

基于迭代算法的Bézier曲线求交方法研究

毛雯慧¹, 慈晏平², 李宏娜²

¹辽宁师范大学数学学院, 辽宁 大连

²辽宁师范大学计算机与人工智能学院, 辽宁 大连

收稿日期: 2025年5月25日; 录用日期: 2025年6月17日; 发布日期: 2025年6月27日

摘要

在计算机图形学及相关领域中, Bézier曲线求交问题备受关注。本文着重探究基于迭代算法的Bézier曲线求交方法。研究发现, 当Bézier曲线的控制多边形不相交时, 曲线本身必定不相交; 而当控制多边形相交时, 采用De Casteljau作为核心迭代手段, 通过不断对曲线区间进行二分处理, 逐步逼近曲线交点, 有效提升求交的准确性与效率。此方法为解决Bézier曲线求交难题提供了新的思路与途径, 在图形设计、计算机辅助几何设计等方面具有潜在应用价值。

关键词

迭代算法, Bézier曲线, 曲线求交, De Casteljau算法, 控制多边形

Intersection Computation of Bézier Curves Based on Iterative Methods

Wenhui Mao¹, Yanping Ci², Hongna Li²

¹School of Mathematics, Liaoning Normal University, Dalian Liaoning

²School of Computer Science and Artificial Intelligence, Liaoning Normal University, Dalian Liaoning

Received: May 25th, 2025; accepted: Jun. 17th, 2025; published: Jun. 27th, 2025

Abstract

In the field of computer graphics and related areas, the problem of intersection of Bézier curves has received much attention. This paper focuses on exploring the method of finding the intersection of Bézier curves based on an iterative algorithm. The study reveals that when the control polygons of Bézier curves do not intersect, the curves themselves must not intersect; while when the control polygons intersect, the De Casteljau algorithm is adopted as the core iterative method. By continuously

performing bisection processing on the curve intervals, the intersection points of the curves are gradually approached, effectively improving the accuracy and efficiency of intersection calculation. This method provides new ideas and approaches for solving the difficult problem of finding the intersection of Bézier curves, and has potential application values in aspects such as graphic design and computer-aided geometric design.

Keywords

Iterative Algorithm, Bézier Curve, Curve Intersection, De Casteljau Algorithm, Control Polygon

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

在计算机图形学(Computer Graphics, CG)与计算机辅助几何设计(Computer Aided Geometric Design, CAGD)领域, Bézier 曲线作为一种灵活且强大的几何建模工具,被广泛应用于各种复杂形状的构建与表示[1]。在实际应用中,判断多条 Bézier 曲线是否相交以及确定交点位置是一个核心且基础的问题。

现有的 Bézier 曲线求交算法虽然丰富多样,但各有局限性。部分算法计算过程复杂,需要消耗大量的计算资源和时间,在处理大规模或复杂曲线模型时效率低下。还有一些算法在精度控制上存在不足,容易受到数值误差的影响,导致求交结果不准确。

本文主要对基于迭代算法 Bézier 曲线求交方法进行研究。此方法计算逻辑简洁高效,能有效减少计算量,显著提升计算效率,在复杂图形处理、精密模具设计等实际场景中具有广阔的应用前景。

2. 一元 Bernstein 基函数定义与性质

n 次多项式 Bernstein 基函数为

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}, i = 0, 1, \dots, n,$$

其中

$$\binom{n}{i} = \frac{n!}{i!(n-i)!}, i = 0, 1, \dots, n.$$

为了方便,当 $i < 0$ 或者 $i > n$ 时,通常我们记 $B_i^n = 0$ [2]。Bernstein 基函数决定了每个控制顶点对曲线上点的控制权重,随参数 t 在 $[0, 1]$ 内变化,各基函数值相应改变,进而影响曲线形状。Bernstein 基函数有如下性质:

(1) 非负性与权性: 在计算曲线上某点时,所有控制顶点对该点的影响总和为 1,确保曲线在控制顶点构成的凸包内,是 Bézier 曲线凸包性的数学依据。

(2) 端点特性: 在端点 $t = 0$ 和 $t = 1$ 处,分别只有一个 Bernstein 基函数取值为 1,其余全部为 0 即

$$B_i^n(0) = \begin{cases} 1, & i = 0, \\ 0, & i \neq 0, \end{cases} \quad B_i^n(1) = \begin{cases} 1, & i = n, \\ 0, & i \neq n. \end{cases}$$

3. n 次 Bézier 曲线的原理及其性质

称参数曲线段

$$P(t) = \sum_{i=0}^n P_i B_i^n(t), t \in [0, 1] \quad (1)$$

为一条 n 次 Bézier, 其中 $B_i^n(t) (i=0, 1, \dots, n)$ 为 n 次 Bernstein 基函数, 空间向量 $P_i \in R^3 (i=0, 1, \dots, n)$ 为控制顶点, 依次用直线连接相邻控制顶点, 所得的 n 边折线多边形称之为 Bézier 多边形[2]。

利用 Bernstein 基函数的性质不难推出由(1)所定义的 Bézier 曲线有如下基本性质[2]:

(1) 端点插值性质: Bézier 曲线插值于首末两个控制顶点, 即 $P(0) = P_0, P(1) = P_n$ 。

(2) 凸包性质: 在几何意义上 Bézier 曲线完全落在由其控制多边形确定的凸包中。

(3) 对称性质: 若控制顶点顺序反转, 得到的 Bézier 曲线形状不变, 但方向相反。这说明了由同一控制多边形定义的 Bézier 曲线是唯一的。

(4) 几何不变性与仿射不变性: Bézier 曲线的形状取决于控制顶点, 而与坐标系的选取无关, 它是几何不变的。对 Bézier 曲线进行平移、旋转、缩放等仿射变换, 等同于对其控制顶点进行相同的变换。

(5) 参数连续性: 在参数区间 $[0, 1]$ 上, Bézier 曲线具有 $n-1$ 阶连续导数。

4. De Casteljau 算法的迭代求交算法

在计算机辅助几何设计(CAGD)领域, De Casteljau 算法是一种用于生成和计算 Bézier 曲线(Bézier Curves)的经典递归算法[3]。该算法基于凸组合原理, 通过递归地对控制多边形的顶点进行线性插值, 可高效地计算 Bézier 曲线上任意参数点的坐标。其核心思想是将 n 次 Bézier 曲线分解为两个 $n-1$ 次 Bézier 曲线的组合, 通过迭代降阶的方式逐步逼近目标点, 确保计算过程的数值稳定性与几何直观性。其迭代过程如下:

(1) 对于给定的 n 次 Bézier 曲线, 其控制点为 $P_0, P_1, \dots, P_n$, 要计算曲线上参数为 $t \in [0, 1]$ 的点 $B(t)$ 。该算法通过不断地对控制点进行线性插值来逐步逼近曲线上的点[2]。

(2) 计算过程[2]: 一条 n 次 Bézier 曲线从形式上“降阶”为 $(n-1)$ 次 Bézier 曲线对任意的 $0 \leq i \leq n-1$, 新的控制顶点 $P_i^{(1)}$ 落在原控制多边形的边 $P_i P_{i+1}$ 上, 且将边按比例 $t: (1-t)$ 进行分割。一直这样“降阶”下去,

$$P(t) = \sum_{i=0}^{n-1} P_i^{(1)}(t) B_i^{n-1}(t) = \dots = \sum_{i=0}^{n-k} P_i^{(k)}(t) B_i^{n-k}(t) = \dots = P_0^{(n)}(t),$$

最后 n 次 Bézier 曲线从形式上“降阶”为 0 次 Bézier 曲线(即一点) $P_0^{(n)}(t)$, 即我们要计算的曲线上的点 $P(t)$ 。这就是 Bézier 曲线的 De Casteljau 算法, 即为如下递归求值算法:

$$\begin{cases} P_i^{(0)}(t) = P_i^{(0)} = P_i, i = 0, 1, \dots, n, \\ P_i^{(k)}(t) = (1-t)P_i^{(k-1)}(t) + tP_{i+1}^{(k-1)}(t), i = 0, 1, \dots, n-k; k = 1, 2, \dots, n. \end{cases} \quad (2)$$

上标 k 表示递推级数, 每进行一级递推, 控制顶点少一个, 所得中间控制顶点都与参数 t 有关。

求交算法如下:

算法	三次 Bézier 曲线交点查找算法
输入	两组控制顶点: $C1 = [0, 0; 2, 3; 5, 5; 0, 7]$, $C2 = [0, 5; 3, 2; 5, 3; 7, 5]$ (可替换为任意控制点)
输出	相交时: 返回交点坐标 + 迭代次数, 不相交: 提示“无交点”
Step1	初始化控制点与绘图 1) 定义控制点 $C1, C2$ 2) 生成 100 个参数 t (linspace) 3) 计算曲线坐标(bezierCurve) $\rightarrow$ bezierC1/bezierC2 4) 绘制双曲线(蓝/红, 线宽 1.5)
Step2	凸包处理 1) 计算凸包 $K1/K2$ (convhull) 2) 绘制虚线凸包边界 3) 凸包相交检测(convexHullIntersect): - 不相交 $\rightarrow$ 终止 - 相交 $\rightarrow$ 继续
Step3	变量初始化 $currentC1/C2 \leftarrow C1/C2$ $n = 0, tolerance = 1e^{-5}$ 准备 7 种绘图颜色 $intersectionPoint = []$
Step4	迭代处理(De Casteljau subdivision) 1) 分割曲线(deCasteljauSplit): - 生成 leftC1/rightC1, leftC2/rightC2 - 记录中间点 midC1/midC2 2) 可视化分割过程(plotDeCasteljau): - 绘制控制多边形 - 标记中间点(不同颜色)
Step5	子曲线相交检测 1) 检查 4 种子组合(findIntersectingSubCurves): $leftC1 \times leftC2/leftC1 \times rightC2/rightC1 \times leftC2/rightC1 \times rightC2$ 2) 返回相交子对 subC1/subC2 及方向
Step6	收敛判断 1) 计算曲线尺寸(checkConvergence) 2) 若 $\max(sizeC1, sizeC2) < 1e^{-5}$: - 取中点作为交点 - 退出循环 否则: - 更新 $currentC1/C2 \leftarrow subC1/subC2$ - $n++$
Step7	结果输出 1) 绘制黄色五角星交点标记 2) 标注坐标值 3) 命令行输出坐标 + 迭代次数 4) $C1, C2$ 对应的参数 $t1, t2$

续表

辅助函数	
bezierCurve	根据公式： $B(t) = \sum b_i B_{i,n}(t)$ 计算曲线点
convexHullIntersect	分离轴定理：检查多边形在所有边法向量上的投影重叠
project	计算多边形在给定轴上的[minVal, maxVal]投影范围
deCasteljauSplit	递归分割控制点→返回左/右子曲线 + 3 层中间点
plotDeCasteljau	可视化：控制多边形(虚线)、中间点(彩色)、连线
findIntersectingSubCurves	测试 4 种子曲线组合的凸包相交性
checkConvergence	判断曲线范围 $\max(\Delta x, \Delta y)$ 是否 $1e^{-5}$

De Casteljau 算法通过递归细分 Bézier 曲线，将求交问题转化为控制多边形检测：每次分割后，排除控制多边形不相交的子段，对可能相交的区间继续进行二分处理，直至逐步逼近曲线交点。这种分治策略利用曲线的特性，在保证一定精度的同时，显著减少算法所需的计算量。

De Casteljau 算法作为一种基于几何特性的细分算法，在 Bézier 曲线求交问题上展现出显著优势。相较于代数法(受限于高次方程求解的数值稳定性)和传统细分法(效率受制于全局均匀细分) [4]，该算法采用二分策略：通过控制多边形检测快速剔除非交区域，显著提升计算效率；同时其递归细分过程自然保持几何精度，避免了代数方法的数值误差累积问题[5]。特别适合处理高次 Bézier 曲线的局部精确求交。相比之下，代数方法仅适用于低次曲线，牛顿迭代等数值方法则严重依赖初始值选取，而 De Casteljau 算法能够保持更高的数值稳定性和计算效率。

5. 数值实例

例 1：给定两条三次 Bézier 曲线的控制顶点，分别为 $C1 = [0, 0; 1, 2; 3, 4; 4, 0]$ ， $C2 = [1, 3; 3, 2; 2, 3; 4, 4]$ ，其具体图形为图 1 所示，经算法检测，两条曲线的凸包不相交，因此判定两条曲线不相交。

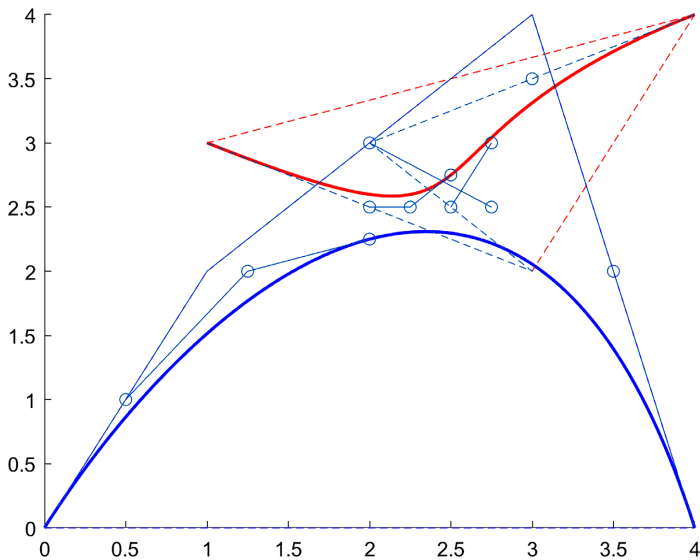


Figure 1. Two Bézier curves without intersection points
图 1. 无交点的两条 Bézier 曲线

例 2: 给定两条三次 Bézier 曲线的控制顶点, 分别为 $C1 = [0, 0; 2, 3; 5, 5; 0, 7]$, $C2 = [0, 5; 3, 2; 5, 3; 7, 5]$, 其具体图形为图 2 所示, 经算法检测, 两条曲线的凸包相交, 进一步通过迭代分割检测, 确认两条曲线的交点坐标。

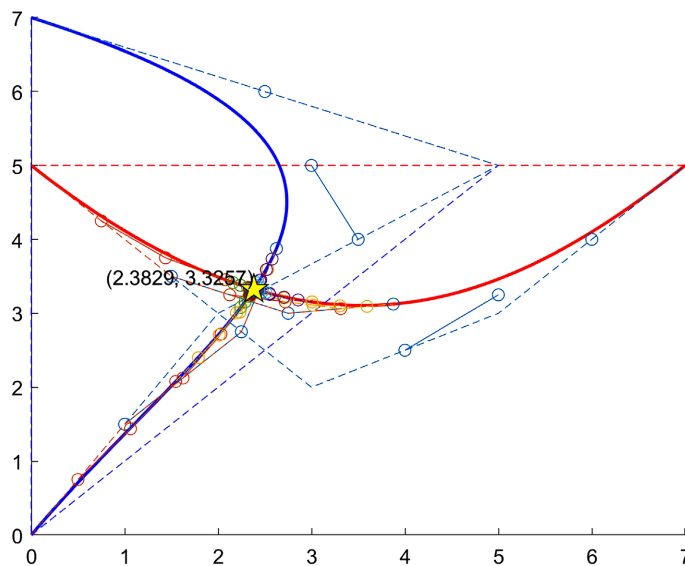


Figure 2. Two Bézier curves with intersection points

图 2. 有交点的两条 Bézier 曲线

在 MATLAB 后端运行算法, 得到凸包相交处理信号, 如图 3 所示; 并获取最终 Bézier 曲线交点坐标, 迭代次数及相交时 C1 C2 对应的参数 t_1 、 t_2 , 如图 4 所示, 整个过程流程图如图 5 所示。

凸包相交, 继续处理

迭代 0 的中间点:

c1中间点:

Layer1:

1.0000 1.5000

3.5000 4.0000

Figure 3. Convex hull intersection and midpoint iteration diagram

图 3. 凸包交集与中点迭代图

方向选择: c1-右, c2-右

交点坐标: (2.3829, 3.3257)

迭代次数: 19

C1对应参数 $t_1 \in [0, 1]$, $t_1 = 0.4201211929$

C2对应参数 $t_2 \in [0, 1]$, $t_2 = 0.2901029587$

Figure 4. Bézier curve intersection and parameter diagram

图 4. Bézier 曲线交点及参数示意图

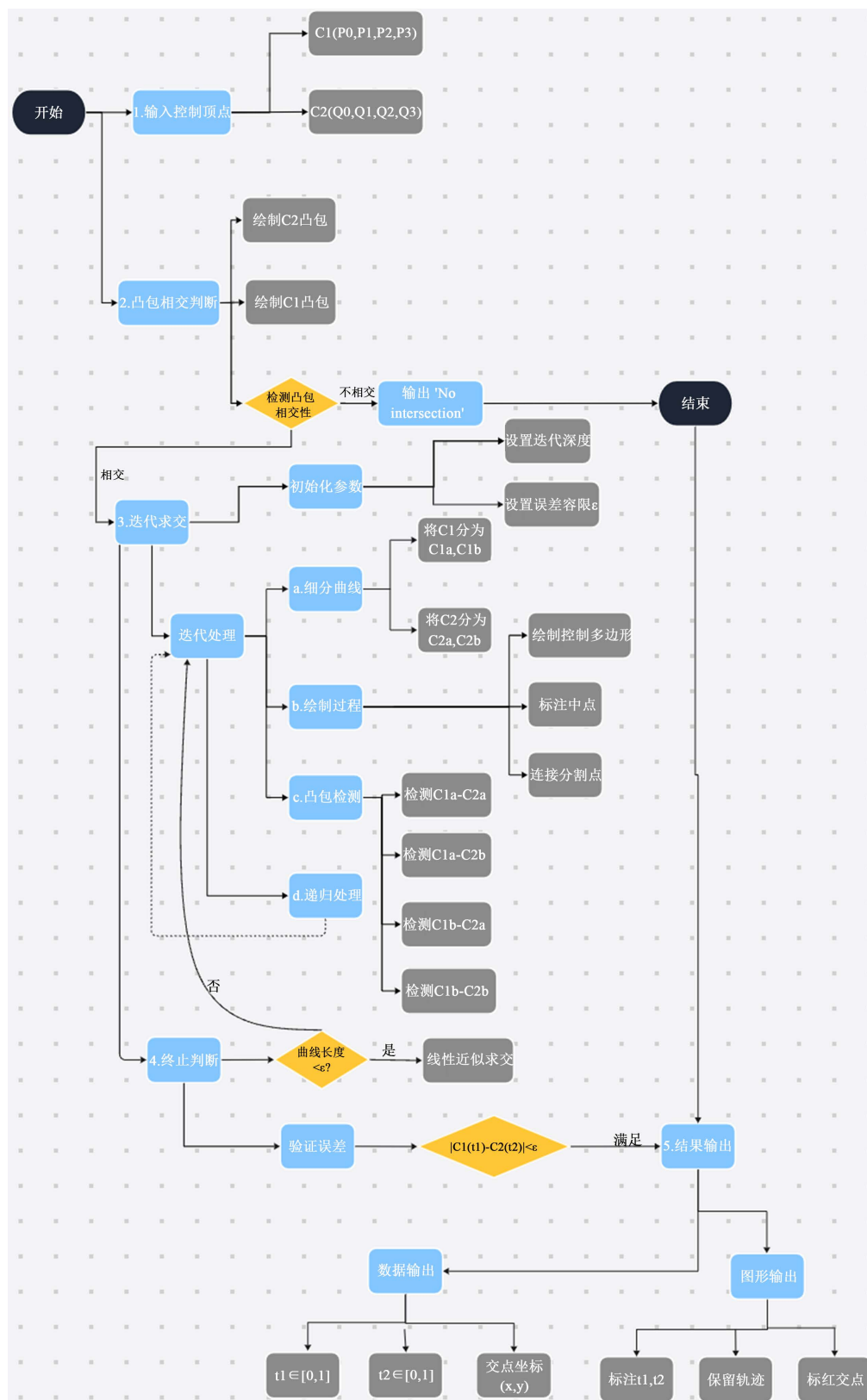


Figure 5. Flowchart
图 5. 流程图

6. 结论

本文围绕基于迭代算法的 Bézier 曲线求交方法展开深入研究,以控制多边形的相交情况作为核心判断依据,构建出一套完整的求交策略。经研究发现,当 Bézier 曲线的控制多边形不相交时,曲线本身必然不相交,这一特性为快速筛选不相交曲线对提供了简洁高效的方法,有效避免了冗余计算,显著提升了处理大规模曲线数据时的前期筛选效率。针对控制多边形相交的情形,本文采用 De Casteljau 开展迭代运算。该方法以其简洁的逻辑和高效的运算过程,在众多测试案例中展现出良好的适应性。通过逐步逼近的策略,精准定位交点位置,在保证计算精度的同时,大幅减少了不必要的计算步骤。在与传统算法的对比实验中,基于 De Casteljau 算法的迭代求解策略在处理复杂曲线时,展现出更强的稳定性和更快的收敛速度,有力地证明了该方法在解决 Bézier 曲线求交问题上的有效性与优越性。然而,该方法也存在一定局限性。在处理具有极高阶数的 Bézier 曲线时,由于算法迭代次数增多以及浮点数运算精度损失的累积,计算效率会有所下降,且求交结果的准确性可能受到影响。此外,当曲线数量庞大且分布复杂时,控制多边形相交情况的判断开销会显著增加,进而对整体算法性能产生制约。未来研究可以考虑以下具体优化方向:在数值计算方面,可引入基于区间算术的自适应精度控制方法[6],通过动态调整浮点运算精度来平衡高阶曲线的计算效率与准确性;同时结合误差补偿技术,减少迭代过程中的累积误差。在空间索引方面,可采用基于四叉树或 BVH 层次包围盒的空间划分结构,通过预计算建立空间索引来加速控制多边形的相交检测。此外,针对特定应用场景(如 CAD 图形处理),可研究基于 GPU 并行计算的加速方案,利用 CUDA 等框架实现控制多边形相交判断的并行化处理。这些优化方案可以先通过 MATLAB/Python 进行算法验证,再使用 C++进行性能优化,最终形成完整的优化解决方案。从而探索更高效的空间索引结构,改进控制多边形相交判断流程,以进一步拓展该方法的应用范围与性能表现。

基金项目

辽宁师范大学教师指导本科生科研训练项目(项目编号:JSZDBKSXM2024077);

辽宁师范大学 2025 年大学生创新创业训练计划项目(项目名称:障通无阻,安全智行——基于有理 Bézier 曲线的自动驾驶智能避障规划系统研究及应用)。

参考文献

- [1] 马秀红,宋丽娟,黄守丽. 基于 Bezier 曲线的图形绘制技术的研究[J]. 中国新通信, 2024, 26(9): 11-14.
- [2] 朱春钢,李彩云. 数值逼近与计算几何[M]. 北京: 高等教育出版社, 2020: 210-249.
- [3] 张锋,董旭华. 基于 de Casteljau 算法的 Bézier 曲线细分[J]. 西南民族大学学报(自然科学版), 2008, 34(6): 1118-1121.
- [4] 丁永胜. 细分曲线的形状控制及应用[D]: [硕士学位论文]. 西安: 西北工业大学, 2005.
- [5] 陈小雕,段晓慧,杨超,等. 2 条有理三次 Bézier 曲线的部分重合条件[J]. 计算机辅助设计与图形学学报, 2015, 27(9): 1648-1652.
- [6] 崔长彩,黄富贵,范伟,等. 自适应迭代区域搜索法用于直线度精密评定[J]. 机械科学与技术, 2010, 29(11): 1525-1529.