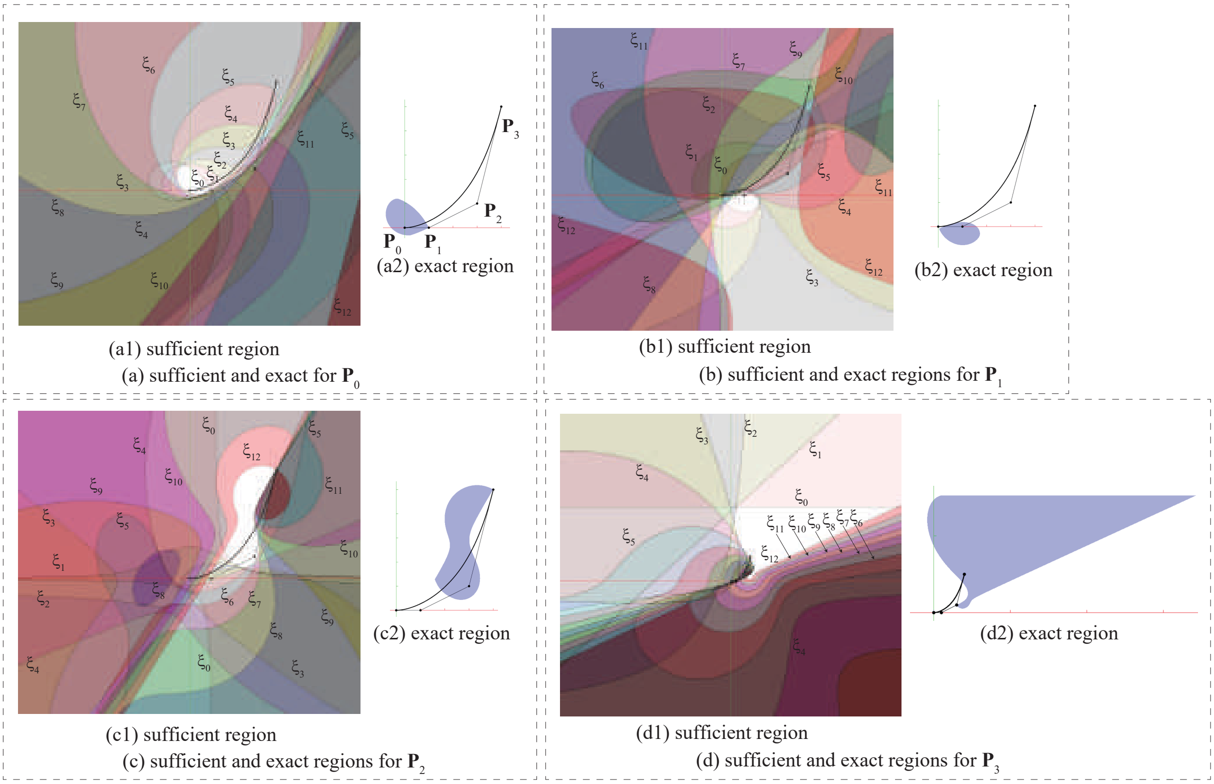
(d) sufficient and exact regionsfor $\mathbf{P}_3$

(e1) sufficient region

(e2) exact region

(e) sufficient and exact regionsfor $\mathbf{P}_4$

Figure 4: Regions of a quartic Bézier curve for each control point



(a1) sufficient region

(a2) exact region

(a) sufficient and exact for $\mathbf{P}_0$

(b1) sufficient region

(b2) exact region

(b) sufficient and exact regions for $\mathbf{P}_1$

(c1) sufficient region

(c2) exact region

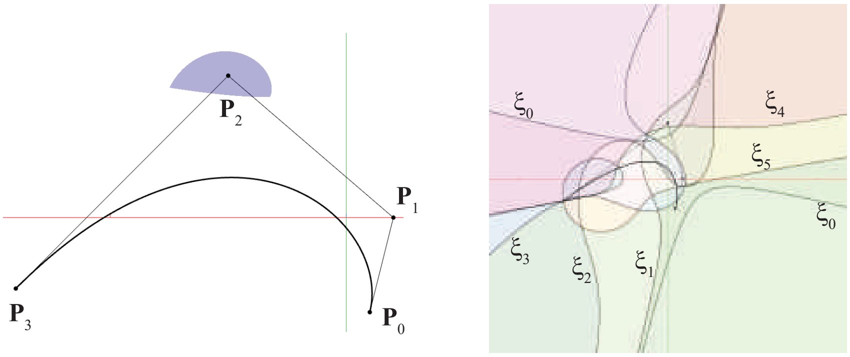
(c) sufficient and exact regions for $\mathbf{P}_2$

(d1) sufficient region

(d2) exact region

(d) sufficient and exact regions for $\mathbf{P}_3$

Figure 5: Regions of a rational cubic Bézier curve for each control point



(a) exact region for $\mathbf{P}_2$

(b) ξ_is

Figure 6: Cubic Bézier curve with no sufficient region of $\mathbf{P}_2$

Table 1: Degrees of ξ_i

	$\mathbf{P}_0$				$\mathbf{P}_1$			
degree	3	2	1	0	3	2	1	0
cubic polynomial	ξ_0, ξ_1	ξ_2,ξ_3	ξ_4, ξ_5	-	ξ_0 to ξ_4	ξ_5	-	-
quartic polynomial	ξ_0 to ξ_2	ξ_3 to ξ_5	ξ_6 to ξ_8	ξ_9	ξ_0 to ξ_5	ξ_6, ξ_7	ξ_8,ξ_9	
quintic polynomial	ξ_0 to ξ_3	ξ_4 to ξ_7	ξ_8 to ξ_{11}	ξ_{12},ξ_{13}	ξ_0 to ξ_6	ξ_7 to ξ_9	ξ_{10} to ξ_{12}	ξ_{13}
rational cubic	ξ_0 to ξ_8	ξ_9,ξ_{10}	ξ_{11}, ξ_{12}	-	ξ_0 to ξ_{11}	ξ_{12}	-	-
rational quartic	ξ_0 to ξ_{13}	ξ_{14} to ξ_{16}	ξ_{17} to ξ_{19}	ξ_{20}	ξ_0 to ξ_{16}	ξ_{17}, ξ_{18}	ξ_{19},ξ_{20}	
rational quintic	ξ_0 to ξ_{18}	ξ_{19} to ξ_{22}	ξ_{23} to ξ_{26}	ξ_{27},ξ_{28}	ξ_0 to ξ_{21}	ξ_{22} to ξ_{24}	ξ_{25} to ξ_{27}	ξ_{28}
	$\mathbf{P}_2$				$\mathbf{P}_3$			
degree	3	2	1	0	3	2	1	0
cubic polynomial	ξ_1 to ξ_5	ξ_0	-	-	ξ_4,ξ_5	ξ_2, ξ_3	ξ_0,ξ_1	-
quartic polynomial	ξ_1 to ξ_8	ξ_0, ξ_9	-	-	ξ_4 to ξ_9	ξ_2, ξ_3	ξ_0,ξ_1	
quintic polynomial	ξ_1 to ξ_9	ξ_0, ξ_{10}, ξ_{11}	ξ_{12},ξ_{13}	-	ξ_4 to ξ_{12}	ξ_2,ξ_3,ξ_{13}	ξ_0, ξ_1	-
rational cubic	ξ_1 to ξ_{12}	ξ_0	-	-	ξ_4 to ξ_{12}	ξ_2,ξ_3	ξ_0,ξ_1	-
rational quartic	ξ_1 to ξ_{19}	ξ_0, ξ_{20}	-	-	ξ_4 to ξ_{20}	ξ_2, ξ_3	ξ_0,ξ_1	
rational quintic	ξ_1 to ξ_{24}	ξ_0, ξ_{25}, ξ_{26}	ξ_{27}, ξ_{28}	-	ξ_4 to ξ_{27}	ξ_2, ξ_3, ξ_{28}	ξ_0,ξ_1	-
	$\mathbf{P}_4$				$\mathbf{P}_5$			
degree	3	2	1	0	3	2	1	0
quartic polynomial	ξ_7 to ξ_9	ξ_4 to ξ_6	ξ_1 to ξ_3	ξ_0	-	-	-	-
quintic polynomial	ξ_7 to ξ_{13}	ξ_4 to ξ_6	ξ_1 to ξ_3	ξ_0	ξ_{10} to ξ_{13}	ξ_6 to ξ_9	ξ_2 to ξ_5	ξ_0,ξ_1
rational quartic	ξ_7 to ξ_{20}	ξ_4 to ξ_6	ξ_1 to ξ_3	ξ_0	-	-	-	-
rational quintic	ξ_7 to ξ_{28}	ξ_4 to ξ_6	ξ_1 to ξ_3	ξ_0	ξ_{10} to ξ_{28}	ξ_6 to ξ_9	ξ_2 to ξ_5	ξ_0,ξ_1

6 CONCLUSIONS

We propose a method to visualize the curvature monotonicity regions for 2D polynoimal and rational Bézier curves based on a sufficient condition. The visualization of these sufficient regions uses a real-time method for visualizing the half-space of implicit curves, facilitated by GPU technology. Using a GPU enables interactive adjustments of a control point, allowing for the real-time display of the sufficient region alongside all implicit curves ($\xi_i = 0$). We established that the sufficient region constitutes the intersection of half-spaces defined by implicit curves up to degree 3, a finding verified for polynomial and rational Bézier curves of degrees up to 10. Although our demonstrations focused primarily on sufficient regions for cubic and quartic polynomial Bézier curves, as well as for a rational cubic Bézier curve, our program is capable of accommodating higher-degree curves, contingent upon the compatibility of the fragment shader code with the GPU's capacity.

Our visualization of a sufficient region is based on Eq. (3). Thus, our sufficient region may include the region where an inflection point is generated. To avoid this, the 3D equation of $\frac{d\kappa}{ds}$, Eq. (7) in [10], should be used. The degree of $\lambda(t)$ increases to $6n - 11$ for polynomial curves or $11n - 8$ for rational curves, resulting in an increase in the number of implicit curves.

An exact curvature monotonicity region encompasses its sufficient region. Thus, our examination of the sufficient region partially elucidates the construction of curvature monotonicity regions. However, a more rigorous analysis of the exact curvature monotonicity region is deferred to future work. We are currently extending the concept proposed in this paper to 3D curves.

ACKNOWLEDGEMENTS

The authors express their gratitude to the reviewers for their very helpful comments. This work was supported by JSPS KAKENHI Grant Number 24K07280.

Norimasa Yoshida, https://orcid.org/0000-0001-8889-0949
Takafumi Saito, https://orcid.org/0000-0001-5831-596X

REFERENCES

[1] Alberti, L.; Mourrain, B.: Visualisation of implicit algebraic curves. In 15th Pacific Conference on Computer Graphics and Applications (PG'07), 303–312, 2007. http://doi.org/10.1109/PG.2007.32.

[2] Farin, G.: Curves and Surfaces for CAGD, 5th edition. Morgan Kaufmann, 2001.

[3] Farin, G.: Class A Bézier curves. Computer Aided Geometric Design, 23(7), 573–581, 2006. http://doi.org/10.1016/j.cagd.2006.03.004.

[4] Frey, W.H.; Field, D.A.: Designing Bézier conic segments with monotone curvature. Computer Aided Geometric Design, 17(6), 457–483, 2000. http://doi.org/10.1016/S0167-8396(00)00011-X.

[5] Gomes, A.J.: A continuation algorithm for planar implicit curves with singularities. Computers & Graphics, 38, 365–373, 2014. http://doi.org/10.1016/j.cag.2013.11.006.

[6] Jinsan Cheng, S.L.; Peñaranda, L.; Pouget, M.; Rouillier, F.; Tsigaridas, E.: On the topology of planar algebraic curves. In SCG '09: Proceedings of the twenty-fifth annual symposium on Computational geometry, 361–370, 2009. http://doi.org/10.1145/1542362.1542424.

[7] Lu, L.; Xiang, X.: Quintic polynomial approximation of log-aesthetic curves by curvature deviation. Journal of Computational and Applied Mathematics, 296, 389–396, 2016. http://doi.org/10.1016/j.cam.2015.10.002.

[8] Mineur, Y.; Lichah, T.; Castelain, J.M.; Giaume, H.: A shape controlled fitting method for Bézier curves. Computer Aided Geometric Design, 15(9), 879–891, 1998. http://doi.org/10.1016/S0167-8396(98)00025-9.

[9] Romani, L.; Viscardi, A.: Planar class A Bézier curves: The case of real eigenvalues. Computer Aided Geometric Design, 89, 2021. http://doi.org/10.1016/j.cagd.2021.102021.

[10] Saito, T.; Yoshida, N.: Curvature monotonicity evaluation functions on rational Bézier curves. Computers & Graphics, 114, 219–229, 2023. http://doi.org/10.1016/j.cag.2023.05.019.

[11] Sánchez-Reys, J.: Algebraic manipulation in the bernstein form made simple via convolutions. Computer Aided Design, 35(10), 959–967, 2003. http://doi.org/10.1016/S0010-4485(03)00021-6.

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

[13] Sederberg, T.W.: Computer Aided Geometric Design. BYU Faculty Publication 1, 2012. https://scholarsarchive.byu.edu/facpub/1.

[14] Tong, W.; Chen, M.: A sufficient condition for 3d typical curves. Computer Aided Geometric Design, 87, 2021. http://doi.org/10.1016/j.cagd.2021.101991.

[15] Walton, D.J.; Meek, D.S.: A pythagorean hodograph quintic spiral. Computer Aided Design, 28(12), 943–950, 1996. http://doi.org/10.1016/0010-4485(96)00030-9.

[16] Wang, A.; He, C.; Zheng, J.; Zhao, G.: 3D class A Bézier curves with monotone curvature. Computer Aided Design, 159, 2023. http://doi.org/10.1016/j.cad.2023.103501.

[17] 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), 515–527, 2004. http://doi.org/10.1016/j.cagd.2004.04.001.

[18] Yoshida, N.; Fukuda, R.; Saito, T.: Interactive generation of 3D class A Bézier curve segments. Computer-Aided Design and Applications, 7(2), 163–172, 2010. http://doi.org/10.3722/cadaps.2010.163-172.

[19] Yoshida, N.; Hiraiwa, T.; Saito, T.: Interactive control of planar class A Bézier curves using logarithmic curvature graphs. Computer-Aided Design & Applications, 5(1–4), 121–130, 2008. http://doi.org/10.3722/cadaps.2008.121-130.

[20] Yoshida, N.; Saito, T.: Quasi-aesthetic curves in rational cubic Bézier forms. Computer-Aided Design and Applications, 4(1-4), 477–486, 2007. http://doi.org/10.1080/16864360.2007.10738567.

[21] Yoshida, N.; Sakurai, S.; Yasuda, H.; Inoue, T.; Saito, T.: Visualization of the curvature monotonicity regions of polynomial curves and its application to curve design. Computer-Aided Design and Applications, 21(12), 75–87, 2024. http://doi.org/10.14733/cadaps.2024.75-87.