

两条 NURBS 曲线求交的高效迭代算法

杨子函, 高竞闻, 陈小雕*

(杭州电子科技大学计算机学院 杭州 310018)
(xiaodiao@hdu.edu.cn)

摘要: 针对 NURBS 曲线求交问题, 提出一种基于有理函数插值的高效迭代算法. 针对传统方法如牛顿法和割线法对初始值敏感且难以处理相切情况, 隐式化法和纯细分法存在计算复杂度高或数值不稳定问题, 提出结合细分法的新型有理函数插值迭代策略, 有效地提升求交计算效率和迭代计算稳定性. 首先构造双点导数约束的有理插值函数, 逼近曲线参数方程; 然后推导出一元二次方程获取相应的迭代公式, 实现高效的收敛. 实验包含贯穿、相切及 4 000 例不同次数曲线的基准测试. 结果表明, 与现有的同类迭代法相比, 在相切的情况下, 所提算法效率提高约 200%; 数值实例结果表明, 该算法的整体求交效率是 OCCT 几何内核的 5~50 倍, 比 ACIS 几何内核高 21%~180%, 证明了其有效性.

关键词: B 样条曲线/曲线求交; 有理函数插值; 迭代; 相切

中图分类号: TP391.41 **DOI:** 10.3724/SP.J.1089.2025-00139

Efficient Iterative Method for Intersecting Two NURBS Curves

Yang Zihan, Gao Jingwen, and Chen Xiaodiao*

(College of Computer, Hangzhou Dianzi University, Hangzhou 310018)

Abstract: This paper presents an efficient iterative algorithm using rational function interpolation for NURBS curve intersection. Traditional methods, like Newton's and Secant methods, are sensitive to initial values and struggle with tangential cases, while implicitization and subdivision methods exhibit high computational complexity or instability. To address these shortcomings, we developed a novel iterative strategy that integrates rational function interpolation with a subdivision method, which significantly improves both the efficiency and stability of the intersection calculation. The algorithm first constructs a rational interpolation function, constrained by two-point derivatives, to approximate the curves' parametric equations. It then derives a corresponding iterative formula from a quadratic equation to achieve rapid convergence. We performed extensive experiments, including scenarios with crossing and tangential intersections, alongside a benchmark test involving 4 000 curves of varying degrees. Results showed that for tangential intersections, our algorithm is approximately 200% more efficient than comparable iterative methods. Furthermore, numerical examples demonstrate that the overall intersection efficiency is 5 to 50 times that of the OCCT and 21% to 180% higher than the ACIS, confirming the algorithm's effectiveness.

Key words: B-spline curve/curve intersection; rational function interpolation; iteration; tangent

曲线与曲线求交问题在 CAD&CG 领域有着广泛的应用, 如交互设计、布尔运算和数控加工路径

规划等^[1-4]. 本文讨论 2 条参数曲线的求交问题, 非均匀有理 B 样条曲线(non-uniform rational B-spline,

收稿日期: 2025-04-30; 修回日期: 2025-09-14. 基金项目: 国家自然科学基金面上项目(61972120); 工信部项目(CEIEC-2024-ZM02-0073). 杨子函(2001—), 男, 硕士研究生, 主要研究方向为 CAGD; 高竞闻(2001—), 男, 硕士研究生, 主要研究方向为计算机图形学; 陈小雕(1976—), 男, 博士, 教授, 博士生导师, CCF 会员, 论文通信作者, 主要研究方向为 CAD、CAGD.

NURBS)间求交问题的核心通常是求解非线性方程组, 传统的迭代方法(如牛顿法^[5]和割线法^[6])因具有快速收敛性, 被广泛应用于求解过程中. 然而, 牛顿法的高效性依赖于精确的导数计算和良好的初始解, 这在实际应用中并不总是可行, 特别是在处理高次或复杂曲线相交时, 这种依赖性更加突出.

为了提高算法鲁棒性, 许多研究采用裁剪法^[7]或隐式化法^[8-9]. 裁剪法具有良好的稳定性, 能够有效地排除非解区域, 但计算复杂度和内存需求高^[7]. 隐式化法通过将曲线求交转化为代数方程求解, 但对于 5 次或更高次曲线上可能面临数值不稳定问题^[10].

近年来, 为了兼顾数值稳定性和计算效率, 研究人员提出了多种混合方法, 如鸡尾酒算法^[11]、混合裁剪法^[4,12]、区间细分法^[13]和迭代细分法^[14]. 这些方法结合稳定的裁剪方法以及快速收敛的迭代方法, 以获得更好的性能. 在给定初始值良好的情况下, 使用牛顿迭代求解曲线/曲线求交问题被认为是最快的算法^[11], 但在处理曲线相切的情况时, 收敛速度仍不够理想.

为了克服上述问题, 本文提出一种有理插值迭代算法, 其具有更优的时间效率或收敛阶数, 可适用于相切情况. 通过构造有理函数的插值公式, 本文算法能够有效地减少对初始值的依赖, 显著地提高计算效率.

1 相关工作

1.1 问题描述

设 $N_i^m(t)$ 表示 B 样条基函数, 则 2 条待计算的曲线分别为

$$C_1(u) = \frac{\sum_{i=0}^n N_{i,p}(u) w_i P_i}{\sum_{i=0}^n N_{i,p}(u) w_i}, 0 \leq u \leq 1 \quad (1)$$

$$C_2(v) = \frac{\sum_{i=0}^n N_{i,p}(v) w_i Q_i}{\sum_{i=0}^n N_{i,p}(v) w_i}, 0 \leq v \leq 1 \quad (2)$$

其中, $\{P_i\}$ 表示控制顶点; $\{w_i\}$ 表示权因子. 对应的求交问题需要求解所有满足 $C_1(u) - C_2(v) = 0$, $0 \leq u, v \leq 1$ 的 (u, v) 点对.

1.2 研究现状

王永澳等^[15]介绍了 3 种常用的曲线/曲线求交算法: 牛顿法^[16]、割线法^[6]和裁剪法^[7]. 牛顿法具有二阶收敛速度, 依赖初始猜测的准确性且雅可比矩阵非奇异. 割线法收敛阶为 1.618, 避免计算雅可比矩阵, 比牛顿法单次迭代计算量更低, 但同样依赖初始值的准确性且需要更多迭代次数. 裁剪法利用几何条件快速排除无交区域, 平均时间复杂度为 $O(n \log n)$, 当计算精度要求很高时效率骤降. 对于高精度相切情况, 3 种方法的时间效率都很低.

此外, 常用方法还有细分法^[17]和隐式化法^[8-9]等. 细分法递归分割参数空间, 直到子曲线段足够简单, 再进行快速相交测试和排除, 平均时间复杂度为 $O(n \log n)$, 当计算精度要求很高时效率骤降. 隐式化法通过代数消元直接求根, 该方法理论上很直接, 并且可以避免迭代法对于初始值的依赖; 但是对于高次有理曲线, 隐式化过程本身会产生高次多项式, 在实际应用中存在隐式化困难、数值不稳定、仅适用于低次代数曲线的问题.

2 本文算法

本文算法中, 迭代分为 2 步: (1) 提出一种基于有理函数的插值方法, 用于近似求解函数; (2) 求解非线性方程组, 用于迭代计算精确交点.

2.1 前置算法

本文算法的迭代过程指由初始点收敛至精确交点, 初始点信息可由细分法计算得到. Yap^[17]提出了完整的细分方法, 给定 2 条 Bézier 曲线

$$C_i(t) = \sum_{j=0}^{n_i} P_{i,j} B_j^{n_i}(t), i=1,2; \text{ 其中, } P_{i,j} \text{ 表示控制}$$

顶点, $B_j^{n_i}(t) = C_{n_i}^j(1-t)^{n_i-j} t^j$ 表示 Bézier 基函数.

前置算法把整条曲线看作一对候选片段, 放入一个待处理队列 Q ; 然后反复地从队列中取出一对曲线片段(称作 A 和 B). 如果它们的凸包没有重叠, 则直接舍弃, 继续下一对, 直到找到有重叠的一对曲线片段. 设 Δ 为曲线细分对应的阈值, 根据 A 和 B 各自的控制顶点计算得到二者最大弦高

$$h_{\max}^i = \max_{j=1}^{n_i-1} \left(\frac{\left| (P_j - P_0) \times (P_{n_i} - P_0) \right|}{\|P_{n_i} - P_0\|} \right), i=1,2.$$

若 Δ 值太小, 会导致细分次数过多, 整体耗时迅速

上升; 否则, 可能会导致在高曲率区域遗漏初始点, 或曲线近似粗糙, 给出的初始点质量低下, 迭代无法收敛. 根据经验, 本文设置 $\Delta = 0.01$.

如果 $h_{\max}^i \leq \Delta, i=1,2$, 则进入求交点阶段; 否则, 将更大长度的曲线按照参数中点切两半, 得到 A_1 和 A_2 , 再将 (A_1, B) 和 (A_2, B) 放回队列 Q .

在求交阶段, 将 2 条小曲线段看作线段, 用线段相交的方法快速判断它们是否有交点, 并将相应的 2 条线段的交点作为后续迭代的候选初始点插入集合 Ω .

传统的牛顿迭代等方法对于初始值的选取相对敏感, 且在相切情况下收敛阶会退化为一阶. 为了解决上述问题, 本文提出基于有理函数插值的迭代算法. 实验结果表明, 该算法可以取得更高的收敛阶和计算效率.

2.2 算法步骤

算法 1. 2 条 NURBS 曲线的求交算法.

输入. 2 条 NURBS 曲线 $C_1(u)$ 和 $C_2(v)$, $u, v \in [0,1]$, 并给定的容差 ε .

输出. 交点集合 A .

Step1. 采用细分法^[17]初步筛选出初始点对的集合 Ω .

Step2. 如果 $\Omega = \emptyset$, 转 Step7.

Step3. 从 Ω 中弹出一个点对作为初始值 (u_i, v_i) , 对其应用一次牛顿迭代获得下一个点对 (u_{i+1}, v_{i+1}) .

Step4. 如果 $|(u_{i+1}, v_{i+1}) - (u_i, v_i)| \leq \varepsilon$, 转 Step6.

Step5. 使用有理函数插值 $C_1(u) = C_2(v)$ 得到插值迭代公式(见第 2.3 节), 应用公式获取下一组点对 (u_{i+2}, v_{i+2}) . 更新 $(u_i, v_i) = (u_{i+1}, v_{i+1}), (u_{i+1}, v_{i+1}) = (u_{i+2}, v_{i+2})$. 转 Step4.

Step6. 将精确解 $(u_i^*, v_i^*) = (u_{i+1}, v_{i+1})$ 放入解集 A 中. 转 Step2.

Step7. 输出 A 作为最终结果.

2.3 基于有理函数插值的迭代方法

牛顿法等传统迭代方法在处理曲线相切时收敛速度显著下降, 主要原因是相切点附近雅可比矩阵趋近于奇异或病态, 导致迭代方向难以准确指向根的位置. 为了克服这一瓶颈并提升迭代效率, 特别是在相切情形下的鲁棒性和收敛速度, 本文采用有理函数插值构建局部逼近模型.

与非有理多项式插值相比, 有理函数的分母项为其提供了额外的自由度, 具有更强的函数逼近能力, 能够更好地适应目标函数在插值点附近的导数和曲率特征, 获得比同阶非有理多项式更好的局部近似, 对于相切情况的处理尤为重要.

本文采用类似二阶泰勒展开的方式求解相应的插值有理多项式. 对于算法 1 中 Step5 的每次迭代, 有式(1)(2)定义的 2 条 NURBS 曲线 $C_1(u)$ 和 $C_2(v)$, 以及初始点 (u_i, v_i) 和初始点经过一次牛顿迭代获得的下一个点对 (u_{i+1}, v_{i+1}) .

曲线 1 和曲线 2 分别可以表示为

$$\begin{aligned} C_1(u) &= \begin{cases} x = x_1(u) \\ y = y_1(u) \end{cases}, \\ C_2(v) &= \begin{cases} x = x_2(v) \\ y = y_2(v) \end{cases}. \end{aligned}$$

若 2 条曲线求交, 则对应的方程组为

$$\begin{cases} X(u, v) = x_1(u) - x_2(v) = 0 \\ Y(u, v) = y_1(u) - y_2(v) = 0 \end{cases} \quad (3)$$

以 $X(u, v)$ 为例, 由于有理函数插值具有逼近复杂函数效果佳、低阶插值精度比多项式插值高、可以缓解龙格现象等优势^[18], 使用有理函数 $\bar{X}(u, v)$ 插值逼近 $X(u, v)$. 其中,

$$\bar{X}(u, v) = \frac{a + bu + cv + duv}{1 + Au^2 + Bv^2} \quad (4)$$

为了求解式(4)的未知系数, 取 $X(u, v)$ 在 (u_i, v_i) 和 (u_{i+1}, v_{i+1}) 这 2 个点处的函数值, 偏导数值列方程组为

$$\begin{cases} \bar{X}(u_i, v_i) = X^i \\ \bar{X}'_u(u_i, v_i) = X^i_u \\ \bar{X}'_v(u_i, v_i) = X^i_v \\ \bar{X}(u_{i+1}, v_{i+1}) = X^{i+1} \\ \bar{X}'_u(u_{i+1}, v_{i+1}) = X^{i+1}_u \\ \bar{X}'_v(u_{i+1}, v_{i+1}) = X^{i+1}_v \end{cases} \quad (5)$$

其中, $X^i = X(u_i, v_i)$, $X^i_u = \frac{\partial X}{\partial u}(u_i, v_i)$, $X^i_v = \frac{\partial X}{\partial v}(u_i, v_i)$.

特别地, 为了降低式(5)的求解复杂度, 对式(4)进行预处理

$$\bar{X}(u, v) = \frac{a_1 + b_1 \delta u + c_1 \delta v + d_1 \delta u \delta v}{1 + A_1 \delta u^2 + B_1 \delta v^2}.$$

其中, $\delta u = u - u_{i+1}$, $\delta v = v - v_{i+1}$. 由 Maple 进行公式推导, 计算系数结果为

$$\begin{cases} a_1 = X^{i+1} \\ b_1 = X^{i+1}_u \\ c_1 = X^{i+1}_v \\ d_1 = \frac{-MN - X^i N - X^{i+1} M + 2X^i \Delta X}{\Delta u \Delta v M} \end{cases}.$$

其中, $\Delta u = u_i - u_{i+1}$, $\Delta v = v_i - v_{i+1}$, $\Delta X = X^i - X^{i+1}$,
 $M = X_u^i \Delta u + X_v^i \Delta v$, $N = X_u^{i+1} \Delta u + X_v^{i+1} \Delta v$.

同理, 可以构造有理函数 $\bar{Y}(u, v)$ 插值逼近 $Y(u, v)$.

下一步, 根据式(3)列迭代方程组为

$$\begin{cases} a_1 + b_1 \delta u + c_1 \delta v + d_1 \delta u \delta v = 0 \\ a_2 + b_2 \delta u + c_2 \delta v + d_2 \delta u \delta v = 0 \end{cases} \quad (6)$$

求解式(6)得到 2 组根, 选择距离 (u_{i+1}, v_{i+1}) 欧几里得距离更近的根作为下一个迭代点 (u_{i+2}, v_{i+2}) , 公式为

$$\begin{cases} v_{i+2} = \frac{L \pm \Delta}{2(c_1 d_2 - c_2 d_1)} \\ u_{i+2} = \frac{(-d_1 u_{i+1} + c_1) \Delta + M}{\mp d_1 \Delta + N} \end{cases}$$

其中,

$$\begin{cases} \Delta = (a_1^2 d_2^2 - 2a_1 a_2 d_1 d_2 - 2b_1 c_2 d_2 a_1 - \\ 2b_2 (c_1 d_2 - 2c_2 d_1) a_1 + a_2^2 d_1^2 + \\ ((4c_1 d_2 - 2c_2 d_1) b_1 - 2b_2 c_1 d_1) a_2 + \\ (b_1 c_2 - b_2 c_1)^2)^{\frac{1}{2}} \\ L = (2d_2 v_{i+1} - b_2) c_1 + (-2d_1 v_{i+1} + b_1) c_2 - a_1 d_2 + a_2 d_1 \\ M = -a_2 d_1^2 u_{i+1} + ((b_2 u_{i+1} + a_2) c_1 + \\ (a_1 d_2 + b_1 c_2) u_{i+1} - 2a_1 c_2) d_1 + \\ c_1 (-2b_1 d_2 u_{i+1} + a_1 d_2 + b_1 c_2 - b_2 c_1) \\ N = -a_2 d_1^2 + (a_1 d_2 + b_1 c_2 + b_2 c_1) d_1 - 2b_1 c_1 d_2 \end{cases}$$

至此, 完成算法 1 的 Step5. 通过本文提出的基于有理函数插值的迭代方法, 可以高效、精确地得到曲线 $C_1(u)$ 和 $C_2(v)$ 的所有交点.

2.4 插值函数的误差分析

精确函数 $X(u, v)$ 与插值函数 $\bar{X}(u, v)$ 的误差定义为 $E_X(u, v) = X(u, v) - \bar{X}(u, v)$. 插值函数为有理函数, 虽然分母的设计可以提升逼近能力, 但是误差分析仍然需要基于一般原理.

误差在插值点处值和梯度均为 0, 误差的阶取决于函数的平滑度和点间距. 设 $h_u = |u_{i+1} - u_i|$, $h_v = |v_{i+1} - v_i|$, 并以 $h_m = \max(h_u, h_v)$ 为点间距. 假设精确函数 $X(u, v)$ 是充分光滑的, 则误差可用泰勒展开余项分析, 若采用非有理二次多项式插值, 则式(5)等价二阶泰勒展开, 相应的逼近阶为三阶, 即误差主要来源于泰勒级数三阶项. 设误差表示为

$$E(u, v) = \frac{1}{2} \delta H(\xi, \eta) \delta^T.$$

其中, $\delta = [\delta u, \delta v]$, $\xi \in (u_i, u_{i+1})$, $\eta \in (v_i, v_{i+1})$, $H(\xi, \eta)$ 为 Hessian 矩阵. 根据泰勒定理^[19]和 Lipschitz 条件估计, 当 $u \in (u_i, u_{i+1})$, $v \in (v_i, v_{i+1})$ 时, 可得

$$|E(u, v)| = O(h_m^3) \quad (7)$$

2.5 单根情形的收敛阶分析

设精确根 $s^* = (u^*, v^*)$ 满足 $F(s^*) = 0$, 其中, $F(u, v) = (X(u, v), Y(u, v))$, 由点 $s_i = (u_i, v_i)$ 处插值构造的近似函数为 $\bar{F}_i(u, v) = (\bar{X}_i(u, v), \bar{Y}_i(u, v))$ 满足 $\bar{F}_i(s_i) = 0$. 令 $h_i = \|s^* - s_i\|$, 则有

$$\|s_{i+1} - s_i\| = \|s^* - s_i + s_{i+1} - s^*\| = O(h_i).$$

根据式(7)可得

$$\|\bar{F}_{i+1}(s^*)\| = \|(E_{X,i+1}(s^*), E_{Y,i+1}(s^*))\| = O(h_i^3) \quad (8)$$

其中, K 是常数. 假设 $\bar{F}_{i+1}$ 在 s^* 附近连续可微, 且其雅可比矩阵 $J_{\bar{F}}(u, v)$ 在 s^* 处非奇异, 由中值定理, 存在点 ξ 是 $[s^*, s_{i+1}]$ 上的一点, 使得

$$\bar{F}_{i+1}(s^*) = \bar{F}(s^*) - \bar{F}_{i+1}(s_{i+1}) = J_{\bar{F},i+1}(\xi)(s^* - s_{i+1}),$$

则有

$$s^* - s_{i+1} = [J_{\bar{F},i+1}(\xi)]^{-1} \bar{F}_{i+1}(s^*) \quad (9)$$

结合式(7)~式(9), 得

$$\|s^* - s_{i+1}\| = O(\|s^* - s_i\|^3) \quad (10)$$

由式(10)可知, 本文迭代方法为三阶收敛.

3 数值算例及讨论

通常, NURBS 曲线求交问题可以分成 2 步解决: 首先使用细分法、裁剪法等获取用于迭代的初始点; 然后使用迭代方案获取精确交点, 这也是本文提出的基于有理函数插值迭代方法(简称为 M3)所关注的问题. 将 M3 分别与牛顿法(简称为 M1)、改进的 Newton-Özban 组合法(简称为 M2)进行实验.

本文中, 数值实验在 13th Gen Intel(R) Core(TM) i5-13500H CPU, 32 GB 内存的 PC 机上使用 C++ 语言实现.

为了方便起见, 使用的符号定义如下: K 表示迭代次数; T_{avg} 表示平均迭代时间, T_{tot} 表示总迭代时间, 单位为 ms.

例 1. 对于贯穿性相交情况, 根据式(1)(2)给出

3 组 NURBS 曲线进行实验, 分别交于 1 个点、2 个点和 3 个点, 结果如图 1 所示. 其中, 节点向量均

为 $U = \{0, 0, 0, 0, 1, 1, 1, 1\}$, 权重向量 $w = (1, 0.7, 0.7, 1)$. 控制顶点如表 1 所示.

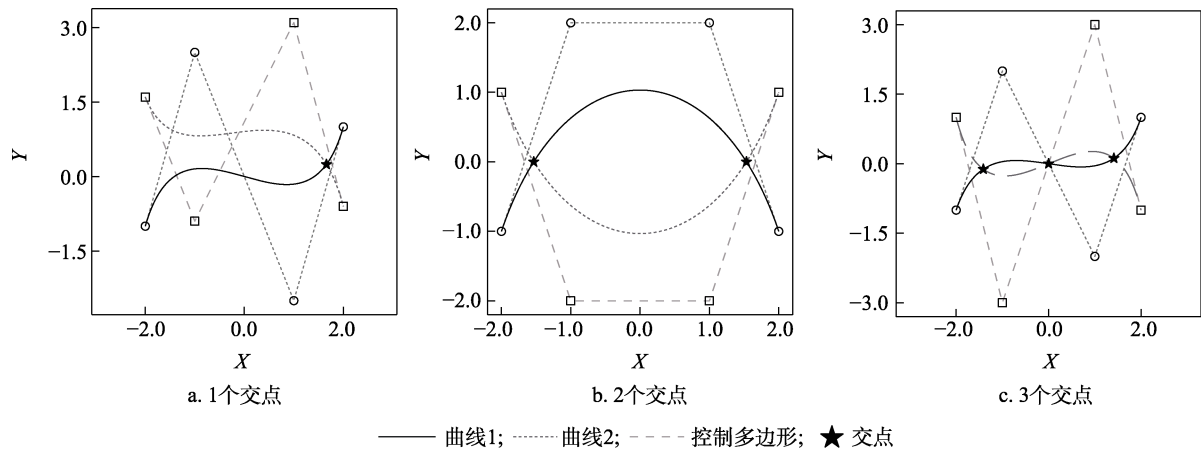


图 1 3 对贯穿性相交曲线对比

表 1 3 组曲线控制顶点

曲线	控制顶点
$C_1^1(u)$	$[(-2, -1), (-1, 2.5), (1, -2.5), (2, 1)]$
$C_2^1(v)$	$[(-2, 1.6), (-1, -0.9), (1, 3.1), (2, -0.6)]$
$C_1^2(u)$	$[(-2, -1), (-1, 2), (1, 2), (2, -1)]$
$C_2^2(v)$	$[(-2, 1), (-1, -2), (1, -2), (2, 1)]$
$C_1^3(u)$	$[(-2, -1), (-1, -2), (1, -2), (2, 1)]$
$C_2^3(v)$	$[(-2, 1), (-1, -3), (1, 3), (2, -1)]$

对图 1 所示的 3 种贯穿性相交情况, 在不同容差下对 3 种迭代算法进行实验, 结果如表 2 所示. 可以看出, M2 的收敛速度最高, 但与其他算法相比, 每次迭代的计算代价时间消耗要高出 1 个数量级; M1 与 M3 每次迭代耗时接近, 但 M3 的收敛速度高于 M1; M3 取得了算法在时间消耗与收敛速度方面的平衡, 达到整体问题求解的时间开销最小.

表 2 贯穿性相交情况下不同容差时 3 种迭代算法对比

实例	方法	K					T_{avg}/ms	T_{tot}/ms	
		10^{-3}	10^{-6}	10^{-9}	10^{-12}	10^{-15}		10^{-6}	10^{-15}
图 1a	M1	2	4	4	5	6	0.004 26	0.017	0.025
	M2	1	2	2	2	3	0.020 14	0.040	0.060
	M3	2	3	4	4	5	0.004 93	0.014	0.024
图 1b	M1	2	4	4	5	6	0.003 97	0.015	0.023
	M2	1	2	2	2	3	0.018 99	0.037	0.056
	M3	2	3	4	4	5	0.004 89	0.014	0.024
图 1c	M1	2	3	4	5	6	0.005 03	0.015	0.030
	M2	1	2	2	2	3	0.024 58	0.049	0.073
	M3	2	3	4	4	5	0.004 43	0.013	0.022

在不同容差下, 将本文算法与常用的三维 CAD 几何建模内核引擎 OCCT 和 ACIS 的时间效率进行实验, 结果如表 3 所示. 其中, 比例指以本文算法为基准, 相应算法与本文算法的时间之比. 可以看出, 在相同精度下, 本文算法具有较高的性能优势, 其效率平均是 OCCT 的 5 倍, ACIS 的 3 倍, 展现出其显著的工程应用价值.

例 2. 在 2 组相切情况下, 对 NURBS 曲线求交实例进行实验, 结果如图 2 所示. 图 2a 中, 曲线的节点向量为 $U = \{0, 0, 0, 0, 1, 1, 1, 1\}$, 权重向量为 $w = (1, 0.7, 0.7, 1)$; 图 2b 中, 曲线的节点向量为 $U = \{0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1\}$, 权重向量为 $w = (1, 1, 0.7, 0.7, 0.7, 0.7, 1, 1)$. 控制顶点如表 4 所示.

表 3 贯穿性相交情况下不同容差时 3 种求交算法时间效率对比

实例	算法	10^{-6}		10^{-9}		10^{-12}	
		时间/ms	比例	时间/ms	比例	时间/ms	比例
图 1a	OCCT	0.148	8.70	0.187	9.35	0.196	9.33
	ACIS	0.071	4.17	0.075	3.75	0.089	4.23
	本文	0.017	1.00	0.020	1.00	0.021	1.00
图 1b	OCCT	0.173	4.67	0.222	5.16	0.237	4.47
	ACIS	0.117	3.16	0.123	2.86	0.133	2.50
	本文	0.037	1.00	0.043	1.00	0.053	1.00
图 1c	OCCT	0.196	5.02	0.256	3.28	0.348	3.78
	ACIS	0.190	4.87	0.213	2.73	0.218	2.36
	本文	0.039	1.00	0.078	1.00	0.092	1.00

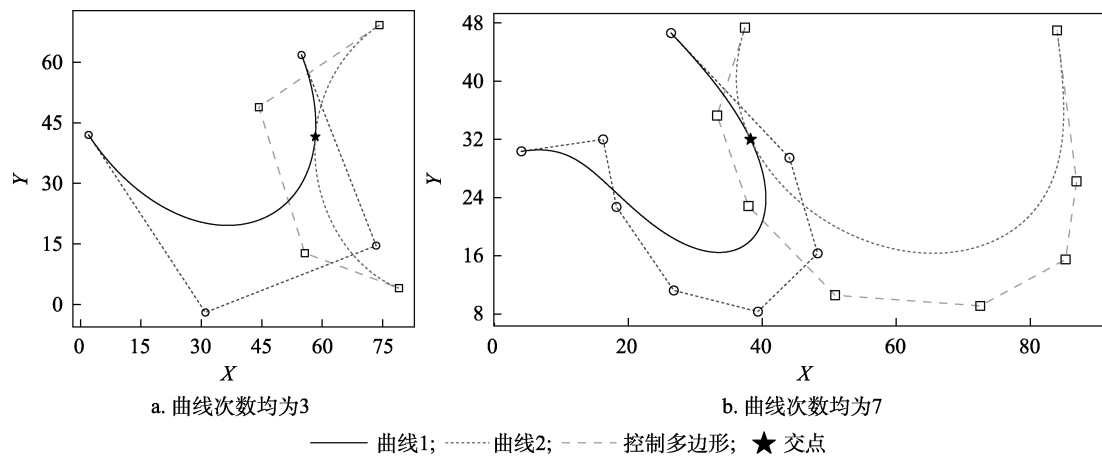


图 2 2 对相切曲线比较

表 4 2 组曲线控制顶点

曲线	控制顶点
$C_1^4(u)$	[(2, 42), (31, -2), (73, 14), (54, 61)]
$C_2^4(v)$	[(74, 69), (44, 48), (55, 12), (79, 4)]
$C_1^5(u)$	[(26, 46), (44, 29), (48, 16), (39, 8), (26, 11), (18, 22), (16, 31), (4, 30)]
$C_2^5(v)$	[(37, 47), (33, 35), (38, 23), (51, 10), (72, 9), (85, 15), (87, 26), (84, 47)]

对图 2 所示的 2 种相切情况, 在不同容差下对 3 种迭代算法进行实验, 结果如表 5 所示. 可以看

出, M1 的收敛速度明显下降, 在初始点不理想的情况下还会发生发散的情况; M2 的收敛速度最快, 但由于每次迭代计算代价较大, 因此整体速度缓慢; M3 的收敛速度明显优于 M1, 且每次迭代的计算代价与 M1 保持在同一数量级, 在整体耗时上最少.

在不同容差下, 将本文算法与 OCCT 和 ACIS 时间效率进行实验, 结果如表 6 所示. 可以看出, 在精度一致的情况下, 本文算法的效率显著高于 OCCT, 平均性能可达 OCCT 的 50 倍, 比 ACIS 高 40%. 本文算法保持稳定的领先水平.

表 5 相切情况下不同容差时 3 种迭代算法对比

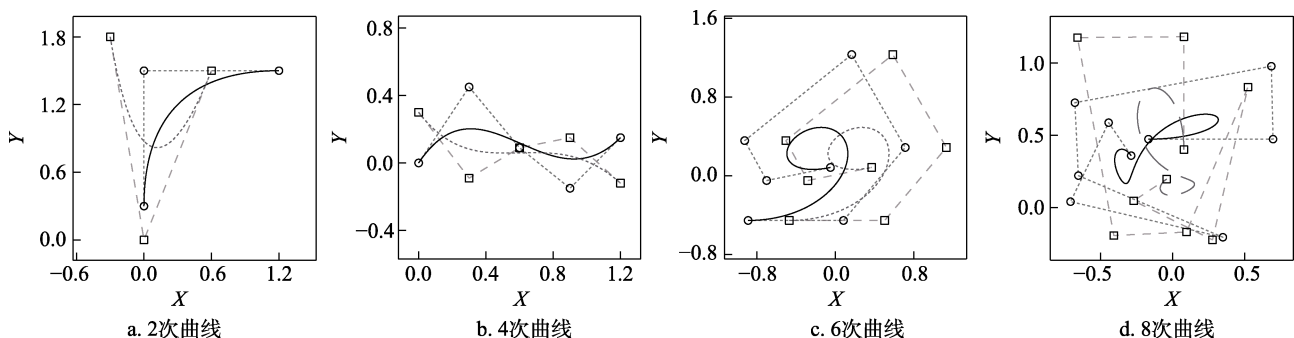
实例	方法	K					T_{avg}/ms	T_{tot}/ms	
		10^{-3}	10^{-6}	10^{-9}	10^{-12}	10^{-15}		10^{-6}	10^{-15}
图 2a	M1	5	7	8	13	15	0.007 87	0.055	0.118
	M2	1	2	4	7	8	0.021 33	0.042	0.171
	M3	3	4	7	10	11	0.006 68	0.026	0.073
图 2b	M1	5	12	14	17	19	0.008 29	0.099	0.157
	M2	4	5	7	9	11	0.037 33	0.186	0.410
	M3	3	6	10	13	15	0.007 10	0.042	0.106

例 3. 如图 3 所示, 分别构建了 2, 4, 6, 8 次曲线求交实例; 并在上述基础上固定第 1 条曲线, 对第 2 条曲线进行平移或旋转, 共生成基准测试实例

4 000 例, 保证覆盖有交、无交、相切等极限情况. 实验结果如表 7 所示, 其中, 容差为 10^{-6} . 可以看出, 本文算法平均性能比 ACIS 高 76%.

表 6 相切情况下不同容差时 3 种求交算法时间效率对比

实例	算法	10^{-6}		10^{-9}		10^{-12}	
		时间/ms	比例	时间/ms	比例	时间/ms	比例
图 2a	OCCT	0.592	6.43	1.173	15.00	2.224	15.20
	ACIS	0.137	1.48	0.128	1.64	0.183	1.25
	本文	0.092	1.00	0.078	1.00	0.146	1.00
图 2b	OCCT	0.931	58.10	1.459	85.80	1.796	119.00
	ACIS	0.019	1.18	0.024	1.41	0.019	1.26
	本文	0.016	1.00	0.017	1.00	0.015	1.00



—— 曲线1; - - - 曲线2; - - - 控制多边形

图 3 基准测试实例

表 7 图 3 基准测试实例效率对比

算法	2 次曲线		4 次曲线		6 次曲线		8 次曲线	
	时间/ms	比例	时间/ms	比例	时间/ms	比例	时间/ms	比例
ACIS	0.087	2.80	0.102	1.59	0.175	1.45	0.304	1.21
本文	0.031	1.00	0.064	1.00	0.120	1.00	0.251	1.00

4 结 语

针对 NURBS 曲线求交问题, 本文提出一种基于有理函数插值的迭代算法, 将细分法与有理插值迭代结合, 在保证算法鲁棒性的同时显著地提升计算效率. 理论分析与数值算例结果表明: (1) 本文的有理插值公式通过双点导数约束有效地提升逼近精度, 比传统牛顿法减少了对初始值的敏感性依赖; (2) 在相切情况下, 基于有理函数插值的迭代策略的效率可以提升 2 倍以上, 且单次迭代耗时而与经典牛顿法保持在同一量级; (3) 在工程应用层面, 本文算法的整体效率是 OCCT 几何内核的 5~50 倍, 比 ACIS 几何内核高 21%~180%. 为 2 条 NURBS 曲线的实时求交问题提供了新的解决方案.

本文算法中, 1 个初始值只能迭代收敛到 1 个

根; 当 2 条曲线拥有多个交点时, 需要考虑根隔离等方法为每个交点获取相应的初始值. 未来, 将考虑多交点情形下初始点的高效计算. 此外, 将本文算法和思路推广到 B 样条曲线/B 样条曲面, 以及 B 样条曲面间的求交问题中, 也是今后可能的工作之一.

参考文献(References):

- [1] Dokken T. Aspects of intersection algorithms and approximation[D]. Oslo: University of Oslo, 2000
- [2] Mørken K, Reimers M, Schulz C. Computing intersections of planar spline curves using knot insertion[J]. Computer Aided Geometric Design, 2009, 26(3): 351-366
- [3] Ao X F, Fu Q, Wu Z K, et al. An intersection algorithm for disk B-spline curves[J]. Computers & Graphics, 2018, 70: 99-107

- [4] Lou Q, Liu L G. Curve intersection using hybrid clipping[J]. *Computers & Graphics*, 2012, 36(5): 309-320
- [5] Abbasbandy S. Improving Newton-Raphson method for nonlinear equations by modified Adomian decomposition method[J]. *Applied Mathematics and Computation*, 2003, 145(2-3): 887-893
- [6] An H B, Bai Z Z. Directional secant method for nonlinear equations[J]. *Journal of Computational and Applied Mathematics*, 2005, 175(2): 291-304
- [7] Sederberg T W, Nishita T. Curve intersection using Bézier clipping[J]. *Computer-Aided Design*, 1990, 22(9): 538-549
- [8] Marco A, Martinez J J. Using polynomial interpolation for implicitizing algebraic curves[J]. *Computer Aided Geometric Design*, 2001, 18(4): 309-319
- [9] Srijuntongsiri G. Finding all intersections between planar curves[C] //Proceedings of the ECTI-CON2010: The 2010 ECTI International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology. Los Alamitos: IEEE Computer Society Press, 2010: 1241-1244
- [10] Sederberg T W, Parry S R. Comparison of three curve intersection algorithms[J]. *Computer-Aided Design*, 1986, 18(1): 58-63
- [11] Kim D S, Lee S W, Shin H. A cocktail algorithm for planar Bézier curve intersections[J]. *Computer-Aided Design*, 1998, 30(13): 1047-1051
- [12] Liu Y H, Cao J, Chen Z G, *et al.* Ray-triangular Bézier patch intersection using hybrid clipping algorithm[J]. *Frontiers of Information Technology & Electronic Engineering*, 2016, 17(10): 1018-1030
- [13] Srijuntongsiri G. An iterative/subdivision hybrid algorithm for curve/curve intersection[J]. *The Visual Computer*, 2011, 27(5): 365-371
- [14] Christou D, Danelakis A, Mitrouli M, *et al.* A hybrid method for computing the intersection and tangency points of plane curves[J]. *Applied Mathematics and Computation*, 2012, 219(4): 1975-1987
- [15] Wang Yongao, Lyu Hangting, Chen Xiaodiao. An efficient method for computing the intersection between two B-spline curves[J]. *Journal of Computer-Aided Design & Computer Graphics*, 2024, 36(5): 687-700(in Chinese)
(王永澳, 吕杭汀, 陈小雕. 两条B样条曲线求交的高效计算方法[J]. *计算机辅助设计与图形学学报*, 2024, 36(5): 687-700)
- [16] Kantorovich L V. On Newton's method for functional equations[J]. *Doklady Akademii Nauk SSSR*, 1948, 59(6): 1237-1240
- [17] Yap C K. Complete subdivision algorithms, I: intersection of Bezier curves[C] //Proceedings of the 22nd Annual Symposium on Computational Geometry. New York: ACM Press, 2006: 217-226
- [18] Berrut J P, Baltensperger R, Mittelmann H D. Recent developments in barycentric rational interpolation[M] //Mache D H, Szabados J, Bruin M G. *Trends and Applications in Constructive Approximation*. Basel: Birkhäuser, 2005: 27-51
- [19] Sharma R, Sharma J R, Kalra N. A modified Newton-Özban composition for solving nonlinear systems[J]. *International Journal of Computational Methods*, 2020, 17(8): Article No.1950047