Computation on Parametric Curves with Applications in Localization and Grasping

Yan-Bin Jia

Iowa State University, Ames, IA 50011, USA

Abstract. This paper investigates efficient curve processing that originates from the localization and grasping of 2D curved objects. The first algorithm locates the boundary section of an object traced out by a rolling finger based on the length and total curvature information obtained with the finger's tactile sensor. The algorithm slides an imaginary segment along the object boundary by alternatively marching its two endpoints forward, stretching or contracting the segment if necessary.

The second algorithm computes all pairs of antipodal points on an object. Two fingers placed at such a pair of points can achieve a force-closure grasp in the presence of friction. Dissecting the boundary into segments everywhere convex or everywhere concave, the algorithm marches simultaneously on every two segments with opposing normals and alternates marching with numerical bisection recursively. It builds on a procedure that constructs common tangents of two curves with quadratic local convergence.

Completeness (up to numerical resolution) of the above algorithms is established by applying curvature-based analyses. Dissection and the coupling of marching with bisection introduced in this paper are potentially applicable to optimization problems involving curves and curved surfaces.

1 Introduction

Geometry often plays an essential role in robot tasks such as sensing, grasping, pushing, path planning, dextrous manipulation, and control. Sensing, manipulation, and planning strategies for polygonal and polyhedral objects have been studied extensively in the past. Many of these strategies are designed using combinatorial techniques that take advantage of the absence of local (differential) geometry.

Curved shapes are frequent subjects of maneuvers by the human hand and they share the differential nature with contact kinematics and dynamics which are responsible for such maneuvers [2]. This paper presents efficient algorithms that process plane curves for tasks including parts localization and grasping.

Section 2 studies how a disk-like finger with tactile ability localizes itself while rolling on a stationary curved object. Section 3 describes an algorithm that constructs all common tangents of two curve segments. This algorithm is part of the preprocessing in Sect. 5 that prepares for the computation of antipodal grasps in Sect. 4.

The curve $\boldsymbol{\alpha}(s)$ considered in this paper is simple, regular, and twice continuously differentiable. In case no ambiguity arises, the parameter s also refers to the point $\boldsymbol{\alpha}(s)$ on the curve. The curvature $\kappa(s)$ of $\boldsymbol{\alpha}$ is zero at only a finite number of points. A point z on $\boldsymbol{\alpha}$ is a *simple inflection* if $\kappa(z) = 0$ but $\kappa'(z) \neq 0$. The *total curvature* of a segment of $\boldsymbol{\alpha}$ over $[a, b]$ is given by $\Phi(a, b) = \int_a^b \kappa \|\boldsymbol{\alpha}'(u)\| \, du$. This integral measures the amount of rotation of the tangent as a point moves from a to b along the curve[1].

1.1 Related Work

Hong *et al.* [12] proved the existence of two pairs of antipodal points on a closed, simple, and smooth convex curve or surface. Chen and Burdick [6] computed antipodal points on 2D and 3D shapes through minimizing a grasping energy function.

Two vertices of a polygon are antipodal if they admit parallel supporting lines. The polygon's diameter is equal to the distance between the two furthest vertices (which must be antipodal). For n points in three dimensions, the number of antipodal pairs can be $O(n^2)$ while the diameter can be achieved by $2n - 2$ pairs of points [11]. Preparata and Shamos [27] described a linear-time algorithm that finds the diameter of a convex polygon with n vertices in one traversal. Clarkson and Shor [7] presented a randomized algorithm to compute the diameter in $O(n \log n)$ expected time. Near-linear-time deterministic algorithms were developed through derandomizing their algorithm in [5,19]. Yao [32] offered a different approach that led to an $o(n^2)$-time algorithm for computing the diameter of a point set in any dimensions.

Goodman [10] gave an upper bound on the number of inflection points on parametric spline curves. Sakai [29] obtained the distribution of inflection points and cusps on a parametric rational cubic curve. Manocha and Canny [17] employed the Sturm sequence method to find inflection points on rational curves. Mokhtarian and Mackworth [21] used inflection points for plane curve description and matching.

Preparata and Hong [26] developed a "walking" strategy to construct common supporting lines of two convex polygons in linear time. The procedure `Common-Tangent` in Sect. 3.2 can be viewed as its continuous counterpart where tangent lines (in place of supporting lines) are used to determine the next stops.

Allen and Roberts [1] deployed robot fingers to obtain a number of contact points around an object and used fitting to reconstruct the object's shape. Fischler [9] described an algorithm that locates points with extreme curvatures on planar curves and reconstructs the original curves based on these points. Erdmann [8] showed that the local geometry of a curved object with known angular velocity can be recovered by two passive linear tactile sensors.

[1] The total curvature over $[a, b]$ has a closed form if it is within $(0, 2\pi]$ and can be fast computed otherwise.

Extending this work, Moll and Erdmann [22] applied quasi-static dynamics to reconstruct the shape of a convex object and estimate its motion from tactile readings on two palms in frictionless contact with the object.

Mishra *et al.* [20] gave upper bounds on the numbers of frictionless fingers that are sufficient for equilibrium and force-closure grasps, respectively, on objects with piecewise smooth boundaries. Markenscoff *et al.* [18] determined the number of fingers to immobilize 2-D and 3-D objects with piecewise smooth boundaries. Ponce *et al.* [25] employed parallel cell decomposition to compute pairs of maximal-length segments on a piecewise smooth curved 2D object that guarantee force closure with friction.

2 Locating Segments on a Curve

We begin with the problem of determining the location of a finger rolling on a stationary object bounded by $\boldsymbol{\alpha}$. A tactile sensor is mounted on the finger.

To simplify our analysis, the finger is assumed to be a disk described by $r(\cos u, \sin u)$, where u locates the contact on the disk and is determined from the tactile data. The disk is rolling with angular velocity ω. Let s be the location of contact on $\boldsymbol{\alpha}$, as shown in Fig. 1(a).

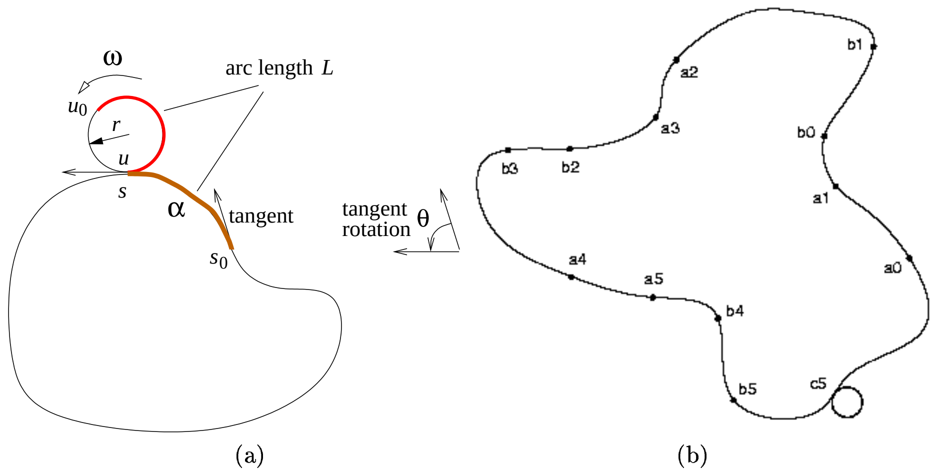

Fig. 1. (a) A disk rolling on a stationary curved object; (b) a disk localizing itself by rolling from a_5 to b_5 and then to c_5. The localization algorithm finds six segments over $[a_i, b_i]$, $0 \leq i \leq 5$, respectively, which have the length and total curvature computed from tactile data. These ambiguities are eliminated using the curve length and total curvature over $[b_5, c_5]$

Contact kinematics [4,23] yield the velocities of the contact point on the disk and the object, respectively:

$$\dot{u} = -\frac{\omega}{1 + r \cdot \kappa(s)}, \tag{1}$$

$$\dot{s} = \frac{\omega}{\|\boldsymbol{\alpha}'(s)\| \Big(1/r + \kappa(s)\Big)}. \tag{2}$$

From (1) and (2), we obtain

$$\dot{s}\, \|\boldsymbol{\alpha}'(s)\| = -r\dot{u},$$
$$\dot{s}\, \kappa(s)\, \|\boldsymbol{\alpha}'(s)\| = \dot{u} + \omega.$$

Integrate these two equations over the time period $[0, \tau]$:

$$\int_{s_0}^{s_1} \|\boldsymbol{\alpha}'(s)\|\, ds = -r(u_1 - u_0) \;\; = \;\; L, \tag{3}$$

$$\int_{s_0}^{s_1} \kappa(s)\, \|\boldsymbol{\alpha}'(s)\|\, ds = u_1 - u_0 + \int_0^{\tau} \omega(\xi)\, d\xi \;\; = \;\; \theta, \tag{4}$$

where $s_0 = s(0)$, $s_1 = s(\tau)$, $u_0 = u(0)$, and $u_1 = u(\tau)$.

The length L of the object boundary traced out by the contact can be determined from tactile readings u_0 and u_1. The amount of disk rotation $\int_0^{\tau} \omega(\xi) d\xi$ is also known. Thus the total curvature θ over $[s_0, s_1]$ is known by (4). Localization thus reduces to *finding a curve segment on* $\boldsymbol{\alpha}$ *with length* L *and total curvature* θ.

When multiple segments are found to have length L and total curvature θ, the disk eliminates the ambiguities by continuing the rolling and using the length and total curvature of the next segment it traces out (see Fig. 1(b)).

Equations (3) and (4) need to be solved numerically. In general, arc length $\ell(a, b) = \int_a^b \|\boldsymbol{\alpha}'(s)\|\, ds$ has no closed form. We first consider the case that the boundary curve $\boldsymbol{\alpha}$ is convex and present a marching algorithm.

The algorithm marches two endpoints s and t of a hypothesized segment counterclockwise along $\boldsymbol{\alpha}$. It starts at location $s_0 = 0$ and advances repeatedly by a step size until it reaches the point q where $\Phi(s_0, q) = \theta$. If $\ell(s_0, q) > L$, at step $i \geq 0$ advance from t_{i-1} to t_i (set $t_0 \leftarrow q$) where $\Phi(s_i, t_i) = \theta$ and then advance from s_i to s_{i+1} where $\ell(s_{i+1}, t_i) = L$. If $\ell(s_0, q) < L$, at step i advance from t_{i-1} (or from q if $i = 0$) to t_i where $\ell(s_i, t_i) = L$ and then advance from s_i to s_{i+1} where $\Phi(s_{i+1}, t_i) = \theta$. Figure 2 illustrates the working of the algorithm in these two cases.

It can be shown by induction [15] that the two sequences $s_0, s_1, \ldots$ and $t_0, t_1, \ldots$ generated above are monotone and will converge to the endpoints of the first segment on $\boldsymbol{\alpha}$ that satisfies (3) and (4). Furthermore, the convergence rate [31, p. 264] is linear and given by the ratio $\min\{\kappa(a)/\kappa(b), \kappa(b)/\kappa(a)\}$. To find the next segment on $\boldsymbol{\alpha}$, we reset s_0 to be slightly greater than s_i for large enough i, and repeat the same procedure.

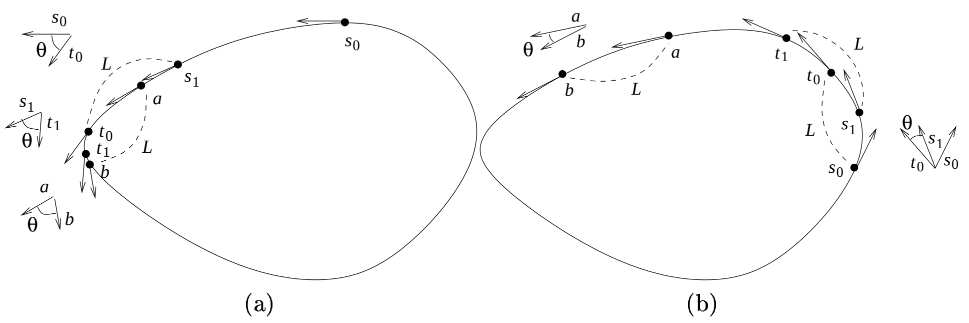

Fig. 2. Two cases of marching s and t: **(a)** $\Phi(s_i, t_i) = \theta$ but $\ell(s_i, t_i) > L$; **(b)** $\ell(s_i, t_i) = L$ but $\Phi(s_i, t_i) > \theta$. Both marches converge to (a, b) where $\ell(a, b) = L$ and $\Phi(a, b) = \theta$

The correctness of the above march relies on that the total curvature $\Phi(s, t)$ has partial derivatives $\partial\Phi/\partial s < 0$ and $\partial\Phi/\partial t > 0$ for all $s < t$. But this is no longer true when $\boldsymbol{\alpha}$ has concavities. For example, $\partial\Phi/\partial s > 0$ when $\kappa(s) < 0$.

When $\boldsymbol{\alpha}$ is not convex, marching has four basic modes: *convex-convex* ($\kappa(s) \geq 0$ and $\kappa(t) \geq 0$), *concave-concave* ($\kappa(s) \leq 0$ and $\kappa(t) \leq 0$), *convex-concave* ($\kappa(s) \geq 0$ and $\kappa(t) \leq 0$), and *concave-convex* ($\kappa(s) \leq 0$ and $\kappa(t) \geq 0$). Within each mode, the hypothesized curve segment over (s, t) slides along $\boldsymbol{\alpha}$ until a simple inflection point is reached by either s or t so that the mode changes. Location(s) of the desired curve segment, if exists, is also found during the advancement. Sliding is done by increasing only one of s and t at a time while simultaneously keeping track of where the other should be (without actually increasing it).

For a detailed description of all operations, we refer to [15] where the following result is established:

Theorem 1. *All segments with length L and total curvature θ can be found on a closed simple curve over domain $[0, D]$ up to numerical resolution in at most $5D/h$ steps, where h is the step size.*

The algorithm runs asymptotically as fast as evaluating the length of the entire curve through numerical integration.

3 Common Tangent Construction

For clarity of presentation, in Sects. 3 and 4 we consider that the curve $\boldsymbol{\alpha}$ is *unit-speed*, that is, $\|\boldsymbol{\alpha}'(s)\| = 1$. All algorithms to be presented have been extended straightforwardly to and implemented on arbitrary-speed curves.

Two segments $\mathcal{S}$ and $\mathcal{T}$ of $\boldsymbol{\alpha}$ are defined on subdomains (s_a, s_b) and (t_a, t_b), respectively, where $(s_a, s_b) \cap (t_a, t_b) = \emptyset$. Here $s_a < s_b$ always holds but $t_a > t_b$ is allowed, in which case (t_a, t_b) refers to the interval (t_b, t_a).

Denote by $T(s) = \boldsymbol{\alpha}'(s)$ the unit tangent of $\boldsymbol{\alpha}$ and denote by $N(s)$ the unit normal such that $T(s) \times N(s) = 1$. The following conditions are satisfied by $\mathcal{S}$ and $\mathcal{T}$:

(i) $\kappa > 0$ everywhere or $\kappa < 0$ everywhere on $\mathcal{S}$ and $\mathcal{T}$, with $\kappa = 0$ possible only at s_a, s_b, t_a, t_b.
(ii) $N(s_a) + N(t_a) = 0$ and $N(s_b) + N(t_b) = 0$.
(iii) $-\pi \le \Phi(s_a, s_b) = -\Phi(t_a, t_b) \le \pi$.

For now we present algorithms working on $\mathcal{S}$ and $\mathcal{T}$. Section 5 will describe how to preprocess $\boldsymbol{\alpha}$ to generate pairs of segments satisfying (i)–(iii).

Under these conditions, a one-to-one correspondence exists between a point s on $\mathcal{S}$ and a point t on $\mathcal{T}$:

$$N(s) + N(t) = 0, \qquad \text{or equivalently,} \tag{5}$$

$$T(s) + T(t) = 0. \tag{6}$$

Let $g(s,t) = N(s) \times N(t)$. Since $\frac{\partial g}{\partial t} = N(s) \times (-\kappa(t)T(t)) = -\kappa(t) \neq 0$ under (5),[2] by the Implicit Function Theorem, the equation $g(s,t) = 0$ defines t locally as a function of s. From now on, we refer to t as the *opposite point* of s. Differentiate (6) with respect to s and then substitute (5) in. We obtain that $\kappa(s) - \kappa(t)\,\frac{dt}{ds} = 0$ and

$$\frac{dt}{ds} = \frac{\kappa(s)}{\kappa(t)}. \tag{7}$$

3.1 Classification of Common Tangents

Theorem 2. *The segments $\mathcal{S}$ and $\mathcal{T}$ under conditions (i)–(iii) have at most two common tangents. The segments are always on the same side of every common tangent or always on different sides of every common tangent.*

A proof of Theorem 2 is given in [14]. Based on the theorem, we classify four configurations (shown in Fig. 3) in which at least one common tangent of $\mathcal{S}$ and $\mathcal{T}$ exists.

In configurations (a) and (c) the two curve segments bend in the same direction while in (b) and (d) they bend in opposite directions. The bending direction of the curve $\boldsymbol{\alpha}$ at a point u is indicated by $\kappa(u)N(u)$. So we check the dot product $\kappa(s_c)N(s_c) \cdot \kappa(t_c)N(t_c) = -\kappa(s_c)\kappa(t_c)$, where $s_c = \frac{s_a + s_b}{2}$ and $N(t_c) = -N(s_c)$. If the dot product is positive, then the two segments $\mathcal{S}$ and $\mathcal{T}$ bend in the same direction; otherwise, they bend in opposite directions.

Define the function $d(s)$ as the translational distance along the normal $N(s)$ from the tangent line of $\mathcal{S}$ at s to the tangent line of $\mathcal{T}$ at its opposite point t. This function is continuous since the curve $\boldsymbol{\alpha}$ is continuously differentiable. Obviously, $d(s) = 0$ if and only if the tangent line through s

[2] The Frenet formulas [24, pp. 56–58] for unit-speed plane curves state that $T'(s) = \kappa(s)N(s)$ and $N'(s) = -\kappa(s)T(s)$.

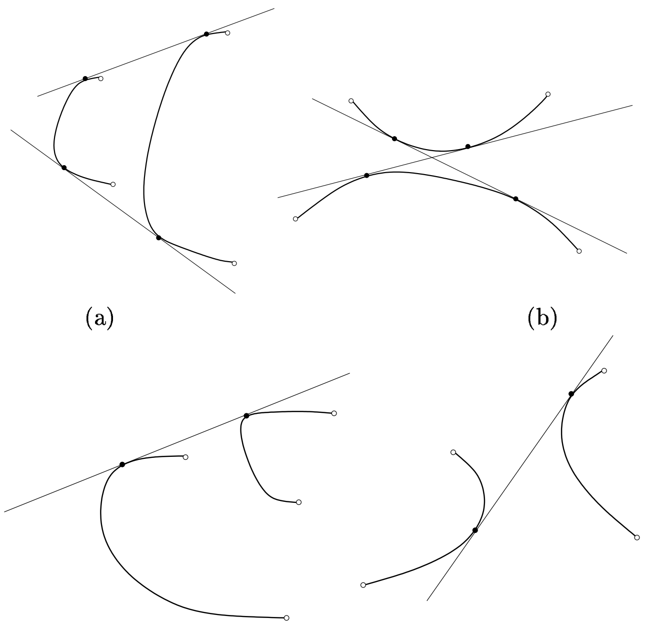

Fig. 3. Four configurations where two segments satisfying conditions (i)–(iii) have at least one common tangent

is a common tangent. It follows from condition (i) that $d'(s) \neq 0$ whenever $d(s) = 0$.

When $d(s_a)$ and $d(s_b)$ have different signs, there exists some $c \in (s_a, s_b)$ such that $d(c) = 0$. Namely, the line passing through c and its opposite point on $\mathcal{T}$ is a common tangent. Because there are at most two common tangents and d has only simple zeros, we can infer that this is the unique common tangent of $\mathcal{S}$ and $\mathcal{T}$ and the configuration is either (c) or (d) in Fig. 3. These two configurations can be further distinguished.

When $d(s_a)$ and $d(s_b)$ have the same sign, $\mathcal{S}$ and $\mathcal{T}$ may have two common tangents as in configurations (a) and (b) or they may not have a common tangent at all. We will need to look at whether the two segments bend in the same direction or in opposite directions and make use of the regions partitioned by the segments as well as the four tangent lines at their endpoints. A procedure is given in [14] to determine whether two or zero common tangent exists.

3.2 An Iterative Method

Below we offer an iterative method to construct a common tangent. The method makes the following two assumptions that will be removed in the end of the section.

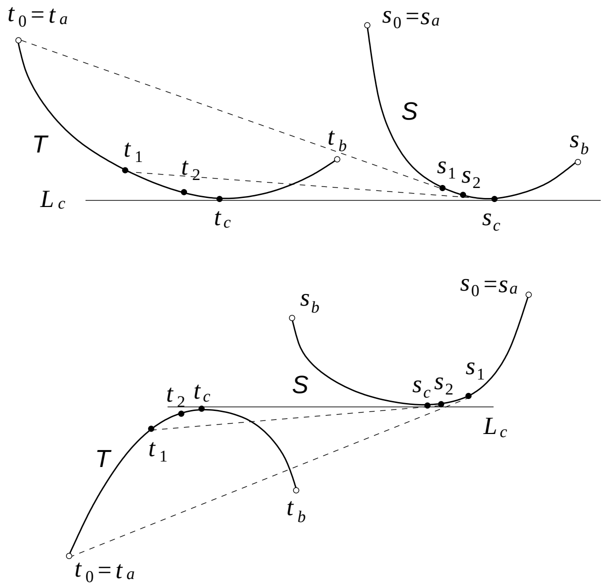

Fig. 4. Constructing a common tangent iteratively

1. There exists exactly one common tangent L_c of $\mathcal{S}$ and $\mathcal{T}$ incident on s_c and t_c, respectively.
2. A point traversing $\mathcal{T}$ from t_a to t_b is moving towards s_c at the time it reaches t_c. So the two segments and L_c are in one of the two configurations shown in Fig. 4.

The iteration starts at $s_0 = s_a$ and $t_0 = t_a$ and maintains two invariants:

$$\Big(\boldsymbol{\alpha}(s_{i+1}) - \boldsymbol{\alpha}(t_i)\Big) \times T(s_{i+1}) = 0; \tag{8}$$

$$N(s_{i+1}) + N(t_{i+1}) = 0. \tag{9}$$

The point $s_{i+1} \in (s_i, s_b)$ is found by a primitive `point-of-tangency`. Such a point of tangency always exists under assumptions 1 and 2. The procedure call for finding the two points of tangency s_c and t_c is `Common-Tangent`(s_a, s_b, t_a, t_b).

We have established [14] that the two sequences $\{s_i\}$ and $\{t_i\}$ generated according to (8) and (9) are strictly increasing (decreasing) and bounded by s_c and t_c. They must converge to two points that satisfy (8), i.e., s_c and t_c.

Next, we determine the order of local convergence of $\{s_i\}$ and $\{t_i\}$. Let g be the iteration function such that $s_{i+1} = g(s_i)$. Differentiate equation (8) with respect to s_i, simplify, and substitute (7) in:

$$-T(t_i) \times T(s_{i+1}) \frac{\kappa(s_i)}{\kappa(t_i)} + \|\boldsymbol{\alpha}(s_{i+1}) - \boldsymbol{\alpha}(t_i)\|\, \kappa(s_{i+1}) \frac{ds_{i+1}}{ds_i} = 0.$$

From above we obtain the derivative of the iteration function $g'(s_i)$, in particular,

$$g'(s_c) = \frac{T(t_c) \times T(s_c)}{\|\boldsymbol{\alpha}(s_c) - \boldsymbol{\alpha}(t_c)\|\kappa(s_c)} \frac{\kappa(s_c)}{\kappa(t_c)} = 0,$$

because $T(t_c) \times T(s_c) = 0$. Meanwhile, we also obtain

$$g''(s_c) = \frac{\kappa(s_c)}{\kappa(t_c)\|\boldsymbol{\alpha}(s_c) - \boldsymbol{\alpha}(t_c)\|} \neq 0.$$

Hence the two sequences $\{s_i\}$ and $\{t_i\}$ have *quadratic* convergence rate.

To remove assumptions 1 and 2 made earlier, we need to determine the order of the four endpoints passed on as arguments for calling `Common-Tangent`. This order determines which of the two pairs of endpoints to start the iteration at, and, within the pair, which endpoint to update first. This would not be an issue for configuration (b) in Fig. 3, where the two common tangents can be found using the procedure calls

`Common-Tangent`(s_a, s_b, t_a, t_b) and `Common-Tangent`(s_b, s_a, t_b, t_a)

regardless of the vertex labeling.

When the configuration is one of (a), (c), (d), we look at the parallel tangent lines L_a and L'_a of $\boldsymbol{\alpha}$ at s_a and t_a, respectively. For instance, if the translation from L_a to L'_a is along the bending direction of $\mathcal{S}$ then we start the iteration by updating from $t_0 = t_a$ to t_1, the point of tangency of a tangent line to $\mathcal{T}$ through s_a, then from $s_0 = s_a$ to s_1, where $N(s_1) + N(t_1) = 0$, and so on.

4 Computation of Antipodal Points

From now on we require that $\boldsymbol{\alpha}$ be also closed. Two points a and b on $\boldsymbol{\alpha}$ are *antipodal* if their normals are collinear and pointing at each other, or more precisely,

$$\begin{aligned} N(a) + N(b) &= 0, \\ N(a) \times \Big(\boldsymbol{\alpha}(b) - \boldsymbol{\alpha}(a)\Big) &= 0, \\ N(a) \cdot \Big(\boldsymbol{\alpha}(b) - \boldsymbol{\alpha}(a)\Big) &> 0. \end{aligned}$$

We need only consider that the curvature of $\boldsymbol{\alpha}$ is not constant[3]. For example, the closed curve in Fig. 5(a) has eight pairs of antipodal points (numbered the same within each pair). Two fingers placed at any of these eight pairs of points will form a force-closure grasp that can resist any arbitrary external

[3] This excludes a circle on which any two points determining a diameter are antipodal.

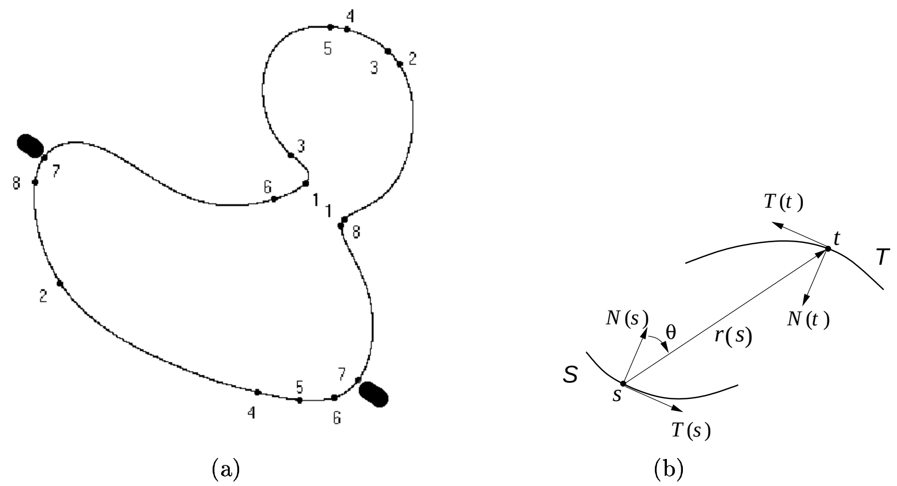

Fig. 5. (a) Eight pairs of antipodal points on a curved shape. The grasp at pair 7 (or at any other pair) is force-closure. (b) Antipodal angle θ

force and torque in the presence of contact friction. Such a grasp is referred to as an *antipodal grasp.*

In this section, we present an algorithm that *finds all antipodal points on a pair of segments $\mathcal{S}$ and $\mathcal{T}$ of $\boldsymbol{\alpha}$* that meet conditions (i)–(iii) in Sect. 3 as well as the conditions below:

(iv) Neither s_a and t_a nor s_b and t_b are antipodal.
(v) $N(s) \cdot \Big(\boldsymbol{\alpha}(t) - \boldsymbol{\alpha}(s)\Big) > 0$ for all $s \in (s_a, s_b)$.

Under condition (iii), no two points on $\mathcal{S}$ (which does not include s_a or t_a) are antipodal and the same holds for $\mathcal{T}$. Condition (v) addresses that two opposite points may be antipodal only if their normals do not point away from each other.

Pairs of segments on $\boldsymbol{\alpha}$ that meet conditions (i)–(v) will be generated in the preprocessing step described in Sect. 5.

4.1 Antipodal Angle

Define the *antipodal angle*[4] $\theta(s)$ as the rotation angle from the normal $N(s)$ to the vector $\boldsymbol{r}(s) = \boldsymbol{\alpha}(t) - \boldsymbol{\alpha}(s)$, where t is the opposite point of s (see Fig. 5(b)). Under condition (v), $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2})$. By definition, s and t are antipodal *if and only if* $\theta(s) = 0$.

To determine the change rate of θ, we first calculate the derivative:

$$\frac{d}{ds}\|\boldsymbol{r}(s)\| = \frac{d}{ds}\sqrt{\boldsymbol{r}(s)\cdot\boldsymbol{r}(s)} \;=\; \left(\frac{\kappa(s)}{\kappa(t)} + 1\right)\sin\theta, \qquad \text{by (7)}.$$

[4] In [3], it is referred to as the friction angle.

From Fig. 5(b) we see that $\sin\theta = N(s) \times \boldsymbol{r}(s)/\|\boldsymbol{r}(s)\|$. Differentiate both sides of this equation and substitute the above expression for $\frac{d}{ds}\|\boldsymbol{r}(s)\|$ in. After a few more steps, we obtain

$$\theta'(s) \quad = \quad -\kappa(s) + \frac{\cos\theta}{\|\boldsymbol{r}(s)\|}\left(\frac{\kappa(s)}{\kappa(t)} + 1\right). \tag{10}$$

Suppose $\boldsymbol{\alpha}$ is at least $k+1 \geq 2$ times continuously differentiable. Two points s^* on $\mathcal{S}$ and t^* on $\mathcal{T}$ are called *antipodal points of order k* if $\theta(s^*) = \theta'(s^*) = \cdots = \theta^{(k-1)}(s^*) = 0$ but $\theta^{(k)}(s^*) \neq 0$. When $k = 1$, s^* and t^* are *simple* antipodal points. When $k = 2$, equation (10) reduces to $\|\boldsymbol{r}(s^*)\| = \frac{1}{\kappa(s^*)} + \frac{1}{\kappa(t^*)}$; that is, the osculating circles at s^* and t^* are *concentric.*

We are primarily interested in finding simple antipodal points. So from now on we assume that *higher order antipodal points do not exist.* Our algorithm employs different strategies according as whether the signs of the antipodal angles $\theta(s_a)$ and $\theta(s_b)$ at the endpoints of $\mathcal{S}$ are the same.

4.2 Opposite Angle Signs at Endpoints

When $\theta(s_a)$ and $\theta(s_b)$ have different signs, at least one pair of antipodal points exists on $\mathcal{S}$ and $\mathcal{T}$. Bisection method [28, pp. 261–263] guarantees to find one such pair.

In the case where both segments are concave, $\kappa(s) < 0$ and $\kappa(t) < 0$, we have $\theta'(s) > 0$ by (10). The antipodal angle θ increases monotonically from s_a to s_b. A unique pair of antipodal points exists if $\theta(s_a) < 0$ and $\theta(s_b) > 0$. Otherwise, no antipodal points exist.

4.3 Same Angle Sign at Endpoints

In this case, at least one of $\mathcal{S}$ and $\mathcal{T}$ must be convex in order to have antipodal points. Multiple pairs of antipodal points may exist.

Two Convex Segments When $\mathcal{S}$ and $\mathcal{T}$ are both convex, we march on both segments using the fact that the vector $\boldsymbol{r}(s)$ rotates counterclockwise as s increases from s_a to s_b. This is because $\frac{d\boldsymbol{r}}{ds} \times \boldsymbol{r} < 0$. To see the inequality, note that the vector $\frac{d\boldsymbol{r}}{ds} = T(t)\left(1 + \frac{\kappa(s)}{\kappa(t)}\right)$ is in the direction of $T(t)$ since $\kappa(s), \kappa(t) > 0$. But $T(t) \times \boldsymbol{r}(s) = -T(s) \times \boldsymbol{r}(s) < 0$ under condition (v).

Figure 6 illustrates the march when $\theta(s_a) < 0$ and $\theta(s_b) < 0$. It starts with s and t at $s_0 = s_b$ and $t_0 = t_b$, respectively. As s moves toward s_a, the vector $\boldsymbol{r}(s)$ rotates clockwise. At the ith iteration step move s from s_i to s_{i+1} at which the normal is parallel to $\boldsymbol{r}(s_i)$. If no such point s_{i+1} exists, stop. Otherwise, move t from t_i to t_{i+1} where $N(t_{i+1}) + N(s_{i+1}) = 0$. The iteration continues until s_i and t_i converge to a pair of antipodal points, as in the figure, or it reaches s_a and t_a, in which case no antipodal points.

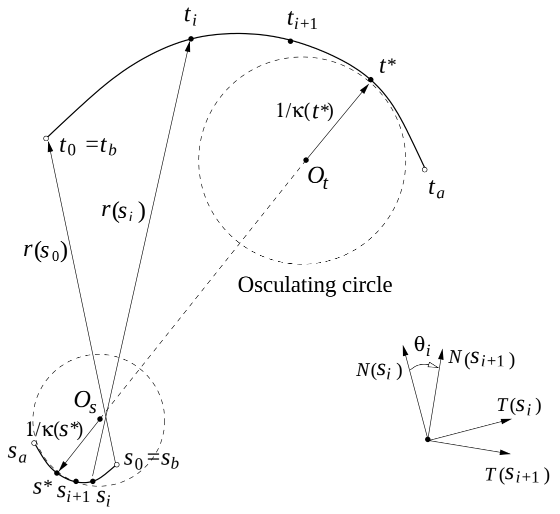

Fig. 6. Geometry at a pair of antipodal points s^* and t^*: $\theta'(s^*) < 0$ and $\|\boldsymbol{r}(s^*)\| > \frac{1}{\kappa(s^*)} + \frac{1}{\kappa(t^*)}$

It can be shown by induction [14] that the sequence $s_0, s_1, \ldots$, generated above according to $N(s_{i+1}) \times \boldsymbol{r}(s_i) = 0$ is *strictly decreasing* and no antipodal point exists on $[s_i, s_b) = \cup_{k=1}^{i} [s_k, s_{k-1})$ for all $i > 0$. The monotone sequence $\{s_i\}$ is bounded from below by s^*, the closest antipodal point to s_b if it exists. So it must converge to some $\xi \in (s_a, s_b)$ where $N(\xi) \times \boldsymbol{r}(\xi) = 0$. Hence $\xi = s^*$.

Next, we show that the local convergence rate of the sequence is *linear*. It suffices to establish that $0 < \frac{ds_{i+1}}{ds}(s^*) = f'(s^*) < 1$. As shown in Fig. 6, $\sin\theta_i = \sin(\theta(s_i)) = N(s_i) \times N(s_{i+1})$. We can derive $f'(s_i)$ by differentiating both sides of this equation with respect to s_i, and in particular,

$$f'(s^*) = \frac{\theta'(s^*)}{\kappa(s^*)} + 1 = \frac{\kappa(s^*) + \kappa(t^*)}{\kappa(s^*)\kappa(t^*)\|\boldsymbol{r}(s^*)\|} > 0,$$

where t^* on $\mathcal{T}$ is the opposite point of s^*. The march starts at s_b where $\theta(s_b) < 0$ and never passes s^*. So $\theta'(s^*) < 0$ must hold. Hence $f'(s^*) < 1$.

When $\theta(s_a) > 0$ and $\theta(s_b) > 0$, the march starts at s_a and t_a and moves toward s_b and t_b. The result on convergence still holds.

A Convex Segment and a Concave Segment We only consider the case that $\mathcal{S}$ is convex and $\mathcal{T}$ is concave as the other case is symmetric. We first determine if one of the rays extending the normals $N(t_a)$ and $N(t_b)$ intersects $\mathcal{S}$. Under conditions (i)–(v), this can be done by checking cross products. If neither ray intersects $\mathcal{S}$, then no antipodal points exist (a proof is given in [14]).

When either the ray of $N(t_a)$ or the ray of $N(t_b)$ intersects $\mathcal{S}$, we carry out the following march. Start at $s_0 = s_a$ and $t_0 = t_a$ if the ray of $N(t_a)$

intersects $\mathcal{S}$, or at $s_0 = s_b$ and $t_0 = t_b$ if the ray of $N(t_b)$ intersects $\mathcal{S}$. In each round, s_{i+1} is generated as the intersection of the ray of $N(t_i)$ and $\mathcal{S}$ and t_{i+1} is generated as its opposite point.

The sequence $\{s_i\}$ is shown to be *strictly increasing* and will converge at *linear* rate to the first antipodal point s^* from s_a if it exists.

Theorem 3. *Let $\mathcal{S}$ and $\mathcal{T}$ be two curve segments satisfying conditions (i)-(v). Suppose the two antipodal angles $\theta(s_a)$ and $\theta(s_b)$ at the endpoints s_a and s_b of $\mathcal{S}$ have the same sign. Then the two iterative strategies described above satisfy the following:*

1. *When no antipodal points exist, the iteration starting at a pair of opposite endpoints of $\mathcal{S}$ and $\mathcal{T}$ will terminate at the other pair of endpoints.*
2. *Otherwise, the iteration will converge at* linear rate *to a pair of antipodal points s^* and t^* that is the closest to the two endpoints at which the iteration starts. Furthermore, $\theta'(s^*) < 0$ if $\mathcal{S}$ and $\mathcal{T}$ are both convex and $\theta'(s^*) > 0$ if one of them is concave.*

4.4 Finding All Pairs of Antipodal Points

Once the first pair of antipodal points s^* and t^* has been found, other pairs, if exist, can be found by alternating marching with bisection recursively.

To illustrate, suppose $\theta(s_a), \theta(s_b) < 0$ and $\mathcal{S}$ and $\mathcal{T}$ are both convex. Then s^* and t^* have been found by a march from s_b. So we already know that no antipodal points exist in (s^*, s_b). That $\theta'(s^*) < 0$ and $\theta(s^*) = 0$ implies $\theta(s^* - \epsilon) > 0$ for small enough $\epsilon > 0$. Therefore the interval $(s_a, s^* - \epsilon)$ contains at least one antipodal point, which can be found using bisection.

Suppose $\theta(s_a)$ and $\theta(s_b)$ have different signs. Then s^* and t^* have been found through bisection. So $\theta(s^* - \epsilon)$ has the sign of $\theta(s_a)$ while $\theta(s^* + \epsilon)$ has the sign of $\theta(s_b)$. Then we march over the intervals $(s_a, s^* - \epsilon)$ and $(s^* + \epsilon, s_b)$ separately to search for possible antipodal points.

Figure 7 illustrates the recursive procedure as applied on an ellipse.

5 Preprocessing, Generalization, and Implementation

Curve $\boldsymbol{\alpha}$ is preprocessed in four steps to generate pairs of segments that satisfy conditions (i)–(v). These steps are illustrated on an example in Fig. 8. To locate antipodal points on $\boldsymbol{\alpha}$, we perform all four steps of preprocessing to generate segment pairs that satisfy conditions (i)–(v). Then find antipodal points on each pair[5] as described in Sect. 4.

We have implemented all algorithms in C++ for arbitrary-speed curves. The eight pairs of antipodal points in Fig. 5(a) has all three combinations of curvature signs. Figure 9 displays the antipodal points found on three

[5] For each pair, we also report opposing endpoints that are antipodal.

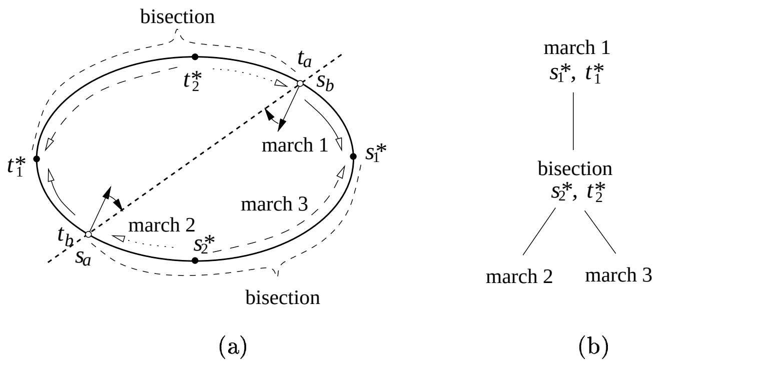

Fig. 7. (a) Two pairs of antipodal points (s_1^*, t_1^*) and (s_2^*, t_2^*) on an ellipse; and (b) the recursion tree in search for them. Here $\boldsymbol{\alpha}(s_b) = \boldsymbol{\alpha}(t_a)$ and $N(s_a) = -N(t_a)$. The recursion always ends at marching

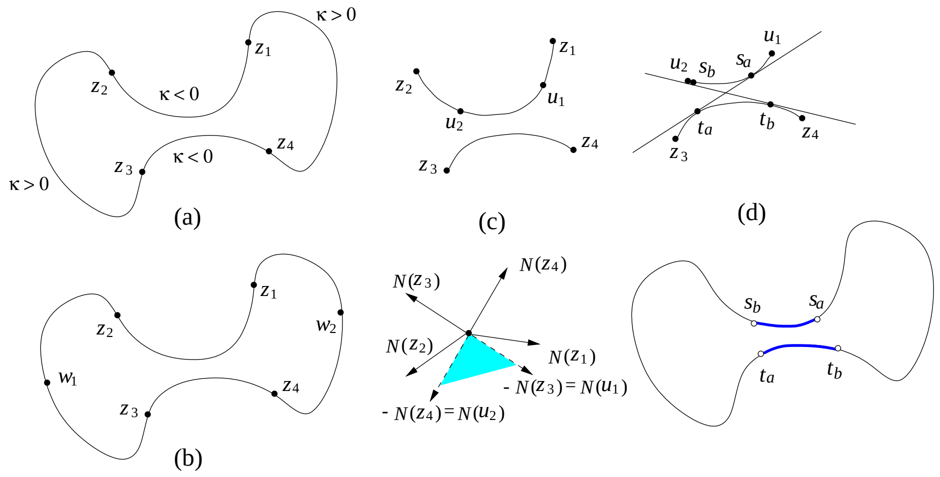

Fig. 8. Four preprocessing steps illustrated on a closed curve. (a) Find all four inflection points z_1, z_2, z_3, and z_4. (b) Split the segments $[z_2, z_3]$ and $[z_4, z_1]$ with total curvatures beyond $[-\pi, \pi]$ at the points w_1 and w_2, respectively. Carry out the next two steps on every pair of segments, for instance, $[z_1, z_2]$ and $[z_3, z_4]$. (c) Intersect the cone of inward normals over $[z_1, z_2]$ with the cone of outward normals over $[z_3, z_4]$. Now (u_1, u_2) and (z_3, z_4) satisfy conditions (i)-(iv) but not (v). (d) Extract the portions defined by the tangency points s_a, t_a, s_b, and t_b of their two common tangents

other shapes [30]: (a) an elliptic lemniscate $\rho = \sqrt{6^2 \cos^2 \phi + 3^2 \sin^2 \phi}$; (b) a limaçon $\rho = 4 + \frac{5}{2} \cos \phi$; and (c) a curve with convexities $\rho = 9/(2 + \frac{2}{5} \cos 4\phi)$.

Let n be the number of inflection points and m the number of pairs of antipodal points. There are $O(n^2)$ pairs of segments after the preprocessing.

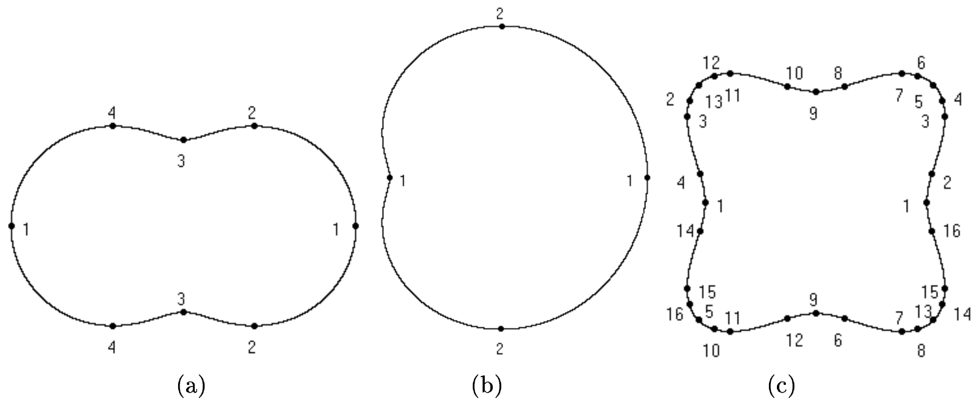

Fig. 9. Antipodal points on three parametric curves

The total number of calls to the marching and bisection procedures in Sect. 4 is $O(n^2 + m)$. This bound is indeed tight since a quadratic number of pairs of segments dissected at inflection points can have at least one pair of antipodal points. An example is the curve $\rho(\phi) = \frac{p}{1+\epsilon \cos m\phi}$ with $p > 0$, $0 < \epsilon < 1$, and integer $m > 1$. In Fig. 9(c), $p = \frac{9}{2}$, $m = 4$, and $\epsilon = \frac{1}{5}$.

6 Discussion

We have presented a collection of algorithms that compute geometric substructures on a simple, closed, and twice continuously differentiable curve. These substructures include curve segments of specified length and total curvature, common tangents, and antipodal points. Based on the monotonicity of dissected curve segments and local geometry, various marching strategies (interleaved with bisection) are employed.

A conventional nonlinear programming approach, inherently local, would rely heavily on initial guesses of antipodal positions. It would be slow and not guarantee to always find antipodal points, not to mention all of them. Our results demonstrate that computational efficiency and completeness can be achieved by taking advantage of both global and differential geometry.

The algorithms can be extended in a straightforward way to curves that are piecewise twice continuously differentiable. Extension of the algorithms to a curved surface in 3D will likely hinge on how efficiently the surface can be partitioned into patches on which a search or an objective function assumes similar monotonicity.

Both the localization algorithm and the algorithm for computing antipodal points step over a found segment for more segments or a pair of antipodal points for more pairs. The completeness is thus subject to the numerical resolution. Marching in localization is also subject to the resolution although marching in antipodal point computation is not.

The design of these algorithms based on dissection suggests a measure of the "combinatorial size" of a curve by the number of inflections. The algorithm for finding antipodal points employs several specialized numerical primitives to complete a basic operation such as moving a point on the curve or determining if a local geometric condition holds. Such a routine call usually completes in tens to hundreds of iteration steps. It seems reasonable to analyze the asymptotic running time of a curve processing algorithm by the number of calls to such numerical primitives.

Acknowledgement Support for this research was provided in part by Iowa State University, and in part by the National Science Foundation through a CAREER Award IIS-0133681. The author would like to thank David Persky for helping with the implementation, David Kriegman for pointing out the possibility of globally solving low-dimensional problems without resorting to optimization, and Ravi Janardan for suggesting to analyze the presented algorithms. He is also very grateful to the anonymous reviewers of this paper as well as of the earlier results [13,16] for providing valuable feedback that has helped improve the structure and presentation of the paper.

References

1. P. K. Allen and K. S. Roberts. Haptic object recognition using a multi-fingered dextrous hand. *Proc. IEEE Intl. Conf. Robot. and Automat.* pp. 342–347, 1989.
2. A. Bicchi. Hands for dextrous manipulation and robust grasping: a difficult road toward simplicity. *IEEE Trans. Robot. and Automation*, 16(6):652-662, 2000.
3. A. Blake and M. Taylor. Planning planar grasps of smooth contours. *Proc. IEEE Intl. Conf. Robot. and Automation*, pp. 834–839, 1993.
4. C. Cai and B. Roth. On the spatial motion of a rigid body with point contact. *Proc. IEEE Intl. Conf. Robot. and Automat.*, pp. 686–695, 1987.
5. B. Chazelle, H. Edelsbrunner, L. J. Guibas, and M. Sharir. Diamester, width, closest line pair and parametric searching. *Disc. & Comp. Geometry*, 10:183–196, 1993.
6. I-M. Chen and J. W. Burdick. Finding antipodal point grasps on irregularly shaped objects. *Proc. IEEE Intl. Conf. Robot. and Automation*, pp. 2278–2283, 1992.
7. K. L. Clarkson and P. W. Shor. Applications of random sampling in computational geometry, II. *Disc. & Comp. Geometry*, 4:387–421, 1989.
8. M. Erdmann. Shape recovery from passive locally dense tactile data. In P. K. Agarwal *et al.*, ed., *Robotics: The Algorithmic Perspective*, pp. 119–132. A. K. Peters, Boston, MA, 1998.
9. M. A. Fischler and H. C. Wolf. Locating perceptually salient points on planar curves. *IEEE Trans. Pattern Analysis and Machine Intell.*, 16(2):113–129, 1994.
10. T. N. T. Goodman. Inflections on curves in two and three dimensions. *Comp. Aided Geom. Design*, 8(1):37–50, 1991.

11. B. Grunbaum. A proof of Vazsonyi's conjecture. *Bulletin of Research Council Israel*, 6(A):77–78, 1956.
12. J. Hong, G. Lafferriere, B. Mishra, and X. Tan. Fine manipulation with multifinger hands. *Proc. IEEE Intl. Conf. Robot. and Automation*, pp. 1568–1573, 1990.
13. Y.-B. Jia. Localization on curved objects using tactile information. *Proc. IEEE/RSJ Intl. Conf. Robots and Systems*, pp. 701-706, 2001.
14. Y.-B. Jia. Geometry and computation of antipodal points on plane curves. Tech. Report ISU-CS-01-04, Computer Science Department, Iowa State University, Ames, IA, 2001. `http://www.cs.iastate.edu/~jia/papers/isu-cs-01-04.pdf`.
15. Y.-B. Jia. Localization and grasping of curved objects using tactile information. Tech. Report ISU-CS-01-08, Computer Science Department, Iowa State University, Ames, IA, 2001. `http://www.cs.iastate.edu/~jia/papers/isu-cs-01-08.pdf`.
16. Y.-B. Jia. Curvature-based computation of antipodal grasps. In *Proc. IEEE Intl. Conf. Robot. and Automation*, pp. 1571-1577, 2002.
17. D. Manocha and J. F. Canny. Detecting cusps and inflection points in curves. *Comp. Aided Geom. Design*, 9(1):1–24, 1992.
18. X. Markenscoff, L. Ni, and C. H. Papadimitriou. The geometry of grasping. *Intl. J. Robot. Res.*, 9(1):61–74, 1990.
19. J. Matoušek and O. Schwarzkopf. A deterministic algorithm for the three-dimensional diameter problem. *Comp. Geometry: Theory and Applications*, 6:253–362, 1996.
20. B. Mishra, J. T. Schwartz, and M. Sharir. On the existence and synthesis of multifinger positive grips. *Algorithmica*, 2(4):541–558, 1987.
21. F. Mokhtarian and A. Mackworth. Scale-based description and recognition of planar curves and two-dimensional shapes. *IEEE Trans. Pattern Analysis and Machine Intell.*, 8(1):34–43, 1986.
22. M. Moll and M. A. Erdmann. Reconstructing shape from motion using tactile sensors. *Proc. IEEE/RSJ Intl. Conf. Intell. Robots and Systems*, pp. 692–700, 2001.
23. D. J. Montana. The kinematics of contact and grasp. *Intl. J. Robot. Res.*, 7(3):17–32, 1988.
24. B. O'Neill. *Elementary Differential Geometry.* Academic Press, Inc., 1966.
25. J. Ponce, D. Stam, and B. Faverjon. On computing two-finger force-closure grasps of curved 2D objects. *Intl. J. Robot. Res.*, 12(3):263–273, 1993.
26. F. P. Preparata and S. J. Hong. Convex hulls of finite sets of points in two and three dimensions. *Communications of the ACM*, 20(2):87–93, 1977.
27. F. P. Preparata and M. I. Shamos. *Computational Geometry: an Introduction.* Springer-Verlag, 1985.
28. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T.Vetterling. *Numerical Recipes in C.* Cambridge Univ. Press, Inc., 1988.
29. M. Sakai. Inflection points and singularities on planar rational cubic curve segments. *Comp. Aided Geom. Design*, 16:149–156, 1999.
30. E. V. Shikin. *Handbook and Atlas of Curves.* CRC Press, Inc., 1995.
31. J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis.* Springer-Verlag, 2nd edition, 1993.
32. A. C. Yao. On constructing minimum spanning trees in k-dimensional space and related problems. *SIAM Journal of Computing*, 11(4):721–736, 1982.