Appeared in *Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems.*

Localization on Curved Objects Using Tactile Information*

Yan-Bin Jia

Department of Computer Science
Iowa State University
Ames, IA 50011-1040
jia@cs.iastate.edu

Abstract

This paper offers a computational study of finger localization on 2-D curved objects using tactile data which builds on efficient numerical processing of curves.

Our first algorithm localizes one rolling finger on a stationary object. It finds all boundary segments with the same arc length and total curvature computed from tactile data. The algorithm slides an imaginary segment along the object boundary by alternatively marching its two endpoints forward, stretching or contracting the segment if necessary. Through a curvature-based analysis we establish the global convergence of the algorithm to every location of such a segment and also derive the local convergence rate. The algorithm runs in time linear in the size of the discretized boundary curve domain, which is asymptotically as fast as computing the object's perimeter through numerical integration.

Based on the above results, we then present a global algorithm to localize two fingers rolling on a free object. This has considerably improved over our previous local algorithm [6] using a least-squares formulation. The algorithm partitions the object boundary into segments over which related total curvature functions are monotonic. Then it combines bisection with forward marching to search for possible locations of the fingers within every pair of such segments.

1 Introduction

A robot grasping strategy often relies on knowledge of the exact placement of the fingers on an object. The use of tactile information combined with object geometry can facilitate the localization of the fingers relative to the object. In this paper, we investigate how one or two fingers with tactile capability can actively determine their locations on a curved object from a few contact points recorded on the fingertips.

The action taken by the fingers is rolling on the object, which is bounded by a known parametric 2-D curve with nontrivial curvature undefined at no more than a few isolated points. The kinematics of rolling together with recorded finger contacts will reduce localization to identifying curve segments on the object boundary that meet geometric requirements on arc length and total curvature.

In Section 2 we study a simple version of localization with one disk-like finger rolling on a stationary curved object. We will present a numerical algorithm with provable behavior. In Section 3 we move on to the problem of localizing two rolling fingers on a free object. The problem was studied earlier in [6] but only partially solved by a least-squares method. Built on the results in Section 2, a global algorithm is presented to find locations of both rolling fingers. Simulations will be presented in Section 4, followed by further discussions in Section 5.

1.1 Related Work

This work is founded on contact kinematics which describe the motion of a point contact between two rigid bodies by a set of differential equations [11, 2]. The special kinematics of rolling motion allow finger localization to be formulated as a purely geometric problem.

Grimson and Lozano-Pérez [5] used tactile measurements of positions and surface normals for recognition and localization of 3D polyhedra. Kriegman and Ponce [8] applied elimination theory to match curved three-dimensional objects with the shape of image contours through fitting. Rimon and Blake [12] showed how to "cage" a 2-dimensional object using two fingers but only one degree of control.

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. Erdmann [3] showed that the local geometry of an object with known angular velocity can be recovered by two passive linear tactile sensors. Extending this work, Moll and Erdmann [10] 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.

Fischler [4] described an algorithm to locate points with

*Support for this research was provided by Iowa State University.

extreme curvatures on planar curves and reconstruct the original curves based on these points. Mokhtarian and Mackworth [9] used inflection points (where curvatures are zero) for planar curve descriptions and matching.

In the author's recent work [6], the problem of localizing two rolling fingers on a free object was presented with a least-squares solution. Nevertheless, the solution method guaranteed neither convergence nor completeness and was slow. In Section 3, we will present a global numerical algorithm to solve the same problem.

2 Localizing on a Stationary Object

We first look at how to determine the location of a finger rolling on a stationary object. Both the finger and the object can be of any shape as long as they maintain point contact. Let the object be bounded by a regular curve $\boldsymbol{\alpha}(s)$, where s is the location of contact. To simplify the analysis, we assume the finger to be a disk described by $r(\cos u, \sin u)$, where u locates the contact on the disk and can be detected by its tactile sensor. See Figure 1.

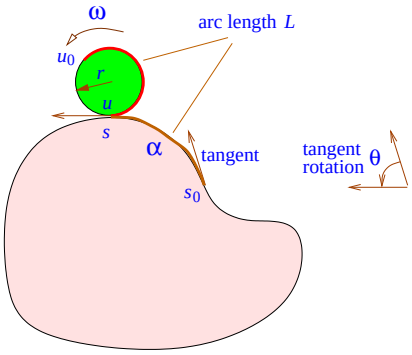

Figure 1: A disk rolling on a stationary curved object.

Contact kinematics [11, 2] give the velocities of contact:

$$\dot{u} = \frac{du}{d\tau} = -\frac{\omega}{1 + r \cdot \kappa(s)}, \quad (1)$$

$$\dot{s} = \frac{ds}{d\tau} = \frac{\omega}{\|\boldsymbol{\alpha}'(s)\| \left(1/r + \kappa(s)\right)}, \quad (2)$$

where τ is time, ω the angular velocity of the disk, and $\kappa(s)$ the curvature of $\boldsymbol{\alpha}$ at s. From (1) and (2), we obtain $r\dot{u} = -\dot{s}\,\|\boldsymbol{\alpha}'(s)\|$. The length L of $\boldsymbol{\alpha}$ over $[s_0, s_1]$ is $-(u_1 - u_0)r$, hence known from the tactile readings. Plugging this equality into (1) yields

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

Integrate this equation over the time period $[0, \tau]$:

$$u_1 - u_0 + \int_0^\tau \omega(\xi)d\xi = \int_{s_0}^{s_1} \kappa(v)\|\boldsymbol{\alpha}'(v)\|\,dv. \quad (3)$$

Here $\Phi(s_0, s_1) = \int_{s_0}^{s_1} \kappa(v)\|\boldsymbol{\alpha}(v)\|\,dv$ is the *total curvature* of $\boldsymbol{\alpha}$ over $[s_0, s_1]$. It gives the amount of rotation of the unit tangent $T = \boldsymbol{\alpha}'/\|\boldsymbol{\alpha}\|$ as it moves from s_0 to s_1.

The amount of rotation $\int_0^\tau \omega(\xi)d\xi$ of the disk is known. Thus the total curvature θ over $[s_0, s_1]$ is also known according to (3). The locations s_0 and s_1 of the disk on $\boldsymbol{\alpha}$ can then be solved from the following equations:

$$\ell(s,t) = \int_s^t \|\boldsymbol{\alpha}'(\xi)\|\,d\xi = L, \quad (4)$$

$$\Phi(s,t) = \int_s^t \kappa(\xi)\|\boldsymbol{\alpha}'(\xi)\|\,d\xi = \theta. \quad (5)$$

Geometrically, the problem is to *locate a curve segment with length L and total curvature θ.*

Often the integral ℓ has no closed form and needs to be evaluated numerically. The integral $\Phi(s,t)$ has the closed form $\arccos(T(s) \cdot T(t))$ if it is within $(-\pi, \pi)$ and otherwise cannot be determined from $T(s)$ and $T(t)$ alone. We need to look for numerical solutions of equations (4) and (5).

2.1 Convex Boundary Curve

We begin with the case that the boundary curve $\boldsymbol{\alpha}$ is convex. Below we present a *marching* algorithm that finds all curve segments on the boundary with length L and total curvature θ. For clarity of presentation, we assume that $\boldsymbol{\alpha}$ is unit-speed. But the results (including the convergence rate) will extend to any regular parameterization. Also we assume $\kappa(s) > 0$ for all s. Extension to $\kappa(s) \geq 0$ will not be difficult.

Define the function $\rho(s)$ such that $\Phi(s, s + \rho(s)) = \theta$. Namely, $\rho(s)$ is the length of the curve segment starting at s over which the tangent rotates by the angle θ. The algorithm starts at location $s_0 = 0$ and generates two sequences $s_0, s_1, s_2, \ldots$ and $t_0, t_1, t_2, \ldots$ under the following rules:

Case 1 $\rho(s_0) > L$:
$$t_i = s_i + \rho(s_i);$$
$$s_{i+1} = t_i - L.$$

Case 2 $\rho(s_0) < L$:
$$t_i = s_i + L;$$
$$\Phi(s_{i+1}, t_i) = \theta.$$

Figure 2 illustrates the working of the algorithm in these two cases. For a non-unit speed curve, t_i and s_{i+1} are obtained from t_{i-1} and s_i through numerical integration.

We first study the behavior of the algorithm in Case 1 and establish its convergence. Define the function $\phi(s) = \Phi(s, s + L)$ to measure the total curvature over a segment of length L that begins at s. Let $a > 0$ be a solution to $\Phi(s, s + L) = \theta$ such that every other solution $c \geq s_0$ implies $c > a$. So a is the first feasible starting point of the curve segment. Let $b = a + L$ be its endpoint. It is easy to

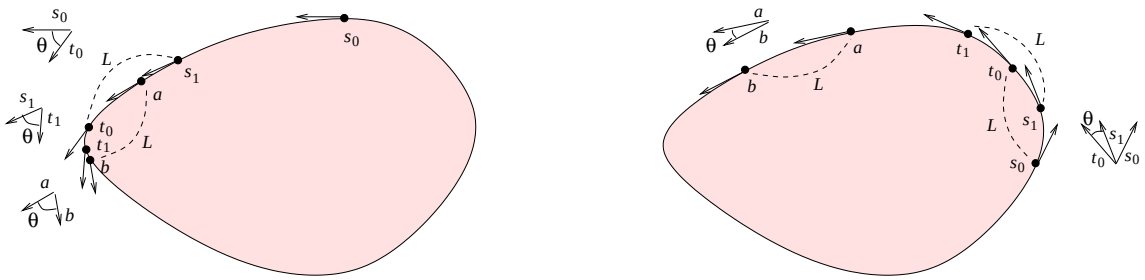

Figure 2: Two cases of marching s and t: **(a)** $\Phi(s_i, t_i) = \theta$ but $\ell(s_i, t_i) > L$; **(b)** $\ell(s_i, t_i) = L$ but $\Phi(s_i, t_i) > \theta$.

show by induction the following Lemma (a proof is given in [7]):

Lemma 1 *In Case 1, $s_i < s_{i+1} < a$ holds for all $i \geq 0$.*

The above lemma establishes that the two sequences $s_0, s_1, \ldots$ and $t_0, t_1, \ldots$ are monotonic and bounded by a and b, respectively. Hence they converge to, say, s^* and t^* where $\Phi(s^*, t^*) = \Phi(s^*, s^* + L) = \theta$. Therefore $s^* = a$ and $t^* = b$ by definition of a and b. The next lemma gives the local convergence rate.

Lemma 2 *Suppose $\phi(0) < \theta$. Then $\kappa(b) \geq \kappa(a)$, where $\kappa(a) = \kappa(b)$ holds if and only if $\phi'(a) = 0$. When $\kappa(b) > \kappa(a)$, the algorithm has linear convergence rate given by a factor of $\kappa(a)/\kappa(b)$.*

Full proof of the above lemma is given in [7].

Similarly, in Case 2, $\phi'(a) \leq 0$. It follows that $\kappa(a) \geq \kappa(b)$ where the equality holds if and only if $\phi'(a) = 0$. The local convergence rate in this case is linear given by $\kappa(b)/\kappa(a)$ when $\kappa(a) > \kappa(b)$.

Proposition 3 *The algorithm converges to the first feasible segment satisfying (4) and (5) in the marching direction.*

To find the next segment on $\boldsymbol{\alpha}$, we reset s_0 to be the sum of a very small positive amount and s_i for large enough i, and repeat the same procedure. Both s and t move along the object boundary no more than once at step size h. So at most $2T/h$ steps are performed in all numerical integrations. The number of numerical steps for obtaining t_0 is $m \leq T/h$. Hence the algorithm performs at most $2T/h + m$ numerical increments.

Theorem 4 *The marching algorithm locates all curve segments with length L and total curvature θ on a non-degenerate closed simple curve $\boldsymbol{\alpha}$ up to numerical resolution in no more than $\Theta(T/h)$ numerical steps.*

2.2 Non-Convex Boundary Curve

In this section we extend the localization procedure in Section 2.1 to an arbitrary-speed closed simple curve $\boldsymbol{\alpha}$. The correctness of that procedure 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 $s < t$. This is no longer true everywhere when $\boldsymbol{\alpha}$ has concavities. For example, if $\kappa(s) < 0$, then $\partial\Phi/\partial s > 0$.

We march the two endpoints s and t of a hypothesized curve segment counterclockwise along $\boldsymbol{\alpha}$. There are 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 of the four modes, the hypothesized curve segment slides along $\boldsymbol{\alpha}$ until an inflection point[1] 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 one of s and t and simultaneously keeping track where the other should be (without actually increasing it). Figure 3 illustrates the operations in the modes convex-convex and concave-convex.

Bisection will be invoked in the modes convex-concave and concave-convex. In preprocessing, we compute all points of inflection; and then in $\Theta(T/h)$ time we evaluate arc lengths $\ell(z_1, z_i)$ and total curvatures $\Phi(z_1, z_i)$, for $i = 1, \ldots, n$, and $\ell(z_n, z_1 + T)$ and $\Phi(z_n, z_1 + T)$. For a detailed description of all operations, we refer the reader to [7], where the total number of numerical increments is shown to be at most $5T/h$.

Theorem 5 *All segments with length L and total curvature θ can be found on a closed simple curve defined on $[0, T]$ up to numerical resolution in $\Theta(T/h)$ steps.*

We can directly modify the marching algorithm to find all stationary points of the total curvature function $\Phi(s,t)$ over any segment of length L along a closed curve. It is

[1] A simple point of inflection satisfies $\kappa = 0$ but $\kappa' \neq 0$.

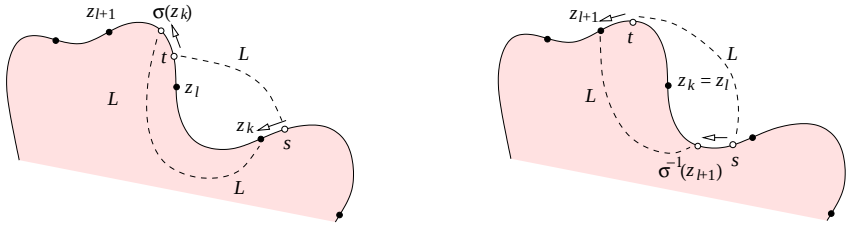

Figure 3: Two of the four modes of the localization algorithm: **(a)** convex-convex, where $\kappa(s) > 0$ and $\kappa(t) > 0$, and its following mode **(b)** concave-convex, where $\kappa(s) < 0$ and $\kappa(t) > 0$. The function $\sigma(s)$ determines the ending point of a curve segment on $\boldsymbol{\alpha}$ of length L that starts at s. In both (a) and (b), z_k corresponds to the first inflection point after s, and z_l the last inflection point before t. In (a), t advances to $\min(\sigma(z_k), z_{l+1}) = \sigma(z_k)$ and s advances accordingly to z_k. In (b), s advances to $\min(z_k, \sigma^{-1}(z_{l+1})) = \sigma^{-1}(z_{l+1})$ and t advances accordingly to z_{l+1}. The next mode is concave-concave.

not hard to show that such a stationary point satisfies condition (4) in addition to the following condition:

$$\kappa(t) - \kappa(s) = \int_s^t \kappa'(u)\, du = 0. \tag{6}$$

The roles of $\Phi(s,t)$, κ, and inflections in the marching algorithm are now replaced by $\kappa(t) - \kappa(s)$, κ', and vertices in the modified version. And the preprocessing involves computing all *simple vertices*[2] of the curve.

3 Localizing on a Movable Object

Now we consider that the object is no longer stationary. It moves in response to the disk rolling. We let a second identical disk with tactile capability rolling on the object, as shown in Figure 4. Let $\omega_{\mathcal{B}}$ and $\bar{\omega}$ be the angular velocities of the object and the second disk, respectively. The contact between the object and the second disk are determined by the parameters v and t, respectively. The kinematics of both contacts are almost the same as (1) and (2) except the numerators need to be replaced with the relative angular velocity $\omega - \omega_{\mathcal{B}}$ and $\bar{\omega} - \omega_{\mathcal{B}}$. We can eliminate the object's angular velocity $\omega_{\mathcal{B}}$ [6]:

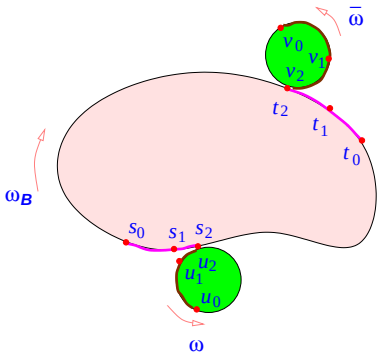

Figure 4: Two fingers rolling on one object.

$$\dot{u} - \kappa(s)\,\|\alpha'(s)\|\,\dot{s} + \omega = \dot{v} - \kappa(t)\,\|\alpha'(t)\|\,\dot{t} + \bar{\omega}. \tag{7}$$

Let the two disks record contact positions s_i, t_i simultaneously at three time instants τ_0, τ_1, τ_2. Integrate equation (7) over $[\tau_0, \tau_1]$ and $[\tau_0, \tau_2]$ yields

$$\Phi(s_0, s_1) - \Phi(t_0, t_1) = \theta_1, \tag{8}$$
$$\Phi(s_0, s_2) - \Phi(t_0, t_2) = \theta_2. \tag{9}$$

Here θ_i is computed from sensor data u_0 and u_i, control data ω and $\bar{\omega}$, and time τ_0 and τ_i. Moreover, the arc lengths $L_1 = \ell(s_0, s_1)$, $L_2 = \ell(s_0, s_2)$, $L_3 = \ell(t_0, t_1)$, and $L_4 = \ell(t_0, t_2)$ are known.

Geometrically, localization is equivalent to finding two points on $\boldsymbol{\alpha}$ at which a) the two segments of lengths L_1 and L_3, respectively, differ by θ_1 in total curvature, and b) the two segments of lengths L_2 and L_4, respectively, differ by θ_2 in total curvature. In the below, we will present a global numerical algorithm that finds all pairs s_0 and t_0 satisfying equations (8) and (9).

3.1 Domain Partitioning

Define the function $\phi_i(s)$ as the total curvature of a segment of the boundary curve $\boldsymbol{\alpha}$ starting at s and having length L_i. The stationary points of ϕ_1 and ϕ_2, found by the modified marching algorithm in the end of Section 2.2, divide the curve domain into n intervals $[a_i, a_{i+1}]$, $i = 0, ..., n-1$ and $a_n = a_0 + T$. The values of ϕ_1 and ϕ_2 at all a_i are computed with one round of integrations of $\|\boldsymbol{\alpha}'\|$ and κ along $\boldsymbol{\alpha}$ in time $\Theta(T/h)$. Similarly, the stationary points of ϕ_3 and ϕ_4 divide $[0, T]$ into m intervals $[b_i, b_{i+1}]$, $i = 0, ..., m-1$ and $b_m = b_0 + T$, and ϕ_3 and ϕ_4 at all b_j are evaluated.

[2] A simple vertex of a curve satisfies $\kappa' = 0$ but $\kappa'' \neq 0$.

Within each interval $[a_i, a_{i+1}]$, both functions ϕ_1 and ϕ_2 increase or decrease monotonically. Similarly, the function ϕ_3 and ϕ_4 are also monotonic within each interval $[b_j, b_{j+1}]$. The localization algorithm enumerates all nm pairs of intervals $[a_i, a_{i+1}]$ and $[b_j, b_{j+1}]$. It determines if each pair contains feasible starting points of the curve segments, and numerically finds them if so.

3.2 A Pair of Monotonic Intervals

The idea is to use the one-to-one correspondence between s and t as defined by equation (8) or (9). We ensure that one of them, say (8), is always satisfied while moving s within the interval $[a_i, a_{i+1}]$ and t (accordingly) within the interval $[b_j, b_{j+1}]$.

Movement of s results in movements of the endpoints of the two segments of length L_1 and L_2 that start at s. This is done through numerically integrating $\|\boldsymbol{\alpha}'\|$. Updates on their total curvatures $\phi_1(s)$ and $\phi_2(s)$ are performed along the way. Similar updates on $\phi_3(s)$ and $\phi_4(s)$ are performed as a result of the movement of t.

Let I_i be the interval defined by $\phi_1(a_i)$ and $\phi_1(a_{i+1})$. And let J_j be the interval defined by $\phi_3(b_j) + \theta_1$ and $\phi_3(b_{j+1}) + \theta_1$. If $I_i \cap J_j = \emptyset$, then the pair $[a_i, a_{i+1}]$ and $[b_j, b_{j+1}]$ can be excluded from consideration.

Otherwise, we first determine the maximal subintervals of $[a_i, a_{i+1}]$ with endpoints x, y where $\phi_1(x) < \phi_1(y)$ and of $[b_i, b_{i+1}]$ with endpoints w, z where $\phi_3(w) < \phi_3(z)$ such that $\phi_1(x) = \phi_3(w) + \theta_1$ and $\phi_1(y) = \phi_3(z) + \theta_1$. There are a number of cases, one of which is illustrated in Figure 5.

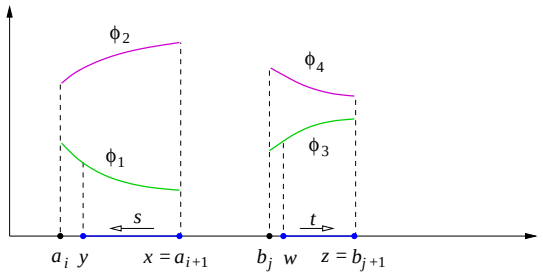

Figure 5: One-to-one correspondence between s and t defined by $\phi_1(s) - \phi_3(t) = \theta_1$ as s decreases from x to y while t increases from w to z . Since $\phi_1(a_i) > \phi_1(a_{i+1})$ and $\phi_3(b_j) < \phi_3(b_{j+1})$, start with $s \leftarrow a_{i+1}$ and $t \leftarrow b_j$. Suppose $\phi_1(a_{i+1}) > \phi_3(b_j) + \theta_1$. Increase t until $\phi_1(a_{i+1}) = \phi_3(t) + \theta_1$ and then let $w \leftarrow t$ and $x \leftarrow a_{i+1}$. Meanwhile, suppose $\phi_1(a_i) > \phi_3(b_{j+1}) + \theta_1$. Let $s \leftarrow a_i$ and increase s until $\phi_1(s) = \phi_3(b_{j+1}) + \theta_1$. Let $y \leftarrow s$ and $z \leftarrow b_{i+1}$.

As s changes from x to y, t changes (accordingly) from w to z. We first consider that one of $\phi_2(s)$ and $\phi_4(t)$ increases and the other decreases. Without loss of generality, suppose $\phi_2(s)$ increases as s changes from x to y while $\phi_4(t)$ decreases as t changes from w to z. Then $\phi_2(s) - \phi_4(t)$ increases. A unique pair of curve locations exist if $\phi_2(x) - \phi_4(w) < \theta_2$ and $\phi_2(y) - \phi_4(z) > \theta_2$. The locations can be found using bisection.

Otherwise, both $\phi_2(s)$ and $\phi_4(t)$ increase or both decrease. We employ an iterative procedure as illustrated in Figure 6.

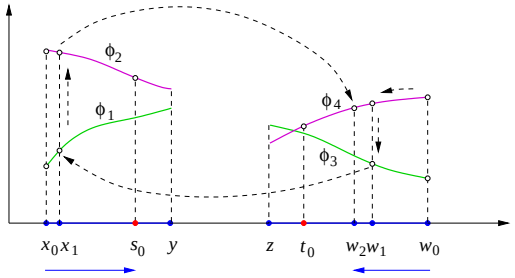

Figure 6: Iterations when both ϕ_2 and ϕ_4 decrease. Start with $s \leftarrow x_0 = x$ and $t \leftarrow w_0 = w$. Alternately move from x_i to x_{i+1} and from w_i to w_{i+1} as follows. Since $\phi_4(w_0) + \theta_2 > \phi_2(x_0)$, t decreases from w_0 to w_1 where $\phi_4(w_1) + \theta_2 = \phi_2(x_0)$. But now $\phi_3(w_1) + \theta_1 > \phi_1(x_0)$. Next, increase s from x_0 to x_1 where $\phi_1(x_1) = \phi_3(w_1) + \theta_1$. A new round starts by decreasing t from w_1 to w_2 to reestablish $\phi_4(w_2) + \theta_2 = \phi_2(x_1)$. The iterations continue until the difference between $\phi_4(w_i) + \theta_2$ and $\phi_2(x_i)$ is small enough or one of x_i and w_i exits the corresponding interval. In the former case, locations have been found. In the later case, locations do not exist.

The two intervals may contain more than one pair of feasible locations of curve segments. To find the next pair, we pass by s_i and t_i for large enough i by a small amount and continue the process.

In the worst case, the algorithm requires $\Theta((m+n)T/h)$ steps. The real running time is usually faster since most of the mn pairs I_i and J_j of intervals are rejected due to $I_i \cap J_j = \emptyset$, as we have observed in simulations.

4 Simulations

We implemented both localization algorithms in C++. Simulations of the marching algorithm in Section 2 were conducted on cubic splines, limaçons, logarithmic spirals, exponential curves, etc. In Figure 7, the disk starts rolling at a_5 on a cubic spline and stops at b_5. The localization algorithm finds six segments over $[a_i, b_i]$, $0 \le i \le 5$, respectively, that have length $\ell(a_5, b_5)$ and total curvature $\Phi(a_5, b_5)$. To eliminate the ambiguities, the disk continues rolling from b_5 to c_5 and relies on the extra information $\ell(b_5, c_5)$ and $\Phi(b_5, c_5)$.

In Figure 8, two fingers (not shown) roll from s_0 to s_1 to s_2 and from t_0 to t_1 to t_2, respectively. Here $\theta_1 = -0.725248$, and $\theta_2 = -1.99849$. The stationary points

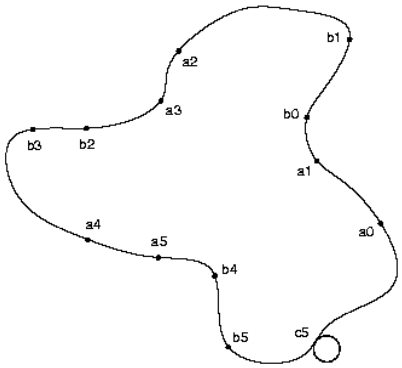

Figure 7: Simulation of the marching algorithm on a cubic spline.

of ϕ_1 and ϕ_2 divides the curve domain into 16 intervals; and the stationary points of ϕ_3 and ϕ_4 also divide the curve domain into 16 intervals. A total of 15 pairs of feasible locations for s_0 and t_0 were found by the algorithm described in Section 3. The ambiguities were eliminated with extra tactile data taken at a fourth pair of contact positions.

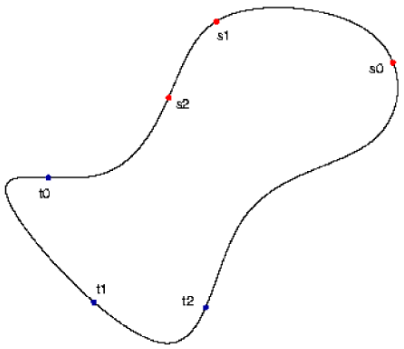

Figure 8: Localization of two curve segments.

5 Discussion

Tactile sensing in the rolling mode reduces the localization of fingers to constraint satisfaction in multiple variables. The presented algorithms exploit curve geometry to run in time *linear* in the size of the discretized curve domain. This is achieved by partitioning the curve domain into intervals monotonic with respect to related functions and then by combining bisection with marching. The completeness of these algorithms distinguish themselves from a local optimization approach (based on least squares, for instance) whose success heavily relies on initial estimates.

The marching algorithm described in Section 2 can be easily generalized to find all length-L segments on a curve, open or closed, over which the integral of a function $f(u)$ equals some constant C.

An industrial application often handles large quantities of identical parts. The precomputation of inflections and vertices is thus a one-time overhead for one part shape.

A key implementation issue is to ensure rolling contact between both fingers and the object. We are currently experimenting with one straight jaw and one passive wheel on a vertical pin.

Acknowledgement The author is grateful to the anonymous reviewers for their valuable feedback. He would also like to thank David Persky for his code used in generating cubic splines for simulation.

References

[1] P. K. Allen and K. S. Roberts. Haptic object recognition using a multi-fingered dextrous hand. *Proc. of the IEEE Intl. Conf. on Robot. and Automat.* pp. 342–347, 1989.

[2] C. Cai and B. Roth. On the spatial motion of a rigid body with point contact. *Proc. of the IEEE Intl. Conf. on Robot. and Automat.*, pp. 686–695, 1987.

[3] 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.

[4] 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.

[5] W. E. L. Grimson and T. Lozano-Pérez. Model-based recognition and localization from sparse range or tactile data. *Intl. J. Robot. Res.*, 3(3):3–35, 1984.

[6] Y.-B. Jia. Grasping curved objects through rolling. *Proc. of the IEEE Intl. Conf. on Robot. and Automat.*, pp. 377-382, 2000.

[7] Y.-B. Jia. Localization and grasping of curved objects using tactile information. Tech. Report ISU-CS-01-08, CS Dept., Iowa State University, Ames, IA, 2001.

[8] D. J. Kriegman and J. Ponce. On recognizing and positioning curved 3-D objects from image contours. *IEEE Trans. Pattern Analysis and Machine Intell.*, 12(12):1127–1137, 1990.

[9] 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.

[10] M. Moll and M. A. Erdmann. Reconstructing shape from motion using tactile sensors. *Proc. of the IEEE/RSJ Intl. Conf. on Intell. Robots and Systems*, 2001.

[11] D J. Montana. The kinematics of contact and grasp. *Intl. J. Robot. Res.*, 7(3):17–32, 1988.

[12] E. Rimon and A. Blake. Caging 2D bodies by 1-parameter two-fingered gripping systems. *Proc. of the IEEE Intl. Conf. on Robot. and Automat.*, pp. 1458–1464, 1996.