Computers & Graphics 36 (2012) 309–320

Contents lists available at SciVerse ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

SMI 2012: Full Paper

Curve intersection using hybrid clipping ☆

Qi Lou, Ligang Liu *

Department of Mathematics, Zhejiang University, Hangzhou 310027, China

ARTICLE INFO

Article history:
Received 2 December 2011
Received in revised form
15 March 2012
Accepted 17 March 2012
Available online 30 March 2012

Keywords:
Bézier curve
Curve intersection
Bézier clipping
Hybrid clipping

ABSTRACT

This paper presents a novel approach, called *hybrid clipping*, for computing all intersections between two polynomial Bézier curves within a given parametric domain in the plane. Like Bézier clipping, we compute a 'fat line' (a region along a line) to bound one of the curves. Then we compute a 'fat curve' around the optimal low degree approximation curve to the other curve. By clipping the fat curve with the fat line, we obtain a new reduced subdomain enclosing the intersection. The clipping process proceeds iteratively and then a sequence of subdomains that is guaranteed to converge to the corresponding intersection will be obtained. We have proved that the hybrid clipping technique has at least a quadratic convergence rate. Experimental results have been presented to show the performance of the proposed approach with comparison with Bézier clipping.

1. Introduction

Computing intersections of two curves has played an important role in many engineering fields, including computer-aided design and manufacturing (CAD/CAM), collision detection, and geometric modeling [1–3].

If the two curves are parametric, the solution is identified by the parameter values of intersection points. Early approaches include the Bézier subdivision algorithm [4], the interval subdivision method [5], and implicitization [6]. One widely used and robust method is Bézier clipping, which was developed by [7]. It utilizes the convex hull property of Bézier curves and proceeds as clipping away regions of the curves that are guaranteed to not intersect. Recently, [8] proved that Bézier clipping has a quadratic convergence rate.

Bézier clipping can be applied to compute roots of polynomials as well. By extending Bézier clipping, [9] developed the quadratic clipping technique to compute all the roots of a univariate polynomial equation within an interval. The basic idea is to generate a strip bounded by two quadratic polynomials which encloses the original polynomial via degree reduction. Combined with subdivision, quadratic clipping generates a sequence of intervals that contain the roots of the original polynomial. Recently, [10] extended this technique to cubic clipping and more general cases. Theoretical and experimental results have shown that both quadratic clipping and cubic clipping have at least quadratic convergence rates.

In many cases intersection algorithms involve numerical methods for solving systems of bivariate polynomial equations system [11]. Ref. [12] presented an algorithm for computing all roots of a given bivariate polynomial system within a given rectangular domain. They construct the best linear approximants to the polynomials and use them to define planar strips enclosing the roots. The work of [13] used one linear and one quadratic approximants to the two polynomials to improve the convergence rate. Recently, Mourrain and Pavone [14] presented an improvement of Interval Projected Polyhedron (IPP) algorithm [15] for solving a system of polynomials, which uses a reduction strategy based on univariate root finder and Descarte's rule.

Instead of extending Bézier clipping to solve roots of two polynomials, we develop a new technique, called *hybrid clipping*, to directly compute all the intersections between two Bézier curves within a given parametric domain. Like Bézier clipping, we first compute a linear strip to bound one of the curves. Second, we compute the best approximant with lower degree (e.g., quadratic or cubic) of the other curve and then compute a curved strip around this curve. By intersecting the linear strip and the curved strip, we obtain a new reduced subdomain that encloses the intersections. This algorithm is applied iteratively combined with bisection steps and a sequence of subdomains that converge to the corresponding intersections are obtained. We have proved that the convergence rate of the proposed hybrid clipping algorithm is at least quadratic which is as high as Bézier clipping and is in many cases better than Bézier clipping.

This paper is organized as follows. After a brief review on the Bézier clipping algorithm, we describe the hybrid clipping algorithm in Section 2. Section 3 gives the convergence rate theorem

☆ If applicable, supplementary material from the author(s) will be available online after the conference. Please see http://dx.doi.org/10.1016/j.cag.2012.03.021.

* Corresponding author. Tel./fax: +86 571 87953668.
E-mail address: ligangliu@zju.edu.cn (L. Liu).

0097-8493/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.cag.2012.03.021

and its proof. Section 4 provides some experimental results on comparisons with Bézier clipping. We conclude the paper with future work in Section 5.

2. Computing curve intersection via hybrid clipping

In this section, we first review the Bézier clipping algorithm and then introduce our hybrid clipping algorithm for computing the intersections between two polynomial Bézier curves.

2.1. Review on Bézier clipping for curve intersection

Let $\Pi^n_{[\alpha,\beta]}$ be the linear space of planar polynomial Bézier curves of degree n within $[\alpha,\beta]$. Any Bézier curve $\mathbf{f}(t)\in\Pi^n_{[\alpha,\beta]}$ can be represented by its Bernstein–Bézier (BB) representation as

$$\mathbf{f}(t)=\sum_{i=0}^{n}\mathbf{a}_i B^n_{i,[\alpha,\beta]}(t),\quad t\in[\alpha,\beta], \tag{1}$$

where

$$B^n_{i,[\alpha,\beta]}(t)=\binom{n}{i}\frac{(t-\alpha)^i(\beta-t)^{n-i}}{(\beta-\alpha)^n},\quad i=0,1,\ldots,n \tag{2}$$

are the Bernstein basis with degree n in $[\alpha,\beta]$ and $\mathbf{a}_i\in\mathbb{R}^2$ $(i=0,1,\ldots,n)$ are the control points of this curve [2].

Bézier clipping was developed by [7] to compute the intersections between two polynomial Bézier curves $\mathbf{f}(t)$ in (1) and $\mathbf{g}(s)=\sum_{i=0}^{m}\mathbf{b}_i B^m_{i,[\xi,\eta]}(s)$ within the parameter domain $[\alpha,\beta]\times[\xi,\eta]$, which is an iterative method by taking advantages of the convex hull property of Bézier curve. First, it computes a '*fat line*', which is defined as the region between two parallel lines, to bound one of the two curves, say, $\mathbf{f}(t)$. Then it clips away regions of the second curve $\mathbf{g}(s)$, which are guaranteed to not intersect the fat line, by computing the distance function of $\mathbf{g}$ to the fat line. Similarly $\mathbf{f}$ is then clipped by a fat line around $\mathbf{g}$. This intersection algorithm proceeds by iteratively applying the clipping procedure and generates a sequence of parameter subdomains $[\alpha_i,\beta_i]\times[\xi_i,\eta_i]$, $i=1,2,\ldots$, which might contain an intersection of these two curves. The lengths of these intervals decrease to zero and the interval with maximum length which is less than ε is returned to locate the intersection, where ε specifies the desired accuracy.

Bézier clipping performs much better than the classical Newton's method for computing the curve intersections. It does not require a suitable initial guess and is guaranteed to find all intersections in a given domain. And it converges more robustly than Newton's method. Recently, [8] proved that Bézier clipping has a quadratic convergence rate at transversal intersection, the equivalent of single root of function. In the case of tangent intersections, the equivalent of multiple root of function, however, only linear convergence was observed.

2.2. Hybrid clipping algorithm for curve intersection

As we see, Bézier clipping needs to compute the intersections between a line and a polynomial Bézier curve in each of its iteration, which can be converted into a root-finding problem for polynomial equation. Recently, the works of [9,10] proposed efficient approaches, which extended Bézier clipping, for computing the roots of a polynomial. The basic idea is to use a polynomial with low degree, say 2 or 3, to approximate the original polynomial based on degree reduction. Then two polynomials with low degrees are obtained to bound the original polynomial and their roots bound the corresponding roots of the original polynomial. These approaches are pretty efficient due to the fact that the roots for quadratic or cubic polynomials can be computed analytically. As shown in [9,10], these clipping approaches based on degree reduction have higher convergence rates than Bézier clipping, which make the intersection computation much faster.

Inspired from these works, we propose a new approach for computing the intersections between two polynomial Bézier curves $\mathbf{f}$ and $\mathbf{g}$ based on degree reduction. As in Bézier clipping, we use a fat line (linear strip) $\mathcal{L}$ to bound one of the curves, say, $\mathbf{g}$. We then compute a curved strip $\mathcal{P}$, called *fat curve*, which is enclosed by two polynomial Bézier curves with low degrees, say, quadratic or cubic, around the other curve $\mathbf{f}$. By intersecting the two strips $\mathcal{L}$ and $\mathcal{P}$, which can efficiently be computed in closed forms, we can compute a reduced subdomain that includes the intersection, see Fig. 1. As we use both a linear strip and a curved strip to bound the intersection, we call this approach *hybrid clipping*.

The hybrid clipping algorithm for computing the intersections between two polynomial Bézier curves is given in Table 1. We will

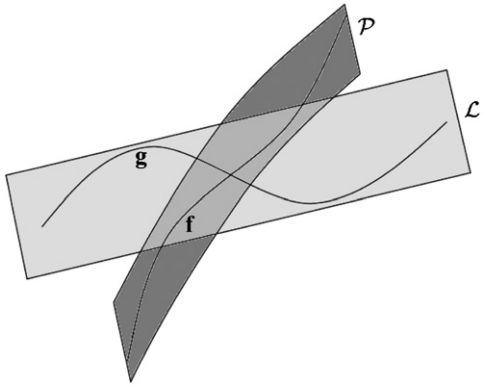

Fig. 1. Illustration of hybrid clipping for intersecting two curves $\mathbf{f}$ and $\mathbf{g}$. The curve $\mathbf{g}$ is bounded by a fat line $\mathcal{L}$ (the linear strip in light gray). A fat curve (the curved strip in dark gray) $\mathcal{P}$ is computed around the curve $\mathbf{f}$. The intersection between $\mathbf{f}$ and $\mathbf{g}$ is bounded by the intersection between two strips $\mathcal{L}$ and $\mathcal{P}$.

Table 1
Hybrid clipping algorithm for computing intersections between two polynomial Bézier curves $\mathbf{f}$ and $\mathbf{g}$ within the domain $[\alpha,\beta]\times[\xi,\eta]$.

Algorithm 1 `HybridClip` ($\mathbf{f},\mathbf{g},[\alpha,\beta],[\xi,\eta],k$) {Hybrid clipping}

```
if max(β−α,η−ξ) ≥ ε then
    if β−α ≥ η−ξ then
        L ← compute the fat line of g
        P ← compute the fat curve of f
        if the fat curve P doesn't intersect the fat line L within [α,β] then
            return (∅)
        else
            Find intervals [α_i,β_i], i = 1,...,l, by intersecting P with L
            if max_{i=1,...,l} |α_i−β_i| ≥ ½|α−β| then
                return HybridClip (f,g, [α, ½(α+β)],[ξ, ½(ξ+η)],k)
                                ∪ HybridClip (f,g, [α, ½(α+β)],[½(ξ+η),η],k)
                                ∪ HybridClip (f,g, [½(α+β),β],[ξ, ½(ξ+η)],k)
                                ∪ HybridClip (f,g, [½(α+β),β],[½(ξ+η),η],k)
            else
                S ← ∅
                for i = 1,...,l do
                    S ← S ∪ HybridClip(f,g,[α_i,β_i],[ξ,η],k)
                end for
                return (S)
            end if
        end if
    else
        return HybridClip (g,f,[ξ,η],[α,β],k)
else
    return ([α,β],[ξ,η])
end if
```

give detailed explanation about this algorithm in the following sections.

2.3. Fat curve construction via degree reduction

The fat line of $\mathbf{g}(s)$ is constructed as in Bézier clipping [7], which is constructed by two lines that are parallel to the chord of $\mathbf{g}$, see Fig. 2. We now introduce the construction of fat curve for $\mathbf{f}(t)$ based on degree reduction.

Denote $\|\cdot\|$ as the canonical Euclidean norm of $\mathbf{f}(t)\in\mathbb{R}^2$. We define three other norms in $\Pi^n_{[\alpha,\beta]}$ as follows:

1. normalized L_2 norm:

$$\|\mathbf{f}\|_2^{[\alpha,\beta]}=\frac{1}{\sqrt{\beta-\alpha}}\left(\int_\alpha^\beta\|\mathbf{f}(t)\|^2\,\mathrm{d}t\right)^{1/2}, \tag{3}$$

2. L_∞ norm:

$$\|\mathbf{f}\|_\infty^{[\alpha,\beta]}=\max_{t\in[\alpha,\beta]}\|\mathbf{f}(t)\|, \tag{4}$$

3. BB norm:

$$\|\mathbf{f}\|_{BB}^{[\alpha,\beta]}=\max_{i=0,\ldots,n}\|\mathbf{a}_i\|. \tag{5}$$

It is easy to verify that the above norms are invariant under affine transformations of the t-axis [9]. Specifically, each of the three norms can be regarded as a functional $F^{[\alpha,\beta]}$ defined in $\Pi^n_{[\alpha,\beta]}$. Then given any affine transformation

$$\mathcal{A}: t\mapsto A_0+A_1t, \tag{6}$$

with $A_1\neq 0$, the following equation holds:

$$F^{[\alpha,\beta]}(\mathbf{f})=F^{\mathcal{A}([\alpha,\beta])}(\mathbf{f}\circ\mathcal{A}^{-1}). \tag{7}$$

Let $\mathbf{p}(t)\in\Pi^k_{[\alpha,\beta]}$ with $k<n$ be the optimal L_2 approximation to $\mathbf{f}(t)\in\Pi^n_{[\alpha,\beta]}$. As shown in Appendix A, $\mathbf{p}(t)$ can be obtained using the degree reduction technique, which can easily be computed by matrix computation. We represent $\mathbf{p}(t)$ as a degree n curve $\mathbf{p}(t)=\sum_{i=0}^n\mathbf{c}_iB^n_{i,[\alpha,\beta]}\in\Pi^n_{[\alpha,\beta]}$ by degree elevation (see Appendix B) and denote

$$\delta=\|\mathbf{f}(t)-\mathbf{p}(t)\|_{BB}^{[\alpha,\beta]}. \tag{8}$$

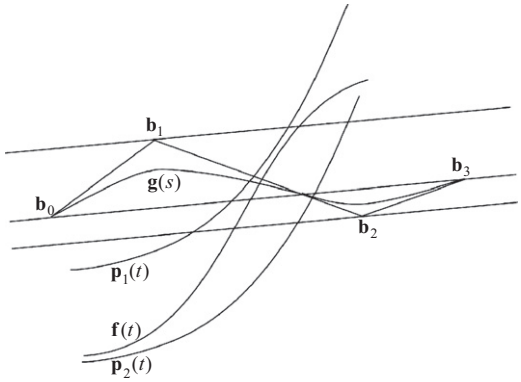

Fig. 2. Illustration of fat line and fat curve. The fat line of $\mathbf{g}(s)$ is bounded by two lines that are parallel to the chord of its control polygon. The fat curve of $\mathbf{f}(t)$ is bounded by two curves $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ with lower degrees.

Thus we have

$$\|\mathbf{f}(t)-\mathbf{p}(t)\|=\left\|\sum_{i=0}^n(\mathbf{a}_i-\mathbf{c}_i)B^n_{i,[\alpha,\beta]}(t)\right\|\le\sum_{i=0}^n\|\mathbf{a}_i-\mathbf{c}_i\|B^n_{i,[\alpha,\beta]}(t)\le\delta. \tag{9}$$

We offset $\mathbf{p}(t)$ along a unit vector $\mathbf{n}(t)$ by the distance δ and get two curves as

$$\mathbf{p}_1(t)=\mathbf{p}(t)+\delta\mathbf{n}(t),\quad \mathbf{p}_2(t)=\mathbf{p}(t)-\delta\mathbf{n}(t). \tag{10}$$

It can be seen by the definition of BB norm that $\mathbf{f}(t)$ lies between $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ on the line segment. That is, $\mathbf{f}(t)$ is 'bounded' by the two curves $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ along the direction of $\mathbf{n}(t)$ at each t. We call $\mathbf{p}_1(t)$ and $\mathbf{p}_2(t)$ the upper bound and the lower bound of $\mathbf{f}(t)$ along $\mathbf{n}$ respectively.

The unit vector $\mathbf{n}$ in (10) should be carefully chosen in order to ease the analysis of convergence rate of our proposed hybrid clipping algorithm. In our algorithm, we set $\mathbf{n}$ as the unit vector that is perpendicular to the chord $(\mathbf{b}_m-\mathbf{b}_0)$ of $\mathbf{g}(s)$, see Fig. 2. Denote

$$d_{\max}=\max_{i=0,\ldots,m}(\mathbf{n}\cdot(\mathbf{b}_i-\mathbf{b}_0)),$$

$$d_{\min}=\min_{i=0,\ldots,m}(\mathbf{n}\cdot(\mathbf{b}_i-\mathbf{b}_0)), \tag{11}$$

$$d(t)=\mathbf{n}\cdot(\mathbf{f}(t)-\mathbf{b}_0),\quad d_0(t)=\mathbf{n}\cdot(\mathbf{p}(t)-\mathbf{b}_0), \tag{12}$$

$$d_1(t)=\mathbf{n}\cdot(\mathbf{p}_1(t)-\mathbf{b}_0)=d_0(t)+\delta,$$

$$d_2(t)=\mathbf{n}\cdot(\mathbf{p}_2(t)-\mathbf{b}_0)=d_0(t)-\delta. \tag{13}$$

Then we have

$$\begin{aligned}|d(t)-d_0(t)|=|\mathbf{n}\cdot(\mathbf{f}(t)-\mathbf{p}(t))|&\le\|\mathbf{n}\|\cdot\|\mathbf{f}(t)-\mathbf{p}(t)\|=\|\mathbf{f}(t)-\mathbf{p}(t)\|\\&\le\|\mathbf{f}(t)-\mathbf{p}(t)\|_\infty^{[\alpha,\beta]}\le\delta,\end{aligned} \tag{14}$$

which means that $d(t)$ is enclosed in the strip bounded by $d_1(t)$ and $d_2(t)$, see Fig. 3. We summarize the algorithm for computing the fat curve $\mathcal{P}$ of $\mathbf{f}$ in Table 2.

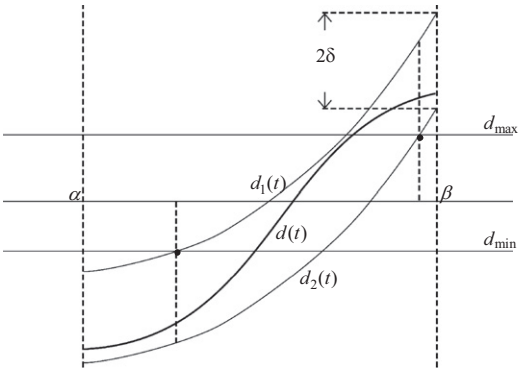

Fig. 3. Computation of the subintervals which bounds the intersections.

Table 2
Compute the fat curve of $\mathbf{f}$ via degree reduction.

Algorithm 2	`ComputeFatCurve` $(\mathbf{f},\mathbf{g},k)$	{Fat curve}

$\mathbf{p}\leftarrow$ generate a degree k curve of $\mathbf{f}$ by degree reduction
$\delta\leftarrow$ compute $\|\mathbf{f}-\mathbf{p}\|_{BB}^{[\alpha,\beta]}$
$\mathbf{n}\leftarrow$ compute a unit vector that is orthogonal to $(\mathbf{b}_m-\mathbf{b}_0)$ of $\mathbf{g}(s)$
$\mathbf{p}_1\leftarrow\mathbf{p}+\delta\mathbf{n}$ {upper bound}
$\mathbf{p}_2\leftarrow\mathbf{p}-\delta\mathbf{n}$ {lower bound}
return $\mathcal{P}=(\mathbf{p}_1,\mathbf{p}_2)$

2.4. Remarks on the algorithms

Some steps of Algorithms 1 and 2 will be explained in more detail in the following:

- In line 4 of Algorithm 1, the fat curve $\mathcal{P}$ of $\mathbf{f}$ is computed by Algorithm 2. Note that the fat curve $\mathcal{P}$ only bounds $\mathbf{f}$ along some direction at each point. The region of $\mathcal{P}$ does not necessarily enclose the curve of $\mathbf{f}$.
- In line 1 of Algorithm 2, when we utilize Eq. (A.1) to compute the best approximation of degree k, all the matrices involved can be precomputed and stored in a lookup-table to save time. Similarly, we precompute and store the matrices for degree elevation given in Eq. (B.1).
- In line 8 of Algorithm 1, after discarding those intervals that satisfy $d_1(t) < d_{\min}$ or $d_2(t) > d_{\max}$, we obtain a series of closed intervals $[\alpha_i,\beta_i]$, $i=1,\ldots,l$, that bound the intersections, see Fig. 3.
- It is obvious that $k < \min(n,m)$. Practically we choose $k=2$ or 3 as the roots of quadratic and cubic polynomials can be analytically calculated [10].
- In line 3 of Algorithm 1, we compute the fat line as in [7]. Actually, any pair of parallel lines that bound the curve can serve as a fat line. Other fat line construction methods were discussed in [8].
- In our practical implementation, we always use the fat line of the curve with smaller interval to clip the fat curve of the other curve, which will make the algorithm more efficiently, see line 20 of Algorithm 1.

`HybridClip`($\mathbf{f},\mathbf{g},[\alpha,\beta],[\xi,\eta],k$) will return some pairs of intervals containing all the intersections of $\mathbf{f}$ and $\mathbf{g}$ within $[\alpha,\beta]\times[\xi,\eta]$ given a prescribed tolerance ε. This technique guarantees to find all the intersections of two planar Bézier curves within a given domain. However, like Bézier clipping, the approach may produce false positive answers (i.e., some pairs of intervals do not contain any intersection) if the two curves get too close to each other.

3. Convergence rate of hybrid clipping

We give the convergence rate of the proposed hybrid clipping algorithm in this section.

Definition 3.1. Suppose that $\mathbf{f}(t)\in\Pi^n_{[\alpha,\beta]}$ and $\mathbf{g}(s)\in\Pi^m_{[\xi,\eta]}$ intersect at $\mathbf{z}_0=\mathbf{f}(t_0)=\mathbf{g}(s_0)$, where $t_0\in[\alpha,\beta]$ and $s_0\in[\xi,\eta]$. The intersection $\mathbf{z}_0$ is called a transversal intersection if $\mathbf{f}'(t_0)\times\mathbf{g}'(s_0)\neq\mathbf{0}$; $\mathbf{z}_0$ is

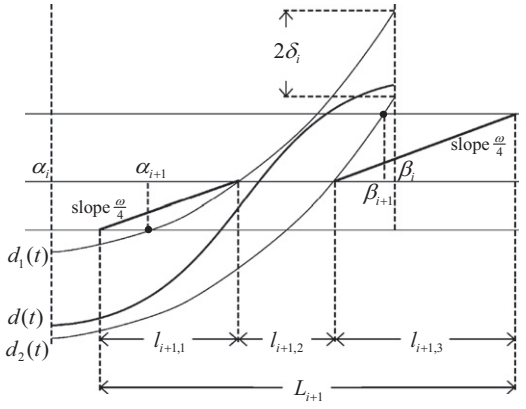

Fig. 4. Proof of Eq. (15).

Table 3
Example 1 (transversal intersections): number pairs of iterations [N_f,N_g] and computing times t in seconds for various values of accuracy ε. (n,m) are degrees of the two curves $\mathbf{f}$,$\mathbf{g}$, respectively.

Degrees (n,m)	ε	10^{-4}			10^{-8}			10^{-16}			10^{-32}			10^{-64}		
		2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip
(4,4)	$[N_f,N_g]$	[2,2]	[2,2]	[2,2]	[3,3]	[2,2]	[3,3]	[3,3]	[3,3]	[4,4]	[4,4]	[3,4]	[5,5]	[4,5]	[4,4]	[6,6]
	t	0.327	0.499	0.374	0.514	0.499	0.561	0.514	0.748	0.764	0.702	0.858	0.920	0.748	0.998	1.092
(8,4)	$[N_f,N_g]$	[1,8]	[1,8]	[1,8]	[2,8]	[1,8]	[2,9]	[2,9]	[2,9]	[3,10]	[3,10]	[3,9]	[4,11]	[4,10]	[3,10]	[5,12]
	t	8.580	11.965	9.250	8.704	12.136	9.516	8.751	12.480	9.843	9.157	12.604	10.218	9.266	12.760	10.514
(8,8)	$[N_f,N_g]$	[2,3]	[2,2]	[3,3]	[3,3]	[2,3]	[4,4]	[3,4]	[3,3]	[4,5]	[4,4]	[3,4]	[5,6]	[4,5]	[4,4]	[6,7]
	t	0.858	0.873	1.170	1.060	1.076	1.575	1.248	1.263	1.747	1.435	1.528	2.168	1.606	1.747	2.574

called a tangent intersection if $\mathbf{f}'(t_0)\times\mathbf{g}'(s_0)=\mathbf{0}$ and $\mathbf{f}'(t_0)\neq\mathbf{0}$, $\mathbf{g}'(s_0)\neq\mathbf{0}$; $\mathbf{z}_0$ is called a degenerate intersection if one of the curves has no tangent at $\mathbf{z}_0$, i.e., $\mathbf{f}'(t_0)=\mathbf{0}$ or $\mathbf{g}'(s_0)=\mathbf{0}$.

The following theorem gives the convergence rates of the hybrid clipping approach for transversal intersections.

Theorem 3.2. *Suppose $\mathbf{f}(t)\in\Pi^n_{[\alpha,\beta]}$ and $\mathbf{g}(s)\in\Pi^m_{[\xi,\eta]}$ have a transversal intersection $\mathbf{z}_0=\mathbf{f}(t_0)=\mathbf{g}(s_0)$. Furthermore, suppose $([\alpha_i,\beta_i])_{i=0,1,2,\ldots}$ is the sequence of generated intervals that contain t_0, and $([\xi_i,\eta_i])_{i=0,1,2,\ldots}$ is the corresponding sequence of generated intervals that contain s_0, then there exist constants $C_{\mathbf{f}}$ depending solely on $\mathbf{f}$ and $C_{\mathbf{g}}$ depending solely on $\mathbf{g}$, such that for sufficiently large $i\in\mathbb{N}$, the following inequality holds*:

$$h_{i+1,\mathbf{f}}\le C_{\mathbf{f}}h_{i,\mathbf{f}}^{k+1}+C_{\mathbf{g}}h_{i,\mathbf{g}}^{2}, \tag{15}$$

where $h_{i,\mathbf{f}}=\beta_i-\alpha_i$ and $h_{i,\mathbf{g}}=\eta_i-\xi_i$.

Similarly, there exist constants $C'_{\mathbf{f}}$ depending solely on $\mathbf{f}$ and $C'_{\mathbf{g}}$ depending solely on $\mathbf{g}$, such that for sufficiently large $i\in\mathbb{N}$, the following inequality holds:

$$h_{i+1,\mathbf{g}}\le C'_{\mathbf{f}}h_{i,\mathbf{f}}^{2}+C'_{\mathbf{g}}h_{i,\mathbf{g}}^{k+1}. \tag{16}$$

We will prove this convergence rate theorem based on two technical lemmas which respectively can be regarded as the 'vector versions' of Lemmas 1 and 2 in [10]. In order to make this paper self-contained, we give these two technical lemmas and their proofs first.

Lemma 3.3. *Given a planar Bézier curve $\mathbf{f}$ of degree n, there exists a constant $C_{0\mathbf{f}}$ depending solely on $\mathbf{f}$, such that for all intervals $[\alpha,\beta]\subseteq[0,1]$ the bound δ in (8) satisfies $\delta\le C_{0\mathbf{f}}h^{k+1}$, where $h=\beta-\alpha$.*

Proof. Due to the equivalence of norms in finite-dimensional real linear spaces, there exist constants C_1, C_2 such that

$$\|\mathbf{r}\|_{BB}^{[\alpha,\beta]}\le C_1\|\mathbf{r}\|_2^{[\alpha,\beta]} \quad\text{and}\quad \|\mathbf{r}\|_2^{[\alpha,\beta]}\le C_2\|\mathbf{r}\|_\infty^{[\alpha,\beta]} \tag{17}$$

for all $\mathbf{r}\in\Pi^n_{[\alpha,\beta]}$. The constants C_1 and C_2 do not depend on the given interval $[\alpha,\beta]$, since all the norms are invariant with respect to affine transformations of the t-axis. Therefore

$$\begin{aligned}\delta=\|\mathbf{f}-\mathbf{p}\|_{BB}^{[\alpha,\beta]}&\le C_1\|\mathbf{f}-\mathbf{p}\|_2^{[\alpha,\beta]}\le C_1\|\mathbf{f}-\mathbf{q}_\alpha\|_2^{[\alpha,\beta]}\le C_1C_2\|\mathbf{f}-\mathbf{q}_\alpha\|_\infty^{[\alpha,\beta]}\\&\le\frac{\sqrt{2}}{(k+1)!}C_1C_2\max_{t\in[0,1]}\|\mathbf{f}^{(k+1)}(t_0)\|h^{k+1},\end{aligned} \tag{18}$$

where each of the components of $\mathbf{q}_\alpha$ is the degree k Taylor polynomials at $t=\alpha$ to the corresponding component of $\mathbf{f}$ and $\mathbf{f}^{(k+1)}$ is the $(k+1)$-th derivative vector. □

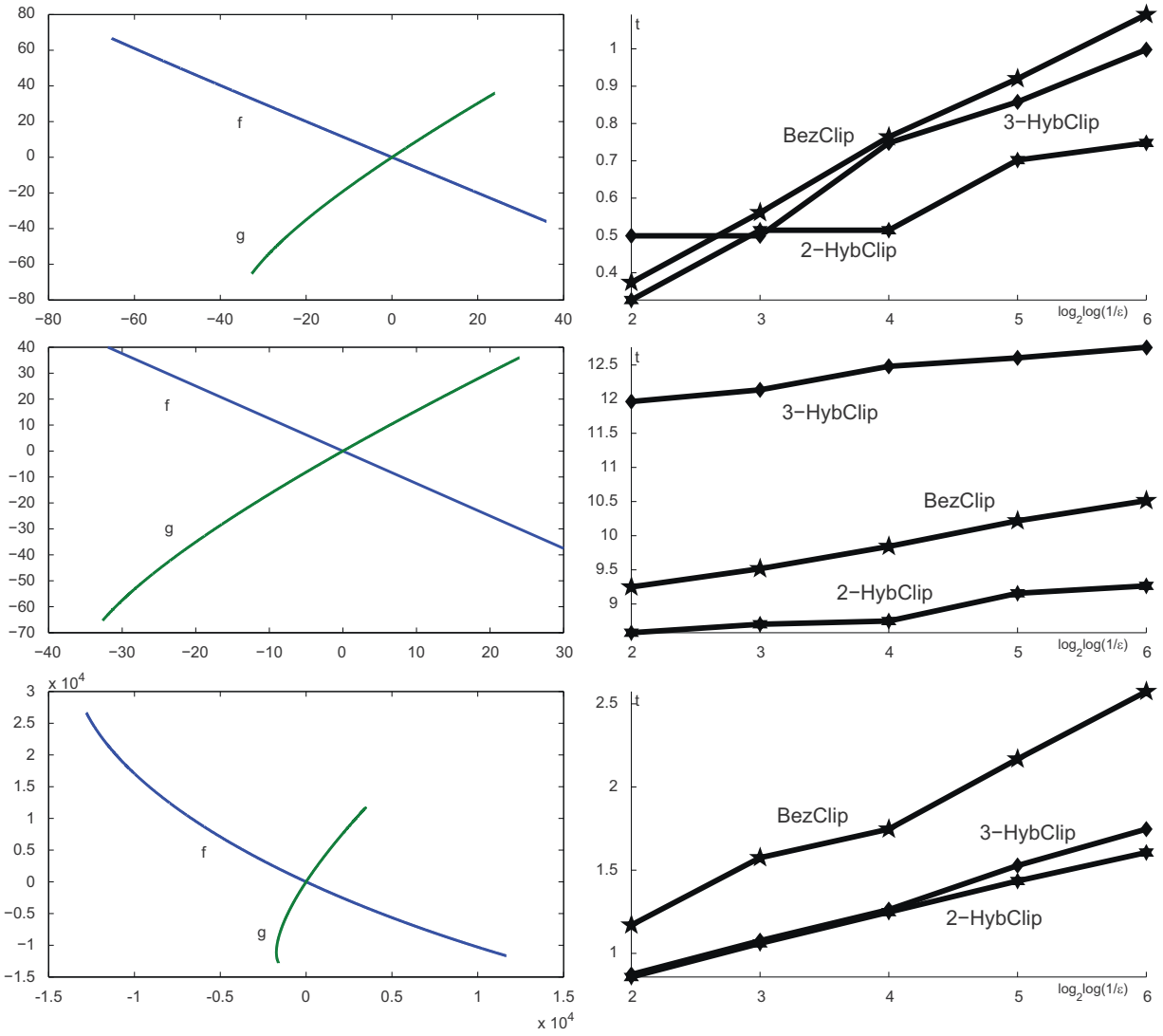

Fig. 5. Example 1 (transversal intersections). Left: the figures of the curve pairs; right: computing time t in seconds vs. accuracy. The degrees (n,m) of the two curves are (4,4) (upper row), (8,4) (middle row), (8,8) (lower row), respectively.

Lemma 3.4. *Given a planar Bézier curve* $\mathbf{f}$ *of degree* n*, there exist* $k+1$ *constants* $C_{1\mathbf{f}},C_{2\mathbf{f}},\ldots,C_{(k+1)\mathbf{f}}$*, depending solely on* $\mathbf{f}$*, such that for all intervals* $[\alpha,\beta]\subseteq[0,1]$ *the optimal approximate curve* $\mathbf{p}$ *of degree* k *to* $\mathbf{f}$ *satisfies*

$$\|\mathbf{f}-\mathbf{p}\|_\infty^{[\alpha,\beta]}\le C_{1\mathbf{f}}h^{k+1},\quad \|\mathbf{f}'-\mathbf{p}'\|_\infty^{[\alpha,\beta]}\le C_{2\mathbf{f}}h^{k},\ldots,$$

$$\|\mathbf{f}^{(k)}-\mathbf{p}^{(k)}\|_\infty^{[\alpha,\beta]}\le C_{(k+1)\mathbf{f}}h, \tag{19}$$

with $h=\beta-\alpha$.

Proof. We construct a new norm in $[\alpha,\beta]$ as

$$\|\mathbf{r}\|_*^{[\alpha,\beta]}=\|\mathbf{r}\|_\infty^{[\alpha,\beta]}+h\|\mathbf{r}'\|_\infty^{[\alpha,\beta]}+\cdots+h^k\|\mathbf{r}^{(k)}\|_\infty^{[\alpha,\beta]}. \tag{20}$$

Due to the affine invariance of the norms, there exists a constant C_3, which does not depend on the interval $[\alpha,\beta]$, such that ([9])

$$\|\mathbf{r}\|_*^{[\alpha,\beta]}\le C_3\|\mathbf{r}\|_2^{[\alpha,\beta]}. \tag{21}$$

Consequently,

$$\begin{aligned}\|\mathbf{f}-\mathbf{p}\|_*^{[\alpha,\beta]}&=\|\mathbf{f}-\mathbf{p}\|_\infty^{[\alpha,\beta]}+h\|\mathbf{f}'-\mathbf{p}'\|_\infty^{[\alpha,\beta]}+\cdots+h^k\|\mathbf{f}^{(k)}-\mathbf{p}^{(k)}\|_\infty^{[\alpha,\beta]}\\&\le C_3\|\mathbf{f}-\mathbf{p}\|_2^{[\alpha,\beta]}\le C_3\|\mathbf{f}-\mathbf{q}_\alpha\|_2^{[\alpha,\beta]}\le C_2C_3\|\mathbf{f}-\mathbf{q}_\alpha\|_\infty^{[\alpha,\beta]}\\&\le\frac{\sqrt{2}}{(k+1)!}C_2C_3\max_{t\in[0,1]}\|\mathbf{f}^{(k+1)}(t_0)\|h^{k+1},\end{aligned} \tag{22}$$

here $\mathbf{q}_\alpha$ has the same meaning as that in the proof of (3.3). Clearly, this implies (19). □

Proof (*Proof of Theorem 3.2*). Note that $\mathbf{f}$ and $\mathbf{g}$ play the same role in the algorithm. We only prove (15) here.

It is observed that the length of intervals $[\alpha_i,\beta_i]$ tends to be 0 as i tends to infinity, since subdivision is introduced in the algorithm (see line 10 of Algorithm 1). The chord $\mathbf{b}_0\mathbf{b}_m$ of $\mathbf{g}$ tends to the tangent line at $\mathbf{z}_0$ when the length of the interval $[\xi_i,\eta_i]$ tends to be 0. Therefore, the unit vector $\mathbf{n}$ (see line 3 of Algorithm 2) tends to the unit normal vector $\mathbf{n}_0$ of $\mathbf{g}$ at $\mathbf{z}_0$.

Denote $\omega=\mathbf{n}_0\cdot\mathbf{f}'(t_0)$. As $\mathbf{f}'(t_0)\times\mathbf{g}'(s_0)\neq\mathbf{0}$, then $\omega\neq0$. Without loss of generality, we assume $\omega>0$ (otherwise we can take $-\mathbf{n}$ instead of $\mathbf{n}$ and $-\mathbf{n}_0$ instead of $\mathbf{n}_0$).

Since $\mathbf{n}_0$ is the limit of $\mathbf{n}$, there exists $\varepsilon_1>0$ such that

$$|d'(t_0)-\omega|=|\mathbf{n}\cdot\mathbf{f}'(t_0)-\mathbf{n}_0\cdot\mathbf{f}'(t_0)|<\omega/4 \tag{23}$$

as $h_{i,\mathbf{g}}<\varepsilon_1$.

Due to the continuity of $\mathbf{f}'(t)$, there exists $\varepsilon_2>0$ such that

$$\|\mathbf{f}'(t)-\mathbf{f}'(t_0)\|<\omega/4 \tag{24}$$

as $h_{i,\mathbf{f}}<\varepsilon_2$.

By Lemma 3.4 we have

$$\begin{aligned}|d'(t)-d_1'(t)|&=|\mathbf{n}\cdot(\mathbf{f}'(t)-\mathbf{p}'(t))|\le\|\mathbf{f}'(t)-\mathbf{p}'(t)\|\\&\le\|\mathbf{f}'(t)-\mathbf{p}'(t)\|_\infty^{[\alpha,\beta]}\le C_{2\mathbf{f}}h_{i,\mathbf{f}}^k.\end{aligned} \tag{25}$$

Thus there exists $\varepsilon_3>0$ such that

$$|d'(t)-d_1'(t)|<\omega/4 \tag{26}$$

as $h_{i,\mathbf{f}}<\varepsilon_3$.

Let $\varepsilon_4=\min(\varepsilon_1,\varepsilon_2,\varepsilon_3)$. By (23) and (24) we have

$$\begin{aligned}|d'(t)-\omega|&=|\mathbf{n}\cdot\mathbf{f}'(t)-\mathbf{n}_0\cdot\mathbf{f}'(t_0)|\\&\le|\mathbf{n}\cdot\mathbf{f}'(t)-\mathbf{n}\cdot\mathbf{f}'(t_0)|+|\mathbf{n}\cdot\mathbf{f}'(t_0)-\mathbf{n}_0\cdot\mathbf{f}'(t_0)|\\&\le\|\mathbf{f}'(t)-\mathbf{f}'(t_0)\|+|d'(t_0)-\omega|\\&<\omega/4+\omega/4=\omega/2\end{aligned} \tag{27}$$

as $h_{i,\mathbf{g}}<\varepsilon_4$, $h_{i,\mathbf{f}}<\varepsilon_4$. Thus $d'(t)>\omega/2$. By (26) we know

$$d_1'(t)=d_2'(t)>\omega/4. \tag{28}$$

From Fig. 4 we have the bound for $h_{i+1,\mathbf{f}}$ as

$$h_{i+1,\mathbf{f}}\le L_{i+1}=l_{i+1,1}+l_{i+1,2}+l_{i+1,3}. \tag{29}$$

Table 4
Example 2 (tangent intersections): number pairs of iterations $[N_\mathbf{f},N_\mathbf{g}]$ and computing times t in seconds for various values of accuracy ε. (n,m) are degrees of the two curves $\mathbf{f},\mathbf{g}$, respectively.

Degrees (n,m)	ε	10^{-4}			10^{-8}			10^{-16}			10^{-32}			10^{-64}		
		2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip
(4,4)	$[N_\mathbf{f},N_\mathbf{g}]$	[4,8]	[7,5]	[0,14]	[10,13]	[15,8]	[0,27]	[28,22]	[28,18]	[0,54]	[54,43]	[56,38]	[0,107]	[110,79]	[111,73]	[0,213]
	t	4.212	3.182	9.516	7.066	6.770	18.548	11.996	13.930	37.596	23.961	29.203	79.248	46.441	61.058	180.524
(8,4)	$[N_\mathbf{f},N_\mathbf{g}]$	[0,14]	[0,14]	[0,14]	[0,27]	[0,27]	[0,27]	[0,54]	[0,54]	[0,54]	[0,107]	[0,107]	[0,107]	[0,213]	[0,213]	[0,213]
	t	9.656	14.336	10.062	19.250	29.686	20.498	43.227	60.699	43.820	100.293	136.142	95.020	225.327	334.700	243.907
(8,8)	$[N_\mathbf{f},N_\mathbf{g}]$	[7,6]	[5,8]	[10,9]	[15,13]	[16,14]	[16,20]	[31,25]	[28,28]	[34,39]	[61,49]	[58,51]	[73,73]	[117,103]	[113,107]	[151,136]
	t	7.831	9.796	12.495	17.518	21.169	24.819	33.462	39.827	48.781	76.112	86.159	108.685	185.063	212.457	295.543

First it is seen that

$$l_{i+1,1}+l_{i+1,3}=\frac{d_{\max}-d_{\min}}{\omega/4}=\frac{4(d_{\max}-d_{\min})}{\omega}. \tag{30}$$

Now we try to obtain an upper bound for $l_{i+1,2}$.

As $d_1'(t)=d_2'(t)>\omega/4>0$, both functions $d_1(t)$ and $d_2(t)$ strictly increase in the interval $[\alpha_i,\beta_i]$. Therefore, for any y_0 such that $d_1(\alpha_i)<y_0<d_2(\beta_i)$, the following two equations have roots t_1,t_2 within the interval $[\alpha_i,\beta_i]$

$$d_1(t)=y_0,$$

$$d_2(t)=y_0. \tag{31}$$

Obviously

$$l_{i+1,2}\le \sup_{y_0\in(d_1(\alpha_i),d_2(\beta_i))}\{|t_1-t_2|: d_1(t_1)=d_2(t_2)=y_0\}, \tag{32}$$

where $t_1,t_2\in[\alpha_i,\beta_i]$. As $d_1(t_1)=d_2(t_2)=y_0$, we have

$$\mathbf{n}\cdot(\mathbf{p}(t_1)-\mathbf{p}(t_2))=2\delta_i, \tag{33}$$

where δ_i is computed as the corresponding one in (8).

Let $\mathbf{p}(t)=(x(t),y(t))$. There exist t^* and $t^\star$ between t_1 and t_2 such that

$$\begin{aligned}\mathbf{p}(t_1)-\mathbf{p}(t_2)&=(x(t_1)-x(t_2),y(t_1)-y(t_2))\\&=(x'(t^*)(t_1-t_2),y'(t^\star)(t_1-t_2)).\end{aligned} \tag{34}$$

Thus we have

$$\begin{aligned}&|\mathbf{n}\cdot(x'(t^*),y'(t^\star))-d'(t)|\\&=|\mathbf{n}\cdot(x'(t^*),y'(t^\star))-\mathbf{n}\cdot\mathbf{f}'(t)|\le\|(x'(t^*),y'(t^\star))-\mathbf{f}'(t)\|\\&=\|(x'(t^*),y'(t^*))-\mathbf{f}'(t)+(0,y'(t^\star))-(0,y'(t^*))\|\\&\le\|\mathbf{p}'(t^*)-\mathbf{f}'(t)\|+\|\mathbf{p}'(t^\star)-\mathbf{p}'(t^*)\|\\&\le\|\mathbf{p}'(t^*)-\mathbf{p}'(t)\|+\|\mathbf{p}'(t)-\mathbf{f}'(t)\|+\|\mathbf{p}'(t^\star)-\mathbf{p}'(t^*)\|\\&\le 2\max_{t^1,t^2\in[\alpha_i,\beta_i]}\|\mathbf{p}'(t^1)-\mathbf{p}'(t^2)\|+\|\mathbf{p}'(t)-\mathbf{f}'(t)\|_\infty^{[\alpha_i,\beta_i]}.\end{aligned} \tag{35}$$

By Lemma 3.4 and the fact that $\mathbf{p}'(t)$ is uniformly continuous in $[\alpha_i,\beta_i]$, there exists $\varepsilon_5>0$ such that

$$\max_{t^1,t^2\in[\alpha_i,\beta_i]}\|\mathbf{p}'(t^1)-\mathbf{p}'(t^2)\|<\omega/16,$$

$$\|\mathbf{p}'-\mathbf{f}'\|_\infty^{[\alpha_i,\beta_i]}<\omega/8 \tag{36}$$

as $h_{i,\mathbf{f}}<\varepsilon_5$.

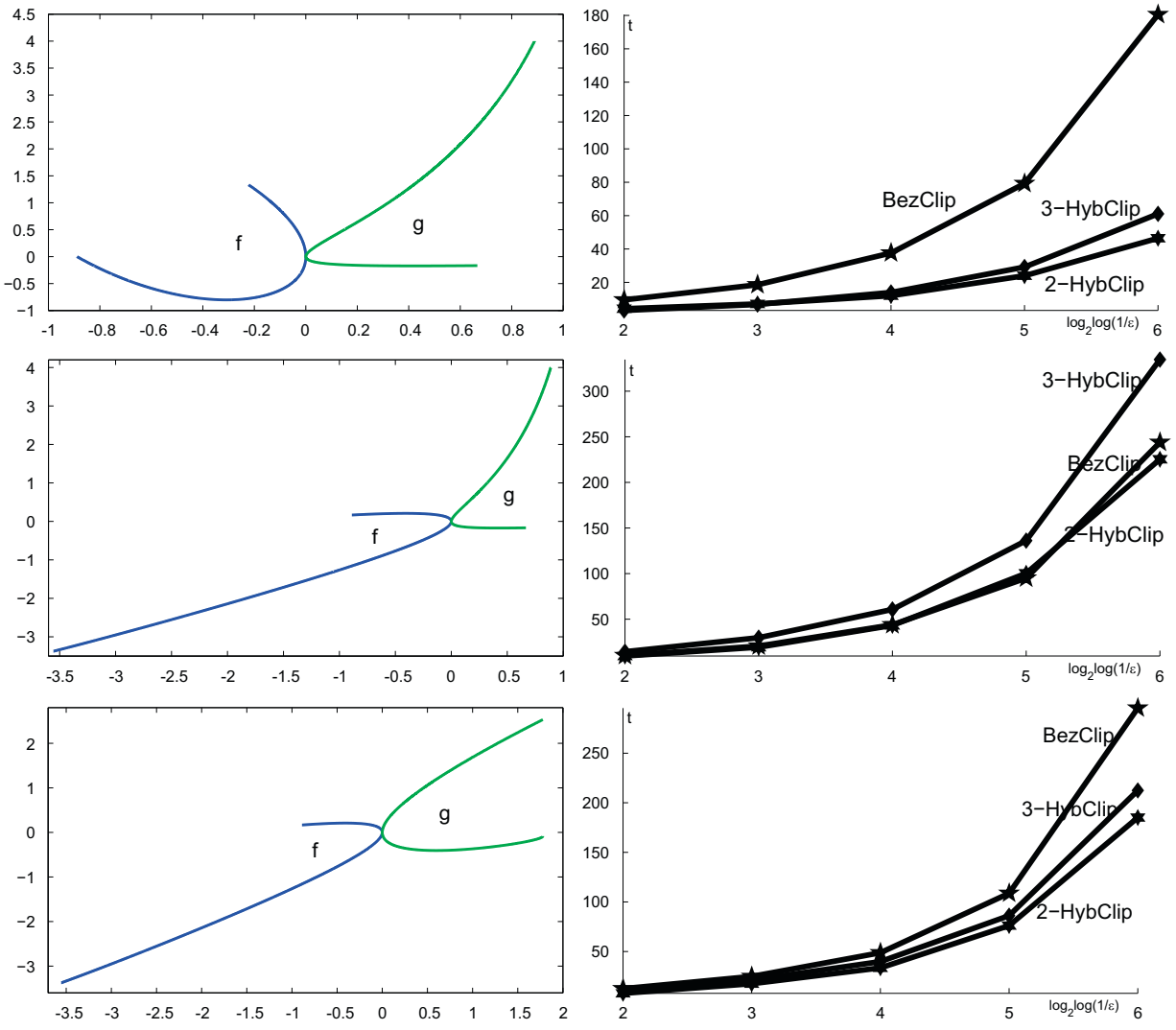

Fig. 6. Example 2 (tangent intersections). Left: the figures of the curve pairs; right: computing time t in seconds vs. accuracy. The degrees (n,m) of the two curves are (4,4) (upper row), (8,4) (middle row), (8,8) (lower row), respectively.

Let $\varepsilon_0 = \min(\varepsilon_4, \varepsilon_5)$. Then we have

$$|\mathbf{n} \cdot (x'(t^*), y'(t^\star)) - d'(t)| < \omega/4 \tag{37}$$

as $h_{i,\mathbf{g}} < \varepsilon_0$ and $h_{i,\mathbf{f}} < \varepsilon_0$.

Combining the above inequalities and (27), we have

$$\mathbf{n} \cdot (x'(t^*), y'(t^\star)) > \omega/4. \tag{38}$$

Since $2\delta_i = \mathbf{n} \cdot (\mathbf{p}(t_1) - \mathbf{p}(t_2)) = (t_1 - t_2)\mathbf{n} \cdot (x'(t^*), y'(t^\star))$, we have

$$|t_1 - t_2| < \frac{2\delta_i}{\omega/4} = \frac{8\delta_i}{\omega}. \tag{39}$$

From (30) and (32), the above inequality implies

$$h_{i+1,\mathbf{f}} \le L_{i+1} < \frac{8\delta_i}{\omega} + \frac{4(d_{\max} - d_{\min})}{\omega}. \tag{40}$$

Clearly, this implies (15) from $\delta_i \le C_{0\mathbf{f}} h_{i,\mathbf{f}}^{k+1}$ (see Lemma 3.3) and $d_{\max} - d_{\min} \le C_{0\mathbf{g}} h_{i,\mathbf{g}}^2$ (see the proof of Theorem 6 of [8]). □

The key of the proof is introducing L_{i+1} to bound $\beta_{i+1} - \alpha_{i+1}$. Meanwhile, L_{i+1} can be decomposed into three components, i.e., $l_{i+1,1}$, $l_{i+1,2}$, and $l_{i+1,3}$, each of which can be bounded separately. On one hand, the bound of $l_{i+1,2}$ relies on the estimation of the difference between the roots of $d_1(t)$ and $d_2(t)$. On the other hand, $l_{i+1,1} + l_{i+1,3}$ can be computed directly. Note that the property that ω is nonzero plays a crucial role in the construction of $l_{i+1,1}$ and $l_{i+1,3}$. That is why transversal intersection is required in this proof.

Due to this theorem, we conclude that the process of subdivision would not exist when intervals are sufficiently small, i.e., hybrid clipping indeed has the convergence rate of at least order 2, which is at least as high a convergence rate as that of Bézier clipping.

Note that we have proved the convergence rates for computing the transversal intersections. But unfortunately we could not obtain the convergence rates for the case of tangent intersection where two curves are tangent at the intersection. As pointed in Bézier clipping [7], a large number of subdivisions may be needed and the algorithm tends to degenerate to a divide-and-conquer binary search for the cases of tangent intersections.

4. Experimental results

In this section, we will do various numerical experiments to illustrate the performance of our proposed hybrid clipping (`HybClip` for short) technique and make comparisons with the classic Bézier clipping (`BezClip` for short) technique.

From Theorem 3.2 we can see that `HybClip` has at least a quadratic convergence rate while `BezClip` has only a quadratic convergence rate [8]. As in [9,10] we introduce a few criteria, such as convergence rate, number of operations per iteration step, time per iteration step, number of iterations, and computing time needed to achieve a certain prescribed accuracy, to make an intensive comparison between these two techniques numerically.

We had done a great number of experiments and the results had shown that `HybClip` performs better than `BezClip` in all these criteria. In order to make our manuscript more compact, here we show the experimental results based on a couple of criteria, that is, number of iterations and computing time needed to achieve a certain prescribed accuracy for convenience, to compare two techniques.

Note that we can use different value of k to construct the fat curve for $\mathbf{f}(t)$ in Algorithm 1. That means we use an optimal

Table 5
Example 3 (degenerate intersections): number pairs of iterations $[N_\mathbf{f}, N_\mathbf{g}]$ and computing times t in seconds for various values of accuracy ε. (n,m) are degrees of the two curves $\mathbf{f}$,$\mathbf{g}$, respectively.

Degrees (n,m)	ε	10^{-4}			10^{-8}			10^{-16}			10^{-32}			10^{-64}		
		2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip	2-HybClip	3-HybClip	BezClip
(4,4)	$[N_\mathbf{f}, N_\mathbf{g}]$	[4,6]	[6,7]	[7,11]	[6,8]	[6,8]	[16,19]	[7,10]	[7,11]	[33,33]	[9,12]	[8,12]	[76,76]	[11,13]	[10,15]	[156,162]
	t	3.588	3.775	4.773	3.931	4.024	6.848	4.180	5.132	11.746	4.570	5.584	27.518	4.804	6.786	53.461
(8,4)	$[N_\mathbf{f}, N_\mathbf{g}]$	[3,8]	[2,8]	[4,12]	[4,10]	[3,9]	[12,21]	[5,11]	[4,12]	[25,36]	[7,13]	[6,14]	[51,68]	[9,15]	[7,15]	[115,126]
	t	4.383	6.708	11.918	4.820	7.035	14.430	5.132	8.782	21.512	5.647	10.374	34.164	6.271	10.810	63.601
(8,8)	$[N_\mathbf{f}, N_\mathbf{g}]$	[4,4]	[4,3]	[9,4]	[6,5]	[5,5]	[17,5]	[8,7]	[7,6]	[34,6]	[11,9]	[8,9]	[68,7]	[13,11]	[9,10]	[135,8]
	t	1.357	1.341	2.511	1.887	1.903	4.305	2.683	2.652	8.346	3.588	5.850	17.160	4.352	6.474	37.658

degree k Bézier curve to approximate $\mathbf{f}(t)$ and then the algorithm involves computing the roots of degree k polynomials. By Theorem 3.2 we know that the larger k is, the faster the convergence rate is. To make the computation practically usable, we generally set k as 2 or 3 because the roots for quadratic or cubic polynomials are easy to compute. We set $k=2,3$ in the following comparisons. And we refer `2-HybClip` and `3-HybClip` as `HybClip` with $k=2$ and $k=3$, respectively, for short.

We have implemented all the algorithms on a PC with Intel(R) Core(TM) Duo CPU (2.00 GHz) with 2 GB of RAM and utilize Maple V12 to do all experiments with 600 significant digits. In these experiments, both curves $\mathbf{f}(t)$ and $\mathbf{g}(s)$ have the parameter domain $[0,1]$.

4.1. Single intersections

We first show experimental results on computing the single intersections between two curves. The cases of transversal intersections are shown as follows.

Example 1 (*Transversal intersection*). We applied the algorithms `2-HybClip`, `3-HybClip`, and `BezClip` to the three pairs of Bézier curves

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)(t-3)(t+6)^2,\ (t-1/3)(t+3)(t-6)^2), \\ \mathbf{g}(s) &= ((s-1/3)(s-2)(s+6)^2,\ (s-1/3)(s-3)(s+6)^2), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)(t-3)^3(t+6)^4,\ (t-1/3)(t+3)^3(t-6)^4), \\ \mathbf{g}(s) &= ((s-1/3)(s-2)(s+6)^2,\ (s-1/3)(s-3)(s+6)^2), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)(t-3)^3(t+6)^4,\ (t-1/3)(t+3)^3(t-6)^4), \\ \mathbf{g}(s) &= ((s-1/3)(s-2)^3(s+6)^4,\ (s-1/3)(s-3)^3(s+6)^4). \end{cases}$$

Table 3 reports the number pairs of iterations $[N_{\mathbf{f}},N_{\mathbf{g}}]$ and the computing times in seconds for various values of desired accuracy ε for computing the transversal intersections between the three pairs of Bézier curves with various degrees (n,m). As pointed out in Section 2.4, we always use the fat line of the curve with smaller interval to clip the fat curve of the other curve in `2-HybClip` and `3-HybClip`, like in `BezClip`. In both algorithms, the number $N_{\mathbf{f}}$ denotes the number of using the fat line of $\mathbf{g}$ to clip $\mathbf{f}$ (in `BezClip`) or the fat curve of $\mathbf{f}$ (in `2-HybClip` or `3-HybClip`). The same for the number $N_{\mathbf{g}}$. Fig. 5 visualizes the relation between computing times and desired accuracy.

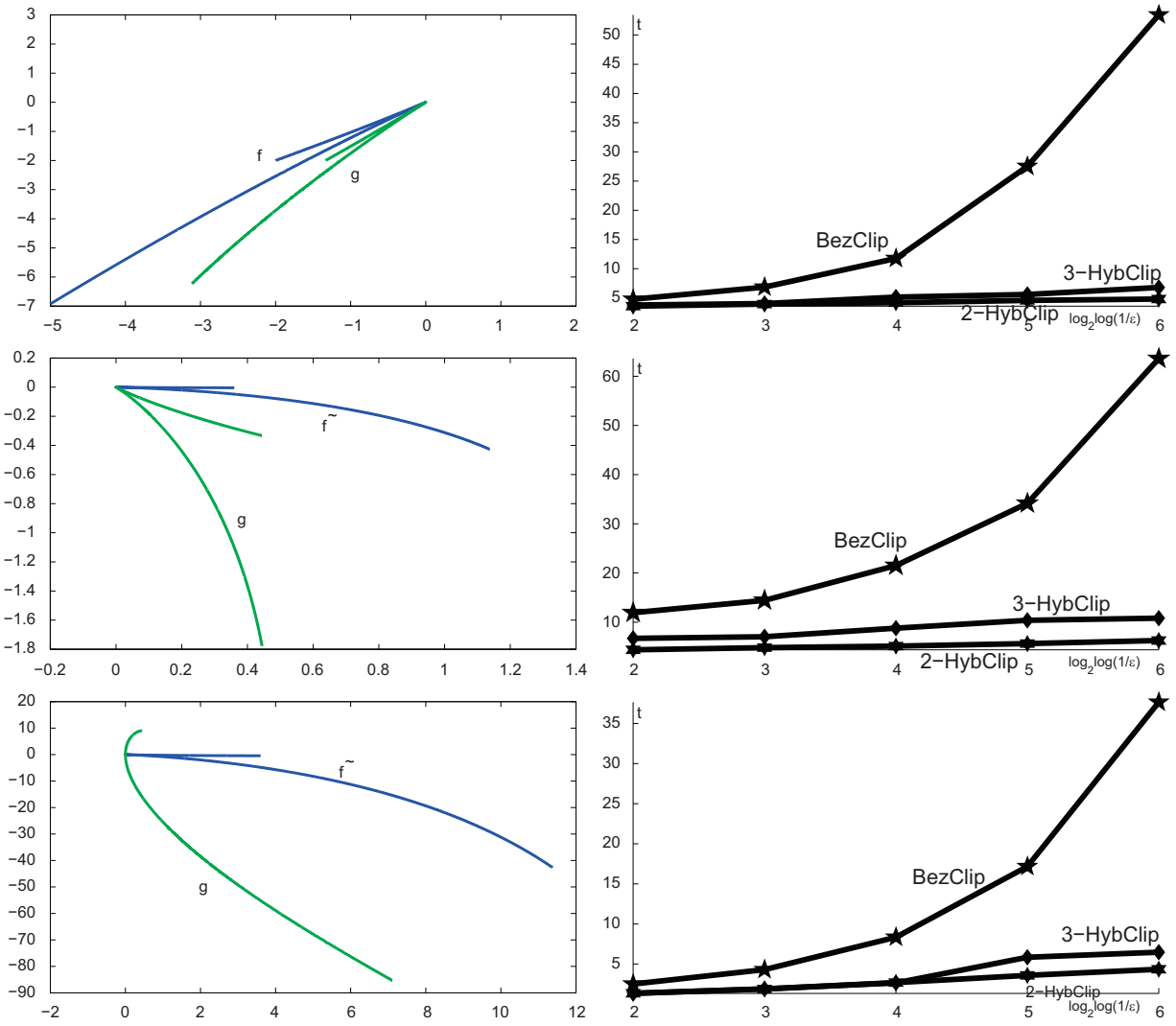

Fig. 7. Example 3 (degenerate intersections). Left: the figures of the curve pairs; right: computing time t in seconds vs. accuracy. The degrees (n,m) of the two curves are (4,4) (upper row), (8,4) (middle row), (8,8) (lower row), respectively.

From Table 3 and Fig. 5, we can see the following observations. Algorithm `2-HybClip` performs slightly better than `BezClip` in running times for these three pairs of Bézier curves. The difference between the overall computing times to achieve a certain accuracy is not that significant although `2-HybClip` has fewer times in all examples. However, Algorithm `3-HybClip` performs not as well as what we have expected. It seems that its performance depends much on the curve pairs. For the third curve pair, `3-HybClip` performs close to `2-HybClip`. Both are better than `BezClip`. For the first curve pair, `3-HybClip` performs only a bit better than `BezClip`. However, for the second curve pair, `3-HybClip` performs even worse than `BezClip`. This is because `3-HybClip` involves more computation in solving the cubic polynomial equations and thus needs more time to finish one iteration step.

We now show experimental results on computing the single tangent intersections between two curves.

Example 2 (*Tangent intersections*). We applied algorithm `2-HybClip`, `3-HybClip`, and `BezClip` to the three pairs of Bézier curves

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)^2(t+1)(t-2),\ (t-1/3)(t-1)(t+2)^2),\\ \mathbf{g}(s) &= ((s-1/3)^2(s-2)(s-3),\ (s-1/3)(s+1/2)(s+1)^2), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)^2(t+1)^3(t-2)^3,\ (t-1/3)(t-2)^3(t+1/2)^4),\\ \mathbf{g}(s) &= ((s-1/3)^2(s-2)(s-3),\ (s-1/3)(s+1/2)(s+1)^2), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)^2(t+1)^3(t-2)^3,\ (t-1/3)(t-2)^3(t+1/2)^4),\\ \mathbf{g}(s) &= ((s-1/3)^2(s-2)^4(s+1)^2,\ (s-1/3)(s-3/2)^5(s+1)^2). \end{cases}$$

Table 4 reports the number pairs of iterations $[N_{\mathbf{f}},N_{\mathbf{g}}]$ and the computing times in seconds for various values of desired accuracy ε for computing the tangent intersections between the three pairs of Bézier curves with various degrees (n,m). Fig. 6 visualizes the relation between computing times and desired accuracy. We can see that both `2-HybClip` and `3-HybClip` generally perform slightly better than `BezClip` for computing the tangent intersections between curves. Unfortunately, we are not able to give the exact convergence rate in theory for these cases.

We show experimental results on computing the single degenerate intersections between two curves as follows.

Example 3 (*Degenerate intersections*). We applied algorithms `2-HybClip`, `3-HybClip`, and `BezClip` to the three pairs of Bézier curves

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)^2(t-3)(t+6),\ (t-1/3)^2(t+3)(t-6)),\\ \mathbf{g}(s) &= ((s-1/3)^2(s-2)(s+6),\ (s-1/3)^2(s-3)(s+6)), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)^2(t-2)^2(t+3)^4,\ (t-1/3)^2(t+1)^5(t-4)),\\ \mathbf{g}(s) &= ((s-1/3)^2(s-2)^2,\ (s-1/3)^2(s-3)(s+1)), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= ((t-1/3)^2(t-2)^2(t+3)^4,\ (t-1/3)^2(t+1)^5(t-4)),\\ \mathbf{g}(s) &= ((s-1/3)^2(s-2)^2(s+1)^4,\ (s-1/3)(s-3)^3(s+1)^4). \end{cases}$$

Table 5 reports the number pairs of iterations $[N_{\mathbf{f}},N_{\mathbf{g}}]$ and the computing times in seconds for various values of desired accuracy ε for computing the tangent intersections between the three pairs of Bézier curves with various degrees (n,m). Fig. 7 visualizes the relation between computing times and desired accuracy. We can see that both `2-HybClip` and `3-HybClip` perform much better than `BezClip` for computing the degenerate intersections between curves. Unfortunately, we are not able to give the exact convergence rate in theory for these cases.

Table 6
Example 4 (multiple intersections): number pairs of iterations $[N_{\mathbf{f}},N_{\mathbf{g}}]$ and computing times t in seconds for various values of accuracy ε. (n,m) are degrees of the two curves $\mathbf{f}$,$\mathbf{g}$, respectively.

Degrees (n,m)	ε	10^{-4}			10^{-8}			10^{-16}			10^{-32}			10^{-64}		
		`2-HybClip`	`3-HybClip`	`BezClip`	`2-HybClip`	`3-HybClip`	`BezClip`	`2-HybClip`	`3-HybClip`	`BezClip`	`2-HybClip`	`3-HybClip`	`BezClip`	`2-HybClip`	`3-HybClip`	`BezClip`
(4,4)	$[N_{\mathbf{f}},N_{\mathbf{g}}]$	[6,9]	[6,8]	[6,10]	[7,11]	[8,8]	[7,12]	[9,11]	[8,10]	[9,14]	[10,13]	[10,10]	[11,16]	[11,14]	[10,12]	[13,18]
	t	3.884	3.806	4.056	4.212	3.993	4.258	4.321	4.258	4.570	4.617	4.539	5.007	4.711	4.680	5.428
(8,4)	$[N_{\mathbf{f}},N_{\mathbf{g}}]$	[10,20]	[9,18]	[11,21]	[13,23]	[12,19]	[15,25]	[15,26]	[13,22]	[19,29]	[18,28]	[16,23]	[23,33]	[20,31]	[17,26]	[27,37]
	t	5.382	4.992	6.084	6.224	5.631	7.160	6.973	6.271	8.392	7.612	6.910	9.547	8.299	7.659	10.670
(8,8)	$[N_{\mathbf{f}},N_{\mathbf{g}}]$	[20,42]	[16,40]	[18,48]	[25,49]	[22,43]	[26,55]	[30,52]	[25,48]	[34,63]	[36,58]	[30,51]	[41,71]	[40,65]	[33,56]	[49,79]
	t	14.461	14.695	23.509	16.926	16.676	28.002	18.220	18.564	31.418	20.576	20.436	35.115	22.604	22.214	38.969

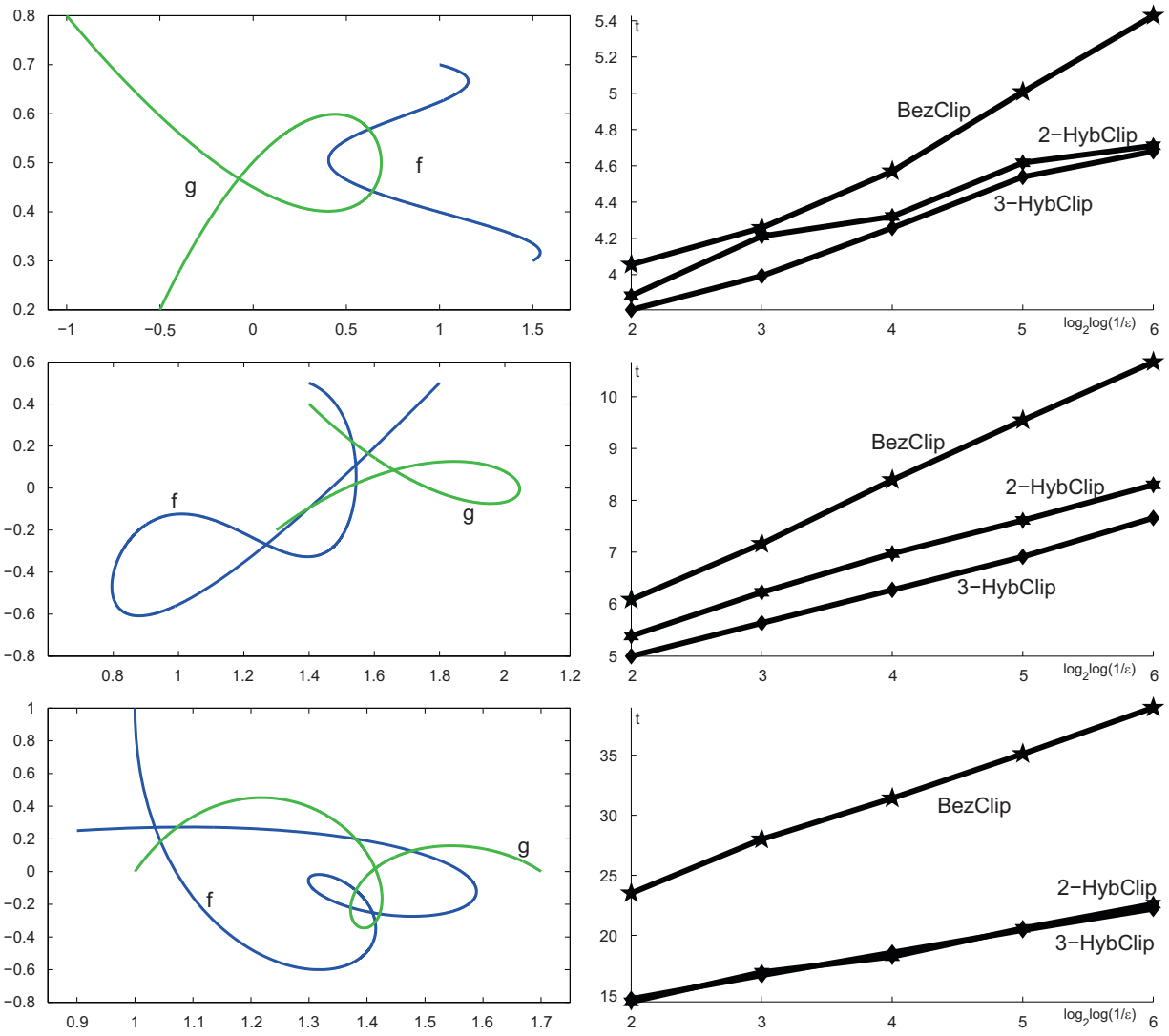

Fig. 8. Example 4 (multiple intersections). Left: the figures of the curve pairs; right: computing time t in seconds vs. accuracy. The degrees (n,m) of the two curves are (4,4) (upper row), (8,4) (middle row), (8,8) (lower row), respectively.

4.2. Multiple intersections

We show experimental results on computing multiple intersections between two curves in this section.

Example 4 (*Multiple intersections*). We applied algorithms `2-HybClip`, `3-HybClip`, and `BezClip` to the three pairs of Bézier curves. Note that these three pairs have respectively 2, 4, and 8 intersections.

$$\begin{cases} \mathbf{f}(t) &= (-\frac{51}{2}t^4+50t^3-27t^2+2t+\frac{3}{2},\ \frac{2}{5}t+\frac{3}{10}),\\ \mathbf{g}(s) &= (s^4-4s^3-\frac{3}{2}s^2+4s-\frac{1}{2},\ \frac{26}{5}s^3-\frac{39}{5}s^2+\frac{16}{5}s+\frac{1}{5}), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= (-\frac{2}{5}t^8-\frac{12}{5}t^7-\frac{84}{5}t^6+\frac{336}{5}t^5-84t^4\\ &\quad +56t^3-\frac{112}{5}t^2+\frac{16}{5}t+\frac{7}{5},\\ &\quad \frac{1123}{10}t^8-\frac{2074}{5}t^7+441t^6+70t^5-\frac{805}{2}t^4\\ &\quad +238t^3-42t^2-2t+\frac{1}{2}),\\ \mathbf{g}(s) &= (\frac{3}{2}s^4-\frac{16}{5}s^3-\frac{3}{5}s^2+\frac{12}{5}s+\frac{13}{10},\\ &\quad \frac{1}{5}s^4+\frac{24}{5}s^3-\frac{36}{5}s^2+\frac{14}{5}s-\frac{1}{5}), \end{cases}$$

$$\begin{cases} \mathbf{f}(t) &= (-\frac{53}{10}t^8+64t^7-\frac{672}{5}t^6+\frac{224}{5}t^5+98t^4\\ &\quad -\frac{448}{5}t^3+\frac{112}{5}t^2+1,\\ &\quad \frac{37}{4}t^8+52t^7-350t^6+588t^5-350t^4\\ &\quad +70t^2-20t+1),\\ \mathbf{g}(s) &= (-\frac{1}{10}s^8+\frac{24}{5}s^7-\frac{84}{5}s^6+\frac{56}{5}s^5+14s^4\\ &\quad -\frac{84}{5}s^3+\frac{14}{5}s^2+\frac{8}{5}s+1,\\ &\quad 4s^8+2s^7-28s^5+56s^3-42s^2+8s). \end{cases}$$

Table 6 reports the number pairs of iterations $[N_{\mathbf{f}},N_{\mathbf{g}}]$ and the computing times in seconds for various values of desired accuracy ε for computing the tangent intersections between the three pairs of Bézier curves with various degrees (n,m). Fig. 8 visualizes the relation between computing times and desired accuracy. We can see that both `2-HybClip` and `3-HybClip` perform better than `BezClip` for computing the multiple intersections between curves. Note that `2-HybClip` and `3-HybClip` perform much similar in the case of third curve pair.

5. Conclusion

We have proposed the hybrid clipping technique for computing all the intersections between two planar Bézier curves within a certain

accuracy in a given domain. The algorithm is based on the convex hull property of Bézier curves. By clipping a fat curve of one curve with a fat line that bounds the other curve, we generate a reduced subdomain that encloses the intersection. Therefore, a sequence of subdomains that is guaranteed to converge to the corresponding intersection will be obtained by combining with bisection steps. We have proved that the hybrid clipping technique has at least a quadratic convergence rate which is at least as fast as Bézier clipping for transversal intersections. Experimental results have shown that the hybrid clipping performs better than Bézier clipping for transversal intersections, much better than Bézier clipping for degenerate intersections, but a slightly better than Bézier clipping for tangent intersections. It is worthwhile pointing out that both tangent intersections and degenerate intersections are rare in practical applications. Therefore, the hybrid clipping technique is practically useful for computing the transversal intersections between planar curves.

We guess that the hybrid clipping has higher convergence rate than Bézier clipping at tangent and degenerate intersections. As one of our future works we will try to find it out. Another future work will focus on the extension of the technique to the intersection computation between two rational curves or between two Bézier surfaces. Finally, we can use other methods, for instance, the SLEVEs [16–18] method, to linearly bound the curves and then compute their intersections. It is worthwhile to try this method for curve intersections which is expected to gain better performance. However, the computation of the SLEVEs for the spline curve is a bit complicated and thus makes the convergence hard to analyze.

Acknowledgement

This work is supported by the National Natural Science Foundation of China (61070071) and the National Basic Research Program of China 2011CB302400.

Appendix A. Optimal degree reduction of Bézier curve

Degree reduction of Bézier curves has been well studied in the literature. To make this paper self-contained, we give the main formulae for computing the optimal L_2 approximation (with lower degree) to a given Bézier curve. More details can be found in [19].

Given a degree n Bézier curve $\mathbf{f}(t)=\sum_{i=0}^{n}\mathbf{a}_i B_{i,[\alpha,\beta]}^{n}(t)\in \Pi_{[\alpha,\beta]}^{n}$, the control points of its optimal degree k $(k<n)$ Bézier curve $\mathbf{p}(t)=\sum_{i=0}^{k}\tilde{\mathbf{c}}_i B_{i,[\alpha,\beta]}^{k}\in \Pi_{[\alpha,\beta]}^{k}$ can be computed as

$$(\tilde{\mathbf{c}}_0,\tilde{\mathbf{c}}_1,\ldots,\tilde{\mathbf{c}}_k)^{\mathrm{T}}=(\mathbf{F}_{(k+1)\times(k+1)}^{k}\mathbf{B}_{(k+1)\times(n+1)}^{n}-\mathbf{F}_{(k+1)\times(k+1)}^{k}\mathbf{C}_{(k+1)\times(n-k)}^{n}\cdot(\mathbf{C}_{(n-k)\times(n-k)}^{n})^{-1}\mathbf{B}_{(n-k)\times(n+1)}^{n})\cdot(\mathbf{a}_0,\mathbf{a}_1,\ldots,\mathbf{a}_n)^{\mathrm{T}},\tag{A.1}$$

where

$$\mathbf{B}_{(n+1)\times(n+1)}^{n}=\begin{pmatrix}\mathbf{B}_{(k+1)\times(n+1)}^{n}\\ \mathbf{B}_{(n-k)\times(n+1)}^{n}\end{pmatrix},$$

$$\mathbf{C}_{(n+1)\times(n-k1)}^{n}=\begin{pmatrix}\mathbf{C}_{(k+1)\times(n-k)}^{n}\\ \mathbf{C}_{(n-k)\times(n-k)}^{n}\end{pmatrix},$$

$$\mathbf{F}_{(n+1)\times(n+1)}^{n}=(f_{ij})_{(n+1)\times(n+1)},$$

$$\mathbf{B}_{(n+1)\times(n+1)}^{n}=(b_{ij})_{(n+1)\times(n+1)},$$

$$\mathbf{C}_{(n+1)\times(n-k)}^{n}=(c_{ij})_{(n+1)\times(n-k)},$$

with

$$f_{ij}=\begin{cases}\binom{n-j}{i-j}/\binom{n}{i} & \text{if } i\geq j;\\ 0 & \text{otherwise,}\end{cases}$$

$$b_{ij}=\begin{cases}(-1)^{i+j}\binom{n}{i}\binom{i}{j} & \text{if } i\geq j;\\ 0 & \text{otherwise,}\end{cases}$$

$$c_{ij}=\begin{cases}\sum_{l=i}^{m+j+1}(-1)^{l+i}\binom{m+j+1}{m+j+1-l}\binom{m+j+1+l}{l}\binom{l}{i} & \text{if } (*),\\ 0 & \text{otherwise,}\end{cases}$$

where $(*)$ means $0\leq j\leq n-m-1$, $0\leq i\leq m+j+1$.

Appendix B. Degree elevation of Bézier curve

To make this paper self-contained, we also give the formulae for computing the degree elevation of Bézier curve. Suppose $\mathbf{p}(t)=\sum_{i=0}^{k}\tilde{\mathbf{c}}_i B_{i,[\alpha,\beta]}^{k}\in \Pi_{[\alpha,\beta]}^{k}$ is a degree $k(k<n)$ Bézier curve. As $\Pi_{[\alpha,\beta]}^{k}\subseteq \Pi_{[\alpha,\beta]}^{n}$, $\mathbf{p}(t)$ can be elevated to a degree n Bézier curve with the form $\mathbf{p}(t)=\sum_{i=0}^{n}\mathbf{c}_i B_{i,[\alpha,\beta]}^{n}$ where the control points can be calculated as [2]

$$(\mathbf{c}_0,\mathbf{c}_1,\ldots,\mathbf{c}_n)^{\mathrm{T}}=\mathbf{F}_{(n+1)\times(n+1)}^{n}\mathbf{E}_{(n+1)\times(k+1)}^{n}\mathbf{B}_{(k+1)\times(k+1)}^{k}\cdot(\tilde{\mathbf{c}}_0,\tilde{\mathbf{c}}_1,\ldots,\tilde{\mathbf{c}}_k)^{\mathrm{T}},\tag{B.1}$$

where $\mathbf{F}$ and $\mathbf{B}$ are the same in Appendix A and

$$\mathbf{E}_{(n+1)\times(k+1)}^{n}=(e_{ij})_{(n+1)\times(k+1)},\quad e_{ij}=\begin{cases}1 & \text{if } i=j;\\ 0 & \text{otherwise.}\end{cases}$$

References

[1] Dokken T. Aspects of intersection algorithms and approximations. PhD thesis. University of Oslo; 1997.
[2] Farin G. Curves and surfaces for CAGD: a practical guide. 5th ed. Morgan Kaufmann Publishers Inc.; 2002.
[3] Mørken K, Reimers M, Schulz C. Computing intersections of planar spline curves using knot insertion. Comput Aided Geometric Des 2009;26(3):351–66.
[4] Lane J, Riesenfeld R. A theoretical development for the computer generation and display of piecewise polynomial surfaces. IEEE Trans Pattern Anal Mach Intell 1980;2:35–46.
[5] Koparkar P, Mudur S. A new class of algorithms for the processing of parametric curves. Comput Aided Des 1983;15:41–5.
[6] Sederberg T, Parry S. Comparison of three curve intersection algorithms. Comput Aided Des 1986;18:58–63.
[7] Sederberg T, Nishita T. Curve intersection using Bézier clipping. Comput Aided Des 1990;22(9):538–49.
[8] Schulz C. Bézier clipping is quadratically convergent. Comput Aided Geometric Des 2009;26(1):61–74.
[9] Bartoň M, Jüttler B. Computing roots of polynomials by quadratic clipping. Comput Aided Geometric Des 2007;24(3):125–41.
[10] Liu L, Zhang L, Lin B, Wang G. Fast approach for computing roots of polynomials using cubic clipping. Comput Aided Geometric Des 2009;26(5):547–559.
[11] McNamee J. Bibliographies on roots of polynomials. J Comput Appl Math 2002;142:433–4.
[12] Bartoň M. B. Jüttler, Computing roots of systems of polynomials by linear clipping. SFB F013 technical report.
[13] Moore B, Jüttler B. Computing roots of polynomials using bivariate quadratic clipping. In: The 2nd international conference on mathematical aspects of computer and information sciences; 2007.
[14] Mourrain B, Pavone JP. Subdivision methods for solving polynomial equations. J Symbolic Comput 2009;44(3):292–306.
[15] Sherbrooke E, Patrikalakis N. Computation of the solutions of nonlinear polynomial systems. Comput Aided Geometric Des 1993;10(5):379–405.
[16] Lutterkort D, Peters J. Optimized refinable enclosures of multivariate polynomial pieces. Comput Aided Geometric Des 2001;18(9):851–63.
[17] Lutterkort D, Peters J. Tight linear bounds on the distance between a spline and its b-spline control polygon. Numer Math 2001;89:735–48.
[18] Peters J, Wu X. SLEVEs for planar spline curves. Comput Aided Geometric Des 2004;21(6):615–35.
[19] Zhang R, Wang G. Constrained Bézier curves best multi-degree reduction in the l_2-norm. Prog Nat Sci 2005;15(9):843–50.