Submission to GMP 2014

High-Precision Continuous Contact Motion for Planar Freeform Geometric Curves

Submission number: 057

Abstract

This paper presents an efficient algorithm for generating a continuous precise contact motion between planar geometric models bounded by piecewise polynomial C^1-continuous parametric B-spline curves. A system of algebraic constraint equations is formulated and then efficiently solved for the two/three-contact configuration between two planar B-spline curves. The result is essentially the same as the generation of configuration space obstacle for a moving curve (with translation and rotation) against a stationary curve. The two-contact motion can be characterized as the intersection curve in the boundary of the configuration space obstacle.

The topology of the reconstructed solution is guaranteed to be correct up to a prescribed tolerance and we demonstrate the effectiveness of the proposed approach using several test examples of continuous contact motion among planar freeform smooth geometric models.

Keywords: Configuration space obstacle, contact motion, freeform geometric models, B-spline curves.

1 Introduction

The geometric concept of configuration space is one of the most important tools for reasoning about planning in robotics and automation [8, 11, 19, 20, 22, 23, 24, 28]. Nevertheless, the construction of explicit configuration spaces has been limited to those for low-dimensional cases such as translational motion in 2D or 3D and planar motion with three degrees of freedom (i.e., translation and rotation in the plane). Furthermore, in the motion planning of non-polygonal curved objects, the configuration space is usually bounded by high-degree curves and surfaces, an efficient and precise construction of which has been a non-trivial problem.

Industrial objects are mostly designed with Non-Uniform Rational B-Spline (NURBS) curves and surfaces, which are the de facto industry standard for 3D modeling. Consequently, the contact motion analysis for numerically controlled (NC) milling machines with respect to NURBS models has been one of the main research topics in computer aided design and manufacturing [14]. Because of the requirement of high-precision in manufacturing, solutions based on the polygonal approximation of NURBS models are often not acceptable. It is also a common practice that the NC tool-paths are approximated with NURBS curves in the planning stage.

In this paper, we consider a general contact motion analysis for a planar freeform smooth curve $C(u)$ with respect to a similar but static curve $D(v)$ in the plane. As the moving curve has two degrees of freedom in translation (x, y) and one degree of freedom in rotation θ, the configuration space is a three-dimensional space (i.e., the $xy\theta$-space). The moving curve can be represented as $C(u, x, y, \theta) = R_\theta(C(u)) + (x, y)$, where R_θ is the planar rotation by angle θ.

The configuration space (C-space) obstacle can be bounded by implicit surfaces of the form $F(x, y, \theta) = 0$, which correspond to the contact configurations between $C(u, x, y, \theta)$ and $D(v)$ but with no interpenetration to each other. The boundary of the C-space obstacle consists of many patches of these implicit surfaces, the exact representation of which is quite complicated. Moreover, these surface patches are bounded by intersection curves with other similar such patches. The intersection curves correspond to *two-contact* configurations between $C(u, x, y, \theta)$ and $D(v)$, i.e., the two curves contact tangentially at two different locations: $C(u_i, x, y, \theta) = D(v_i)$, for $i = 1, 2$, where $u_1 \neq u_2$ and $v_1 \neq v_2$. The intersection curve segment has two endpoints, each of which typically corresponds to a *three-contact* configuration between $C(u, x, y, \theta)$ and $D(v)$. The C-space obstacle boundary thus has a layout which can be represented in a graph structure, where each vertex corresponds to a three-contact configuration and certain pairs of these vertices are connected with edges (representing two-contact continuous motions).

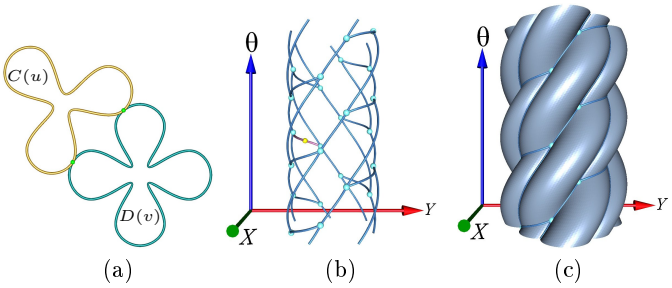

Figure 1: Continuous contact motion between two planar non-convex curves: (a) the contact motion in the work space, (b) the corresponding motion as computed in this work on the boundary of the configuration space obstacle in the $xy\theta$-space (see the pink edge in (b)) and (c) the whole boundary of the C-space obstacle. Note that the angle θ is periodic and the top of the network is continuously connected to the bottom.

Figure 1 shows an example of a continuous contact motion and the corresponding point moving along an edge on the C-space obstacle boundary. Compared with the non-trivial shape complexity of the whole C-space (Figure 1(c)) of the two given planar curves, the two/three contact arrangement of C-space (Figure 1(b)) has a relatively simple network structure of two-contact motion curve segments. The edges and vertices in this graph are two/three-contact points, which can be computed by solving a system of equations of the form: $F_i(x, y, \theta) = 0$, for $1 \leq i \leq K$, where $K = 2$ or 3. Though the C-space obstacle boundary is composed of high-degree curves and surfaces, we have developed a highly stable algorithm for computing the two-contact motion curves based on a geometric condition that can guarantee the topological structure of the C-space curve arrangement.

One may consider an alternative approach using polygonal approximations of the two planar curves $C(u)$ and $D(v)$. Then the C-space obstacle boundary will be approximated with a large number of ruled surface patches

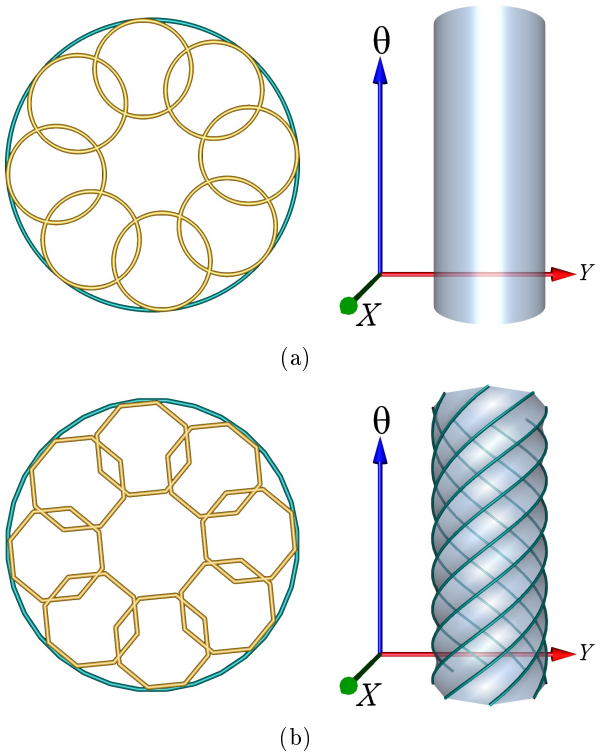

Figure 2: Piecewise linear approximation of input smooth curves and the corresponding C-space obstacle boundary: (a) two circles and their C-space obstacle boundary, (b) piecewise linear approximations of two circles (by an octagon and a 32-gon) and the corresponding C-space obstacle boundary computed by the algorithm of Avnaim et al. [1]. The piecewise linear approximation produces two-contact configurations on the C-space boundary, which don't exist on the C-space boundary of original input curves.

and their intersection curves [1]. This straightforward approach will be a highly inefficient solution that produces low-precision results. More seriously, it is extremely difficult to preserve the contact configurations under the piecewise linear approximation of the two curves. Figure 2 shows a typical example of such problem. Even though the original input curves (two circles) clearly have no two-contact configuration, the piecewise linear approximations of the two circles, using an octagon and a 32-gon, produce eight two-contact motion curves on the C-space obstacle boundary. These redundant solutions from the piecewise linear approximation makes it difficult to find the proper contact configurations on the original C-space obstacle boundary. Thus it is important to develop an algorithm that can directly deal with the given freeform geometric models without a polygonal approximation.

The rest of this paper is organized as follows. In Section 2, we briefly review related previous work. Section 3 establishes the algebraic conditions for the contact configurations between two planar B-spline curves. Sections 4 and 5 discuss the construction of the C-space obstacle boundary and the two-contact motion planning strategy, respectively. Experimental results are reported in Section 6 and the paper will be concluded in Section 7.

2 Related Previous Work

Lozano-Pérez and Wesley [24] and Lozano-Pérez [22, 23] introduced the concept of Configuration space obstacles to robotics as a useful tool for planning collision-free paths of a polygonal or polyhedral object moving amidst polygonal or polyhedral obstacles. For the rotational degrees of freedom, the swept volume of the moving object (for a certain range of rotations) is approximated with a polygon or polyhedron. The C-space obstacle is then approximated with a discrete union of low-dimensional C-space obstacle slices. For the motion planning of polyhedra with six degrees of freedom, Donald [11] incrementally constructed a connected path that consists of 1-manifold curve segments on the 5-manifold C-space obstacle boundary.

Avnaim et al. [1] presented the first algorithm for constructing an explicit C-space obstacle boundary in the $xy\theta$-space for a non-convex polygonal object moving among similar polygonal obstacles. Brost [6, 7] presented efficient implementations for the C-space obstacle boundary construction, represented as the network of two-contact motion constraints. Our current work may be regarded as an extension of Brost [6, 7] to the general case of a C^1-continuous planar curve moving among similar C^1-continuous planar curves. Recently, Milenkovic et al. [25, 26] developed a robust algorithm for constructing the C-space obstacles for planar moving object and obstacles bounded by circular arcs.

Bajaj and Kim [2, 3, 4] considered the generation of C-space obstacles for translational motions of algebraic curves and surfaces, which produce algebraic curves and surfaces of extremely high-degree and thus have never been implemented. For the case of a planar freeform curve moving (with translation only) among similar freeform curves in the plane, Lee et al. [21] presented a high-precision algorithm that can approximate the C-space obstacle boundary with B-spline planar curves. Nevertheless, there is no sufficiently reliable high-precision algorithm developed for the case of a freeform surface moving (with three degrees of freedom in translation) among freeform surfaces. In this work, we attack a different C-space generation problem for the case of a planar curve moving with three degrees of freedom in translation and rotation, using an approach that is potentially extendable to higher dimensions.

Mechanical design is a typical example where high-precision solutions are needed for the generation of C-space obstacles. Joskowicz and Sacks [16, 17, 28] presented various applications of high-precision C-space computation for kinematic design of moving mechanisms. In a recent work, Nelaturi and Shapiro [27] proposed an adaptive sampling technique on the C-space obstacles as an effective way for solving some non-trivial problems in mechanical design.

For freeform geometric models represented with NURBS curves and surfaces, geometric constraints are usually converted to a system of multivariate piecewise-polynomial equations. Sherbrooke and Patrikalakis [30] proposed an early subdivision-based approach for handling this problem. By introducing a simple reduction scheme for the solution domains, Elber and Kim [12] developed a multivariate equation solver for a system of equations that also combines symbolic and numeric techniques. The current work is also based on this solver which has been implemented as a part of the IRIT solid modeling system [15]. In a recent work, Barton et al. [5] presented an efficient algorithm for handling the special case of univariate solution curves, which is the most relevant computational tool for computing two-contact motion curves in this work. This solver finds all solutions globally and ensures the topology of the result and, up to a prescribed tolerance. This guarantee enables the robust reconstruction of the C-space.

3 Contact Conditions for Two Planar Curves

Given two orientable planar C^1-continuous regular parametric curves $C(u) = (x_C(u), y_C(u))$ and $D(v) = (x_D(v), y_D(v))$, for $0 \le u, v \le 1$ (an assumption used throughout this work unless stated otherwise), the algebraic condition for their tangential contact at a common touching point $C(u) = D(v)$ can be characterized as follows:

$$\begin{aligned} C(u) - D(v) &= 0, \\ \det(C'(u), D'(v)) &= 0, \end{aligned}$$

where det() is the 2D determinant of two vectors. Equivalently, we have the following over-constrained system of three scalar equations in two variables u and v:

$$\begin{aligned} x_C(u) - x_D(v) &= 0, \\ y_C(u) - y_D(v) &= 0, \\ x'_C(u)y'_D(v) - y'_C(u)x'_D(v) &= 0, \end{aligned}$$

which has no solution in general. As we allow the curve $C(u)$ to move in the xy-plane, $C(u)$ may have some tangential contact with $D(v)$ at some configurations.

When the curve $C(u)$ is rotated by angle θ and translated by (x, y), the resulting contact condition is given as follows:

$$\begin{aligned} R_\theta(C(u)) + (x, y) - D(v) &= 0, \\ \det(R_\theta(C'(u)), D'(v)) &= 0, \end{aligned}$$

where R_θ is a 2×2 rotation matrix. Equivalently, we have three equations in five variables

$$\begin{aligned} 0 &= \cos\theta \cdot x_C(u) - \sin\theta \cdot y_C(u) + x - x_D(v), \\ 0 &= \sin\theta \cdot x_C(u) + \cos\theta \cdot y_C(u) + y - y_D(v), \\ 0 &= \hat{F}(u, v, \theta) = \left[\cos\theta \cdot x'_C(u) - \sin\theta \cdot y'_C(u)\right] \cdot y'_D(v) \\ &\quad - \left[\sin\theta \cdot x'_C(u) + \cos\theta \cdot y'_C(u)\right] \cdot x'_D(v). \end{aligned}$$

The third equation specifies a constraint $\hat{F}(u, v, \theta) = 0$ among three variables u, v, θ, whereas the first two equations represent x and y as explicit mappings of u, v, θ:

$$\begin{aligned} x &= -\cos\theta \cdot x_C(u) + \sin\theta \cdot y_C(u) + x_D(v), \\ y &= -\sin\theta \cdot x_C(u) - \cos\theta \cdot y_C(u) + y_D(v). \end{aligned}$$

We assume the coordinate functions $x_C(u)$, $y_C(u)$, $x_D(v)$, and $y_D(v)$ are all polynomial or rational functions, and the trigonometric functions $\cos\theta$ and $\sin\theta$ are reparameterized by rational functions of t: $\cos\theta = r_c(t)$ and $\sin\theta = r_s(t)$. Consequently, the rotation matrix R_θ can be reparameterized by t, which is denoted as R_t. Moreover, the constraint equation $\hat{F}(u, v, \theta) = 0$ can be converted to an implicit algebraic equation:

$$\begin{aligned} 0 &= F(u, v, t) \\ &= \left[r_c(t) \cdot x'_C(u) - r_s(t) \cdot y'_C(u)\right] \cdot y'_D(v) \\ &\quad - \left[r_s(t) \cdot x'_C(u) + r_c(t) \cdot y'_C(u)\right] \cdot x'_D(v). \end{aligned}$$

Similarly, x and y can be represented as rational mappings of u, v, t:

$$\begin{aligned} x &= G(u, v, t) \\ &= -r_c(t) \cdot x_C(u) + r_s(t) \cdot y_C(u) + x_D(v), \\ y &= H(u, v, t) \\ &= -r_s(t) \cdot x_C(u) - r_c(t) \cdot y_C(u) + y_D(v). \end{aligned}$$

Multiple contact conditions can also be characterized in a similar way:

$$\begin{aligned} R_\theta(C(u_i)) + (x, y) - D(v_i) &= 0, \\ \det(R_\theta(C'(u_i)), D'(v_i)) &= 0, \end{aligned}$$

for $i = 1, \cdots, K$, where K is the number of tangential contacts. Equivalently, for a K-contact (K tangential contacts), we have the following $3K$ scalar equations in $2K+3$ variables:

$$\begin{aligned} x &= G_i(u_i, v_i, t) \\ y &= H_i(u_i, v_i, t) \\ 0 &= F_i(u_i, v_i, t), \end{aligned}$$

for $i = 1, \cdots, K$. The linear terms x and y can be eliminated by replacing them with $G_1(u_1, v_1, t)$ and $H_1(u_1, v_1, t)$, respectively. Consequently, we get the following $3K-2$ scalar equations in $2K+1$ variables: u_i, v_i, t:

$$0 = G_1(u_1, v_1, t) - G_i(u_i, v_i, t), \text{ for } 2 \le i \le K, \tag{3.1}$$
$$0 = H_1(u_1, v_1, t) - H_i(u_i, v_i, t), \text{ for } 2 \le i \le K, \tag{3.2}$$
$$0 = F_i(u_i, v_i, t), \text{ for } 1 \le i \le K. \tag{3.3}$$

In addition to contact conditions, we can enforce curve $C(u)$ to be locally exterior to $D(v)$ by adding the following inequality condition based on the orientations of the two input curves:

$$\left\langle (R_\theta(C'(u_i)), D'(v_i) \right\rangle < 0, \text{ for } 1 \le i \le K,$$

where $\langle *, * \rangle$ denotes the inner product of two vectors.

4 Generation of C-Space Obstacle Boundary

We discuss the generation of the C-space curve arrangement, mainly concentrating on the curve network of two-contact motions embedded on the boundary. There are two major technical issues we need to consider: (i) one is how to identify collision-free motions from those leading to inter-penetration into the obstacle interior and (ii) the other is how to resolve local two-contactness. For this purpose, we present an efficient algorithm for pruning redundant solutions that do not contribute to the C-space boundary, in Section 4.2. Local two-contactness is hard to detect, especially when the contact locations are very close to each other. Thus we propose simple geometric conditions that guarantee local single-contact, used by a hierarchical subdivision, in Section 4.3. As the construction of two-contact motion curves is a time-consuming process, we also consider acceleration techniques based on a hierarchical data structure for planar freeform curves, in Section 4.4.

4.1 Curve Network of Two-Contact Motions

We solve a system of non-linear equations for two contact motions ($K = 2$), which has four constraint equations (3.1)–(3.3), in five variables: u_1, u_2, v_1, v_2, t (recall that x and y were eliminated). For this purpose, we employ the algorithm of Barton et al. [5] which can handle a general system of non-linear equations that has n equations with $n + 1$ variables. The algorithm yields one dimensional solutions and thus we may assume that the solution curves are parameterized as $(u_1(a), u_2(a), v_1(a), v_2(a), t(a))$, for $0 \le a \le 1$. The corresponding two-contact planar motion is then generated as a one-parameter motion $(x(a), y(a), \theta(t(a)))$ of a, where $x(a) = G_1(u_1(a), v_1(a), t(a)) = G_2(u_2(a), v_2(a), t(a))$ and $y(a) = H_1(u_1(a), v_1(a), t(a)) = H_2(u_2(a), v_2(a), t(a))$.

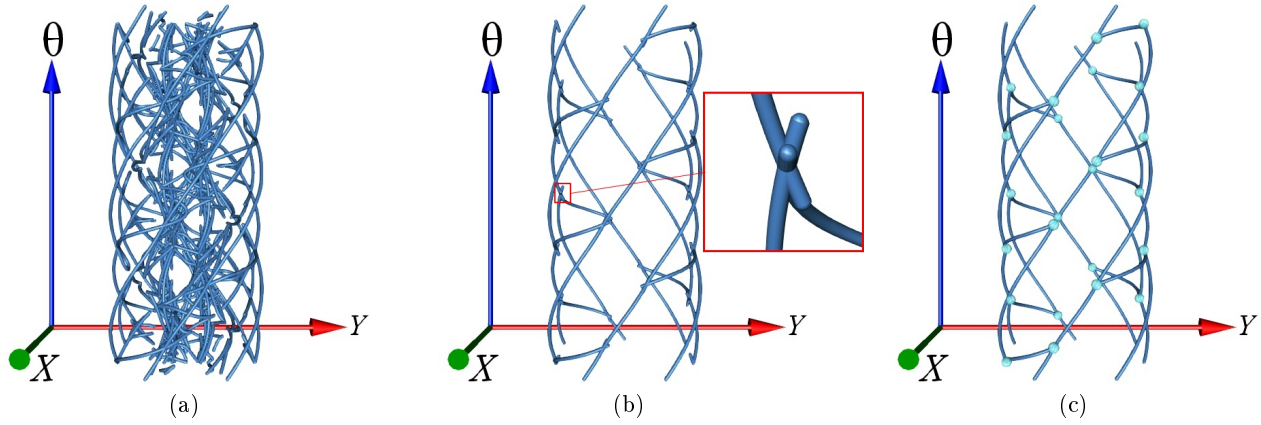

Figure 3: Solutions for the two-contact configurations of Figure 1: (a) all solutions for two-contact configurations, (b) two-contact configurations after eliminating redundant two-contact configurations via domain pruning algorithm, and (c) the complete curve network of collision free two-contact motions, after resolving curve arrangement by computing three-contact configurations, as in Figure 1(b).

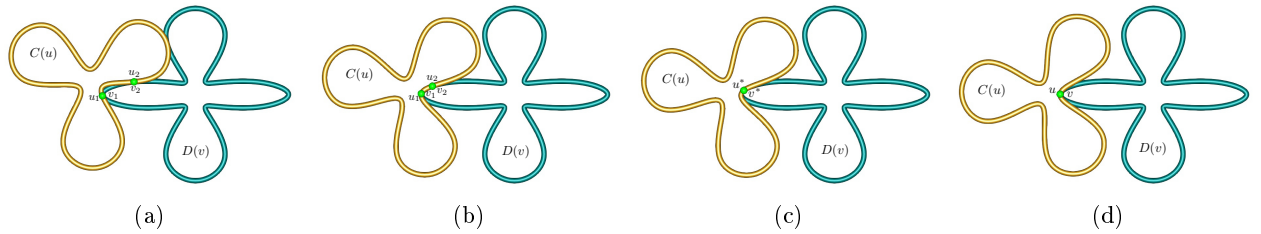

Figure 4: *Curvature Contact* point of two-contact motion: (a) and (b) two-contact motion, (c) curvature matched contact, and (d) one-contact motion.

Note that $\theta(t)$ is the rotational angle defined as follows: $\theta(t) = \arccos(r_c(t))$ or $\theta(t) = \arcsin(r_s(t))$, for some rational functions $r_c(t)$ and $r_s(t)$ of t.

The constraint equations for two-contact motion include many redundant solutions. Figure 3(a) shows an entire set of two-contact motion solutions for the two planar curves in Figure 1. The majority of these solutions do not contribute to the boundary of the C-space obstacle and thus should be pruned out. For an efficient computation, we prune the redundant solutions using a simple culling test, which is discussed in Section 4.2. Figure 3(b) shows the solutions remaining from the pruning step.

Even after applying the pruning algorithm, the solutions may still include some redundant segments (see Figure 3(b)). To complete the C-space curve arrangement, we need to compute the intersection points among the two-contact motion curves on the C-space boundary, which are typically three-contact configurations. The three-contact configuration ($K = 3$) can be computed by solving seven constraint equations in seven variables, which produces discrete configurations. Figure 3(c) shows three-contact configurations (in cyan) with a complete C-space curve arrangement. The contact point can also be a $K > 3$ contact configuration and we treat it as multiple triple contact points (i.e., a four contact is the overlap of four triple contact configurations).

Interestingly enough, three-contact configuration is not always a terminal point of the two-contact motion curve. The two-contact motion curve may, in special cases, converge to a single contact point. Figure 4 shows an example of such a case. Figure 4(c) shows the special single contact, where the two curves share the same curvature at the contact point. After passing the special point, the two curves maintain only a single contact, as shown in Figure 4(d), where the moving curve $C(u)$ has curvature lower than the other curve $D(v)$. The algebraic condition for this special case can be formulated as follows (please see Lemma 2 of Appendix A for the proof):

$$\begin{aligned} \kappa_C(u) - \kappa_D(v) &= 0, \\ \kappa'_C(u) - \kappa'_D(v)\frac{\partial v}{\partial u} &= 0, \\ \det(R_\theta(C'(u)), D'(v)) &= 0, \end{aligned} \tag{4.1}$$

where $\kappa_C(u)$ and $\kappa_D(v)$ are the curvature functions for the two curves. We denote this special condition as *curvature contact*.

The inter-penetration of two-contact motion always starts at a $K \geq 3$ contact configuration and the convergence of two (or more) contact points to a single contact point is realized at a curvature contact. When the two-contact motion causes no inter-penetration and two contact points do not merge to a single contact point, the configurations on the two-contact motion are always valid and the two-contact motion terminates only at $K \geq 3$ contact or curvature contact configurations.

To complete the C-space curve arrangement, we compute curvature contact points together with $K \geq 3$ contact points, and use them as terminal points of the two-contact motion.

4.2 Domain Pruning

Solving the algebraic equations for the two-contact motion curves is a time-consuming process and, as mentioned earlier, many of these solutions are often redundant. As a

consequence, an effective culling of redundant solutions has a significant impact on the performance of the overall algorithm. For example, solving entire two contact constraint equations, shown in Figure 3(a), requires a few gigabytes of memory and several hours of computation time. On the other hand, using an efficient pruning algorithm,, shown in the Figure 3(b), it takes only a couple of minutes and only twenty megabytes of memory for the same problem.

The points on the boundary of the C-space obstacle can be characterized using the geometric concept of *penetration depth*. Given two objects A and B, their penetration depth $PD(A, B)$ can be defined as the distance between their deepest interpenetration points [31]. Using this definition, we can prescribe the points (x, y, θ) on the C-space boundary as follows:

$$\{(x, y, \theta) | PD(C(u, x, y, \theta), D(v)) = 0\}$$

For a given sub-domain $\mathcal{D} = [\underline{u}, \overline{u}] \times [\underline{v}, \overline{v}] \times [\underline{t}, \overline{t}]$ and the corresponding C-space:

$$CS_D = \{(x, y, \theta) | x = G(u, v, t), y = H(u, v, t), \theta = \theta(t), \\ (u, v, t) \in \mathcal{D}\},$$

consider the minimum penetration depth for CS_D:

$$\underline{PD}(CS_D) = \min\{PD(C(u, x, y, \theta), D(v)) | (x, y, \theta) \in CS_D\}$$

If $\underline{PD}(CS_D) > 0$, clearly, CS_D cannot contribute to the C-space boundary and thus we can safely prune out the sub-domain $\mathcal{D}$. But, a precise computation of $\underline{PD}(CS_D)$ is non-trivial and computationally quite expensive; thus, we compute a lower bound of $\underline{PD}(CS_D)$ instead.

Let (u_m, v_m, t_m) be the mid-point of the domain $\mathcal{D}$ and (x_m, y_m, θ_m) be the corresponding configuration. We compute $PD_m = PD(C(u, x_m, y_m, \theta_m), D(v))$ using the 2D version of the algorithm of Tang et al. [31]. Let the deepest interpenetration point on the moving curve be denoted by $C(u^*, x_m, y_m, \theta_m)$. Now consider a maximum distance $Dist_{max}$ between $C(u^*, x_m, y_m, \theta_m)$ and $C(u^*, x, y, \theta)$ where $(x, y, \theta) \in CS_D$. Then $\underline{PD}(CS_D)$ can be bounded by the following inequality:

$$\underline{PD}(CS_D) > PD_m - Dist_{max}.$$

4.3 Conditions for Single-Contactness

Finding all two-contact motion curves on the boundary of the C-space and ensuring its topology is a challenging problem, in particular, when two different tangential contact points $C(u_1) = D(v_1)$ and $C(u_2) = D(v_2)$ are located very closely to each other. Because of numerical errors, it is difficult to distinguish whether two nearby contact points are identical or different. For example, it is non-trivial to detect the degenerate contact as shown in Figure 4(c) when we rely solely on algebraic techniques for solving a system of equations. Thus we consider simple geometric conditions that can guarantee the local single-contactness as in the case of Figure 4(d).

For a given positively oriented (counterclockwise) curve $\mathcal{C}(t)$, $0 \le t \le 1$, we say that the curve $\mathcal{C}(t)$ is convex (or concave), if $\det(\mathcal{C}'(t), \mathcal{C}''(t)) > 0$ (or $\det(\mathcal{C}'(t), \mathcal{C}''(t)) < 0$), for all $t \in [0, 1]$. For example, $C(u)$ is convex and $D(v)$ is concave in Figure 5(a).

When a convex part of $C(u)$ is in contact with a convex part of $D(v)$, the two curves may have only a single-contact (locally, though not globally). Concave curves have no chance of having a tangential contact at all. On the other hand, when the minimal curvature of $C(u)$ is higher than the maximal curvature of $D(v)$, the convex part of $C(u)$ may have only a single-contact (locally) with the concave part of $D(v)$.

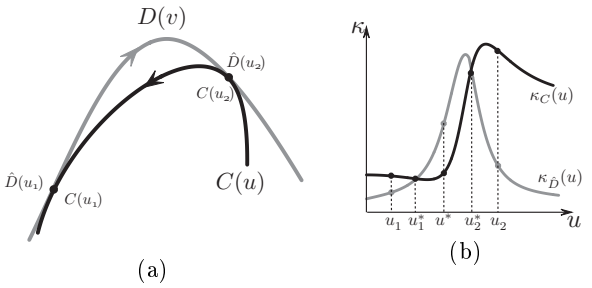

Figure 5: Relative curvature distribution in a two-contact configuration: (a) two planar curves in a two-contact configuration, and (b) their curvature plot.

Depending on the relative curvature distribution of $C(u)$ with respect to the other curve $D(v)$, the convex part of $C(u)$ may have two different tangential contacts with the concave part of $D(v)$ (see Figure 5). The following lemma presents a necessary condition for the two-contactness of the two curves, the negation of which also produces a sufficient condition for the single-contactness of the two curves.

Lemma 1. *Given two different curvature-continuous curve segments $C(u)$ and $D(v)$, where $C(u)$ is convex and $D(v)$ is concave, if there exists a two-contact configuration between the two curves at $C(u_1) = D(v_1)$ and $C(u_2) = D(v_2)$ with $u_1 < u_2$, there are at least two different solutions for the following system of equations:*

$$\kappa_C(u) - \kappa_D(v) = 0, \tag{4.2}$$

$$\det(C'(u), D'(v)) = 0, \tag{4.3}$$

where $\kappa_C(u)$ and $\kappa_D(v)$ are the curvature functions of the curves $C(u)$ and $D(v)$, respectively.

Proof. Due to the convexity and concavity of the two curves, there exists a one-to-one correspondence in their tangent-directions between the two curve domains $[u_1, u_2]$ and $[v_1, v_2]$, which satisfies Equation (4.3). Thus, we assume a reparameterization of $v = v(u)$, $u_1 \le u \le u_2$, so that the two curves $C(u)$ and $\hat{D}(u) = D(v(u))$ share the same tangential direction at the same parameter u: $C'(u) \parallel \hat{D}'(u)$, where $\hat{D}'(u) = v'(u)D'(v(u))$.

The curvature $\kappa_C(u)$ is greater than or equal to $\kappa_{\hat{D}}(u)$ in sufficiently small neighborhoods of $C(u_i) = D(v_i)$, $(i = 1, 2)$; otherwise, the curve $C(u)$ inter-penetrates $D(v)$ in the neighborhood of $C(u_i) = D(v_i)$. Now, when we assume the existence of an intermediate parameter $u^* \in (u_1, u_2)$ such that $\kappa_C(u^*) < \kappa_{\hat{D}}(u^*)$, the continuity of the curvature functions $\kappa_C(u)$ and $\kappa_{\hat{D}}(u)$ guarantees the existence of u_1^* and u_2^* with $\kappa_C(u_i^*) = \kappa_{\hat{D}}(u_i^*)$, $(i = 1, 2)$, (see Figure 5(b)) and we completed the proof.

Thus, we only need to prove the existence of such a parameter $u^* \in (u_1, u_2)$. Assume otherwise; that is, assume no such u^* may exist, then we have $\kappa_C(u) \ge \kappa_{\hat{D}}(u)$, for all $u \in [u_1, u_2]$. For an arc-length parametrization $s \in [s_1, s_2]$, $(u_i = u(s_i))$, of $C(u)$, we have the following relation:

$$\frac{C'(u_2)}{||C'(u_2)||} - \frac{C'(u_1)}{||C'(u_1)||} = \int_{s_1}^{s_2} \kappa_C \cdot \mathbf{n} \, ds,$$

where $\mathbf{n}$ is the common unit normal vector of $C(u)$ and $\hat{D}(u)$. Similarly, for an arc-length parametrization $\overline{s}$ of $\hat{D}(u)$,

$$\frac{\hat{D}'(u_2)}{||\hat{D}'(u_2)||} - \frac{\hat{D}'(u_1)}{||\hat{D}'(u_1)||} = \int_{\overline{s}_1}^{\overline{s}_2} \kappa_{\hat{D}} \cdot \mathbf{n} \, d\overline{s}.$$

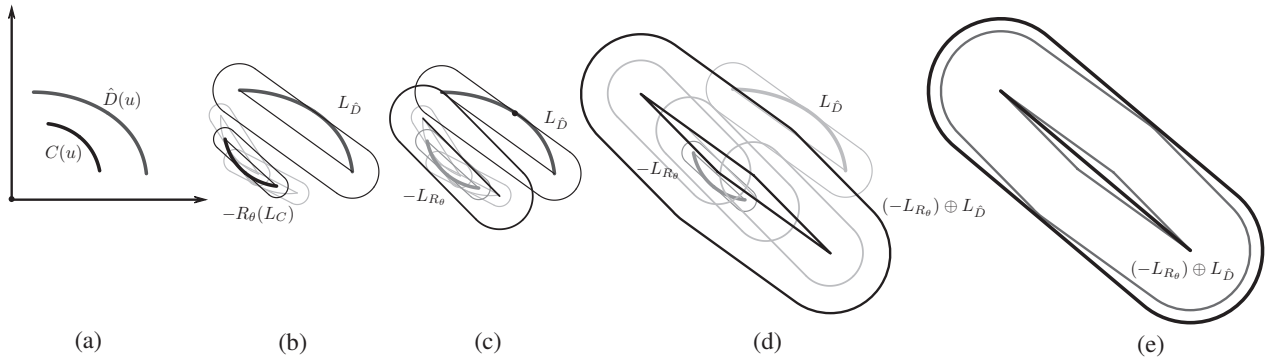

Figure 6: Construction of line swept circle (LSC) regions: (a) the static curve $\hat{D}(u)$ and the initial configuration of the moving curve $C(u)$, (b) their LSC regions with rotation, (c) the LSC region that bounds the rotational swept region $\cup_{\underline{\theta} \le \theta \le \overline{\theta}} (-R_\theta(L_C))$, (d) the Minkowski sum $(-L_{R_\theta}) \oplus L_{\hat{D}}$, and (e) the LSC region that bounds the Minkowski sum.

But then,

$$\int_{s_1}^{s_2} \kappa_C \cdot \mathbf{n} \; ds = \int_{\overline{s}_1}^{\overline{s}_2} \kappa_{\hat{D}} \cdot \mathbf{n} \; d\overline{s},$$

and since $\kappa_C(u) \geq \kappa_{\hat{D}}(u)$, it must be the case that $\kappa_C(u) = \kappa_{\hat{D}}(u)$, for all $u \in [u_1, u_2]$. Therefore, the two curves are identical in contrast to the assumption that they are different. □

Lemma 1 means that the two curves $C(u)$ and $D(v)$ have at least two locations where they share the same curvature as well as the same tangent direction, which is represented by Equations (4.2) and (4.3), respectively. By negating the conclusion of Lemma 1, we get a sufficient condition for the single contactness of a convex-concave curve pair; namely, when the system of Equations in Lemma 1 has at most one solution, the two curves can have no contact or only a single contact. For this purpose, we employ the single solution test of Hanniel and Elber [13], which guarantees that the system of Equations (4.2)–(4.3) can have at most one solution.

We can extend Lemma 1 to the general case between a rotated curve $R_\theta(C(u))$, for $\underline{\theta} \le \theta \le \overline{\theta}$, and a static curve $D(v)$. If Equations (4.2) and (4.3) have at most one solution for all θ, there exists no double contact configuration between the two curves. The single solution test can also be extended simply using the generalized gradient of Equation (4.3) due to the rotation.

4.4 Testing for K-Contactness

Given K curve pairs $R_\theta(C(u_i))$ and $D(v_i)$, $i = 1, \cdots, K$, we check the necessary conditions for a K-contact configuration, which are based on the algebraic conditions (3.1)–(3.3), to eliminate redundant curve pairs from further consideration. We test if there is some overlap between their G- and H-functions, namely, their configurations in the xy-plane. Otherwise, there is no K-contact configuration feasible for the K pairs. To further accelerate this condition check, we employ a hierarchical data structure for the planar freeform curves under consideration, the details of which will be discussed in the next subsection 4.5.

Before we give the details of the data structure, we review the F-, G-, and H-functions. The F-function is simply a condition for the slope matching between the rotated curve $R_\theta(C(u))$ and the other curve $D(v)$:

$$F(u, v, \theta) = \det(R_\theta(C'(u)), D'(v)) = 0,$$

which means that the solution of $F(u, v, \theta) = 0$ provides a reparameterization of $v = v(u, \theta)$ as a bivariate function of u and θ. Thus, we have

$$\begin{aligned} x &= \hat{G}(u, \theta) = G(u, v(u, \theta), \theta) \\ &= -\cos\theta \cdot x_C(u) + \sin\theta \cdot y_C(u) + x_D(v(u, \theta)), \\ y &= \hat{H}(u, \theta) = H(u, v(u, \theta), \theta) \\ &= -\sin\theta \cdot x_C(u) - \cos\theta \cdot y_C(u) + y_D(v(u, \theta)). \end{aligned}$$

Now consider the problem of testing whether there is some overlap among the K bivariate surfaces in the xy-plane: $(x, y) = (\hat{G}(u_i, \theta), \hat{H}(u_i, \theta))$, $\underline{u}_i \le u_i \le \overline{u}_i$, $i = 1, ..., K$, for a certain range of orientations of $C(u_i)$, $\underline{\theta} \le \theta \le \overline{\theta}$. The overlap between the K surfaces is a necessary condition for a possible K-contact between $R_\theta(C(u_i))$ and $D(v_i)$, for $i = 1, ..., K$, at K different locations. Of course, the generation of these bivariate surfaces in the xy-plane is time-consuming. Thus we consider the generation of simple bounding regions for these two surfaces using a hierarchical data structure which can be pre-computed for the two planar freeform curves $C(u)$ and $D(v)$.

4.5 Acceleration using Bounding Region Hierarchy

We build a bounding region hierarchy (BRH) for each curve. The curve is first subdivided at each inflection point. Each sub-curve is then either convex, concave, or linear. A binary tree is generated for each segment by recursively subdividing the sub-curves at mid-parameters until the sub-curves gets shorter than a certain length. Each node of the tree contains geometric data such as a bounding region, and the ranges of tangent directions, curvatures, and curvature derivatives. We employ a line swept circle (LSC) as the bounding region for the curve segments [18]. Each curve segment is approximated by a line segment connecting the two endpoints, and the maximum approximation error is bounded from above by a certain value $\epsilon > 0$. By sweeping a circle of radius ϵ along the line segment, we can generate an LSC region that completely bounds the curve segment.

We use the BRH to detect an inter-penetration between two curves. Starting from the root nodes of the two hierarchies, we test whether the two LSC regions overlap. The overlap between two LSC regions is tested by computing the distance between their center line segments and comparing if the distance is smaller than the sum of the two radii of the LSC regions. If two LSCs overlap, we proceed to the next level of hierarchy and repeat the overlap test until we reach the leaf nodes. Once we reach the leaf nodes,

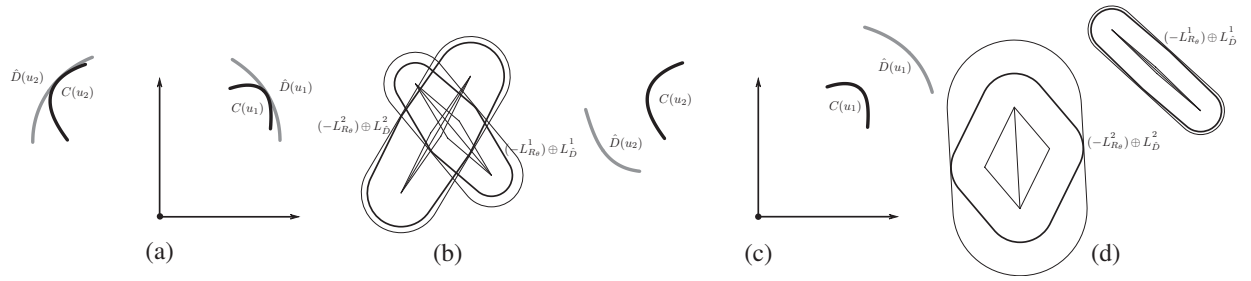

Figure 7: Overlap test for two curve pairs: (a) Curve pairs having a two-contact configuration, (b) overlapping bounding regions for the curve pairs of (a), (c) Curve pairs with no two-contact configuration, and (d) non-overlapping bounding regions for the curve pairs of (c).

we compute the intersection point and check if the intersection is transversal. If the intersection is transversal, we consider it as an inter-penetration of the two curves.

We also use the BRH to accelerate the K-contactness test presented in the previous subsection 4.4. Given two curve segments $R_\theta(C(u))$ and $\hat{D}(u)$, for $\underline{u} \le u \le \overline{u}$, and a range of rotation angles, $\underline{\theta} \le \theta \le \overline{\theta}$, we construct a bounding region for the bivariate surface $(x, y) = (\hat{G}(u, \theta), \hat{H}(u, \theta))$ contained in the xy-plane. The curves $C(u)$ and $\hat{D}(u)$ are bounded by LSC regions L_C and $L_{\hat{D}}$, respectively. The rotational swept region $\cup_{\underline{\theta} \le \theta \le \overline{\theta}} R_\theta(L_C)$ of L_C is then bounded by another LSC region $L_{R_\theta C}$. Finally, the Minkowski sum $(-L_{R_\theta C}) \oplus L_{\hat{D}}$ is bounded by a larger LSC region. The bivariate surface $(\hat{G}(u, \theta), \hat{H}(u, \theta))$ is then bounded by this LSC region as is evident from the following relation:

$$\begin{aligned}(x, y) &= (\hat{G}(u, \theta), \hat{H}(u, \theta)) \\ &= -R_\theta(C(u)) + D(v(u, \theta)) \\ &\subset -R_\theta(L_C) \oplus L_{\hat{D}} \\ &\subset (-L_{R_\theta C}) \oplus L_{\hat{D}},\end{aligned}$$

where the Minkowski sum operation $\oplus$ is defined as $A \oplus B = \{a + b \mid a \in A, b \in B\}$.

Figure 6 illustrates some important construction steps of the LSC regions. Figure 6(a) shows the static curve $\hat{D}(u)$ and an initial configuration of the moving curve $C(u)$. In Figure 6(b), three instances of the rotating curve $-R_\theta(C(u))$ are shown (two in gray and the middle one in black), and each curve is bounded by the respective LSC region. Figure 6(c) shows the LSC region that bounds the rotational swept region $\cup_{\underline{\theta} \le \theta \le \overline{\theta}} (-R_\theta(L_C))$ of $-L_C$. Figures 6(d)-(e) show the Minkowski sum $(-L_{R_\theta C}) \oplus L_{\hat{D}}$ and the LSC region that bounds $(-L_{R_\theta C}) \oplus L_{\hat{D}}$, respectively.

Given two pairs of curve segments: $(C(u_1), \hat{D}(u_1))$, $\underline{u}_1 \le u_1 \le \overline{u}_1$, and $(C(u_2), \hat{D}(u_2))$, $\underline{u}_2 \le u_2 \le \overline{u}_2$, where each pair is now guaranteed to have a local single-contactness, we test if there is an overlap between the corresponding bivariate surfaces $(\hat{G}(u_1, \theta), \hat{H}(u_1, \theta))$ and $(\hat{G}(u_2, \theta), \hat{H}(u_2, \theta))$, using their bounding LSC regions thus constructed. Figure 7 shows two examples of the overlap test using these bounding regions. In Figure 7(a), each pair of curve segments $(C(u_i), \hat{D}(u_i))$, $\underline{u}_i \le u_i \le \overline{u}_i$, $(i = 1, 2)$, has a tangential contact. Consequently, their bounding LSC regions must overlap as shown in Figure 7(b). On the other hand, the two LSC regions of Figure 7(d) have no overlap, which means that at least one pair of curve segments has no tangential contact. Figure 7(c) shows an example, where no pair has a tangential contact.

The overlap test proceeds recursively from the root of the binary tree for each curve segment to the leaf level of the hierarchy. As we go down to the lower levels of the hierarchy, the overlap tests using LSC bounding regions provide more and more accurate results. In this hierarchical approach, we generate a superset of candidates for the two/three-contact configurations, and to each of which we apply the precise computations of solving the system of equations characterizing the two/three-contact condition.

The following algorithms (Algorithms 1–2) summarize the overall procedure for the K-contact computation.

Algorithm 1 K-Contact-Computation
Input: Curve pairs $(R_\theta(C(u_i)), D(v_i))$, $i = 1, \cdots, K$.
Output: K-contact points.

1: **for** $i = 1 \to K - 1$ **do**
2: **if** $\overline{u}_{i+1} < \underline{u}_i$ **then**
3: **return** NULL (redundant domain);
4: **end if**
5: **end for**
6: **if** !K-Overlap-Test($(R_\theta(C(u_i)), D(v_i))$, $i = 1, \cdots, K$) **then**
7: **return** NULL;
8: **end if**
9: **if** Each $(R_\theta(C(u_i)), D(v_i))$ satisfies the single-contactness and small enough **then**
10: **return** Compute-Precise-K-contact();
11: **else**
12: Subdivide the $(R_\theta(C(u_i)))$, $D(v_i))$, $i = 1, \cdots, K)$ into two sub-domains $L(u_i, v_i, \theta)$ and $R(u_i, v_i, \theta)$;
13: K-Contact-Computation($L(u_i, v_i, \theta)$);
14: K-Contact-Computation($R(u_i, v_i, \theta)$);
15: **end if**

Note that a K-contact point, $(u_1, v_1, \cdots, u_k, v_k, \theta)$, has $K!$ potential equivalent representations. For example, a two contact point can be represented as $(u_1, v_1, u_2, v_2, \theta)$ or $(u_2, v_2, u_1, v_1, \theta)$. To prevent redundant computation due to the equivalent representations, we only consider the K-contact point representation $(u_1, v_1, \cdots, u_k, v_k, \theta)$ where $u_i < u_{i+1}$ for $i = 1, \cdots, K$. Algorithm 1 (K-Contact-Computation) ensures this condition (ascending order of u_i values) by rejecting redundant domains (in line 1-4). Algorithm 1 then subdivides curve pairs $R_\theta(C(u_i))$, $D(v_i))$ into a single-contact pair using the single contact conditions introduced in Section 4.3. On the other hand, Algorithm 2 (K-Overlap-Test) checks the necessary conditions for the K-contactness. For the subdivision, we find a curve segment having the largest tangent cone (that bounds all tangent directions) among the $2K$ curve segments and compare the angular span of the tangent cone with the domain size of the rotation, $\overline{\theta} - \underline{\theta}$. If the angular span of the tangent cone is larger, we subdivide the curve segment having the largest tangent cone, otherwise, we subdivide the rotation domain and form two sub-domains

$L(u_i, v_i, \theta)$ and $R(u_i, v_i, \theta)$. Once all the curve pairs get smaller than a certain tolerance, Algorithm 1 invokes a function Computes-Precise-K-contact() (in line 10) as a final stage, which computes the precise K-contact points by using a multi-variate equation solver [5] that ensures the correct topology up to some tolerance. This property allows one to guarantee the detection of valid yet narrow paths, up to some given accuracy, for the moving geometry.

4.6 Extension to Multiple Curves

In many applications, we have obstacles consisting of multiple curves. To handle such obstacles, we extend the K-contact algorithms to the case of multiple stationary curves (though the moving curve is still a single connected component). Given a moving curve $C(u)$ and n stationary curves $D_i(v)$, $i = 1, \cdots, n$, we compute the K-contact between $C(u)$ and $\cup D_i(v)$. In this case, we need to process all possible K curve pairs, $R_\theta(C(u_k))$, $D_i(v_k)$, where $i = 1, \cdots, n$, and $k = 1, ..., K$.

The K-contact algorithm for the multiple curve case requires more computations than that of a single curve pair, but it does not increase the complexity of the algorithm. Example 9(f) shows one example of the multiple curve case.

Algorithm 2 K-Overlap-Test
Input: Curve pairs $(R_\theta(C(u_i)), D(v_i))$, $i = 1, \cdots, K$.
Output: True if there is overlap, False otherwise.

```
1:  for i = 1 → K do
2:    if R_θ(C(u_i)) and D(v_i) share no tangent direction then
3:      return False;
4:    end if
5:  end for
6:  for i = 2 → K do
7:    if (Ĝ(u_1, θ), Ĥ(u_1, θ)) and (Ĝ(u_i, θ), Ĥ(u_i, θ)) don't overlap then
8:      return False;
9:    end if
10: end for
11: return True;
```

5 Continuous Contact Motion Planning

We consider the problem of contact motion planning using the C-space curve arrangement. Unlike traditional motion planning on the configuration space, which requires optimization to determine the extra degrees of freedom, we simply connect two contact motions in the C-space curve arrangement and build a sequence of two contact motions that connects the start and goal configurations.

Given the start and goal configurations, a continuous two-contact motion can be generated between the two configurations by using a graph search on the network of two-contact motion curves embedded in the C-space obstacle boundary. We may assume the start and goal configurations are initially given as contact situations (by translating $C(u)$ to the nearest point of $D(v)$ if necessary). If the start configuration is in one-contact, we transform the moving curve $C(u)$ until it has a two-contact with the other curve $D(v)$. Then, we keep the continuous two-contact motion. We do a similar motion planning for the goal configuration when it is in one-contact configuration. After that, in the graph of two-contact motion curves, we apply Dijkstra's algorithm [10] to find the shortest path from the start configuration to the goal configuration. The weight on each edge is set to the summation of angular displacement and translation. To compute the angular displacement of $C(u)$, we first find a bounding circle of $C(u)$. Then for a given angular distance $\Delta\theta$ and the radius r of the bounding circle, the angular displacement can be computed as $r \times \Delta\theta$. Clearly, the weight we set here is a heuristic which combines translations and angular distances, thus it can be combined differently depending on the application.

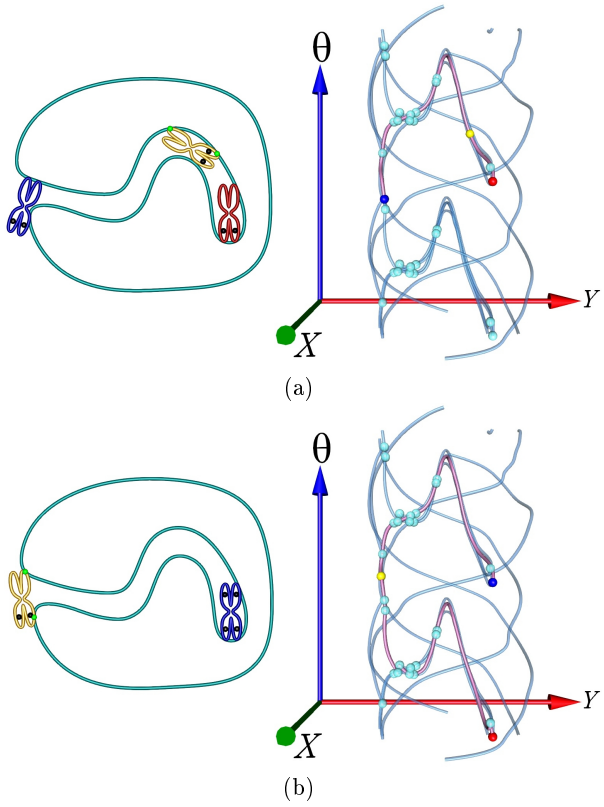

Figure 8: Motion planning with a continuous two-contact motion. Two black eyes indicate the orientation of the moving curve and the pink curve shows a two-contact motion path from start (blue) to goal (red). Note in (b) the start and goal configurations are the same but with flipped orientations

Figure 8(a) shows in the left subfigure an example of motion planning that starts from the blue curve and ends at the location in red. In the right subfigure, the current configuration point (shown in yellow) moves along the motion curve network, starting from the blue start point and going all the way to the red goal point. At each three-contact point where three two-contact motion curves meet, the curve tracing takes an unexplored branch which is closer to the red goal. Figure 8(b) shows a similar example where the start and goal positions are the same but the goal orientation is flipped from the start one (the goal red curve is hidden in this figure as the start and goal curves completely overlap each other). In this case, the moving yellow curve should get out of the narrow cave and then turn around at the entrance to the cave so as to meet the orientation constraint. Though the second problem of Figure 8(b) looks more difficult than the motion planning problem of Figure 8(a), the complexity of motion planning in the curve network is just about the same. (See the right subfigures of Figures 8(a)–(b).)

When we reach a curvature contact point (i.e., an isolated point in the curve network) during two-contact motion, instead of receding backward, we continue the contact motion (in a one-contact traversal) in the goal direc-

tion until a two-contact or a terminal configuration (either a three-contact or a curvature-contact) is reached again. The goal may be taken as a nearby terminal configuration. In case there is no such configuration within a certain distance, we may take the goal as the final destination of the motion planning.

The motion curve can be generated on a one-contact constraint surface $F(u, v, \theta) = 0$ by numerical tracing, starting from a curvature contact point (u_0, v_0, θ_0). Let $N(u, v, \theta) = 0$ denote the normal plane of $F(u, v, \theta) = 0$ at the point (u_0, v_0, θ_0) along the goal direction; namely, the plane $N(u, v, \theta) = 0$ contains the point (u_0, v_0, θ_0) and is spanned by the normal vector of $F(u, v, \theta) = 0$ and the direction vector to the goal. By intersecting the two surfaces $F(u, v, \theta) = 0$ and $N(u, v, \theta) = 0$, we can incrementally make small steps in the goal directions. We can repeat the same procedure at incremental positions (u_k, v_k, θ_k), $k = 1, \cdots$, until we cross a two-contact motion curve or reach a different terminal configuration (three-contact or curvature-contact) within a small distance. When there is no continuous path connecting the start and goal configurations on the C-space obstacle boundary, the sequence of incremental one-contact motions will eventually fail to proceed to the goal configuration. In that case, we declare that there is no solution between the given start and goal configurations.

6 Experimental Results

We have implemented our continuous contact motion planning algorithm in C++ and using the IRIT solid modeling system [15] on an Intel Core i7 3.4GHz PC with a 3.25GB main memory. To demonstrate the effectiveness of our approach, we have tested the algorithm for freeform planar curves of non-trivial complexity (please see the video [32] for the animated version of these results).

Figures 9(a)–(c) show three examples. In each, a closed C^1 curve (in yellow) moves around a stationary closed C^1 curve (in green) in a continuous two-contact motion. In the leftmost column of each example, the curve network of the two-contact motions is shown. The five figures on the right show several steps of two-contact motion where the yellow curve indicates the start configuration and the orange curve indicates the end configuration of the corresponding motion curve. When a two-contact motion reaches a curvature contact, it switches to a single-contact motion until it reaches a nearest two/three contact configuration. Red curves on the configuration space in Figures 9(b)–(c) show examples of single-contact motions. Figures 9(d)–(e) show two additional examples of continuous two-contact motion planning, in each of which the moving curve starts from the configuration in blue and goes to the goal configuration in red. In the leftmost column of each example, the curve network of the two-contact motions is shown, where the start and goal configurations are represented as blue and red dots, respectively.

Table 1 provides some statistics on these five examples. The second and third columns show the numbers of control points of the planar cubic B-spline curves $C(u)$ and $D(v)$, respectively. The next two columns report the number of global triple contacts (intersection free) and the number of two-contact motion curve segments in each example. Table 2 reports the timing results and memory consumptions for the domain pruning, tracing all the motion curve segments and computing all global triple contacts. Compared to solving for the entire domain of the algebraic equations, our algorithm shows significant performance improvement and with a huge reduction in memory usage. Computing a two contact motion curve takes much longer than a single triple contact. Once we compute the two contact motion curve, we can easily find domains where two or more contact motion curves coalesce. Using this information, we can efficiently detect all possible domains that might have triple contact solutions.

In Table 2, the curves $C(u)$ and $D(v)$ of Example (c) have about the same number of control points as Examples (a)–(b). However, the curve network for Example (c) is far more complex and its construction takes around three times more time than in Examples (a)–(b). The complexity of our algorithm seems to be more dependent on the complexity of the shape of the input curves, such as the number of cavities, than on the number of piecewise polynomial curve segments in the input curves, $C(u)$ and $D(v)$.

7 Conclusion

In this paper, we presented an efficient and stable algorithm for constructing the precise C-space obstacle boundary for a planar C^1 freeform curved shape moving around static obstacles bounded by planar C^1 freeform curves. As a roadmap embedded in the C-space obstacle boundary, we have constructed a network of two-contact motion curves, which can be employed for the efficient generation of continuous two-contact motions. We have demonstrated the effectiveness of the proposed approach using several nontrivial examples of continuous two-contact planar motions for planar objects bounded by C^1-continuous piecewise polynomial curves.

For two curves, the three-contact solution is a set of points, the two-contact solution yields univariate curves, and the one-contact solution creates bivariates, in general. Computing the full bivariate solution remains a challenge. The generalization of motion planning to higher-dimensional objects, such as space curves and surfaces in R^3, is conceptually simple but will demand a challenging optimization for the result to be computable in a reasonable time.

One should also consider the extension of the current result to a more general planar case where the curves $C(u)$ and $D(v)$ are C^1 continuous curves that are C^0-continuous at a finite set of locations. All C^1 discontinuity points should be treated carefully, possibly as small arcs of vanishing radius.

References

[1] F. Avnaim, J.-D. Boissonnat, B. Faverjon. A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles. *Proc. of 1988 IEEE Conf. on Robotics and Automation*, Philadelphia, Pennsylvania, April 24–29, 1988, pp. 1656–1661. Also in *Geometry and Robotics*, J.-D. Boissonnat and J.-P. Laumond (Eds.), Workshop, Toulouse, France, May 26–28,1988, Lecture Notes in Computer Science **391**, Springer-Verlag, pp. 67–86.

[2] C. Bajaj and M.-S. Kim. Generation of configuration space obstacles: the case of a moving sphere. *IEEE J. of Robotics and Automation*, **4**(1):94–99, 1988.

[3] C. Bajaj and M.-S. Kim. Generation of configuration space obstacles: the case of moving algebraic curves. *Algorithmica*, **4**(2):157–172, 1989.

[4] C. Bajaj and M.-S. Kim. Generation of configuration space obstacles: the case of moving algebraic surfaces. *The Int'l J. of Robotics Research*, **9**(1):92–112, 1990.

[5] M. Barton, G. Elber, I. Hanniel. Topologically guaranteed univariate solutions of underconstrained polynomial systems via no-loop and single-component tests. *Computer-Aided Design*, **43**(8):1035–1044, 2011.

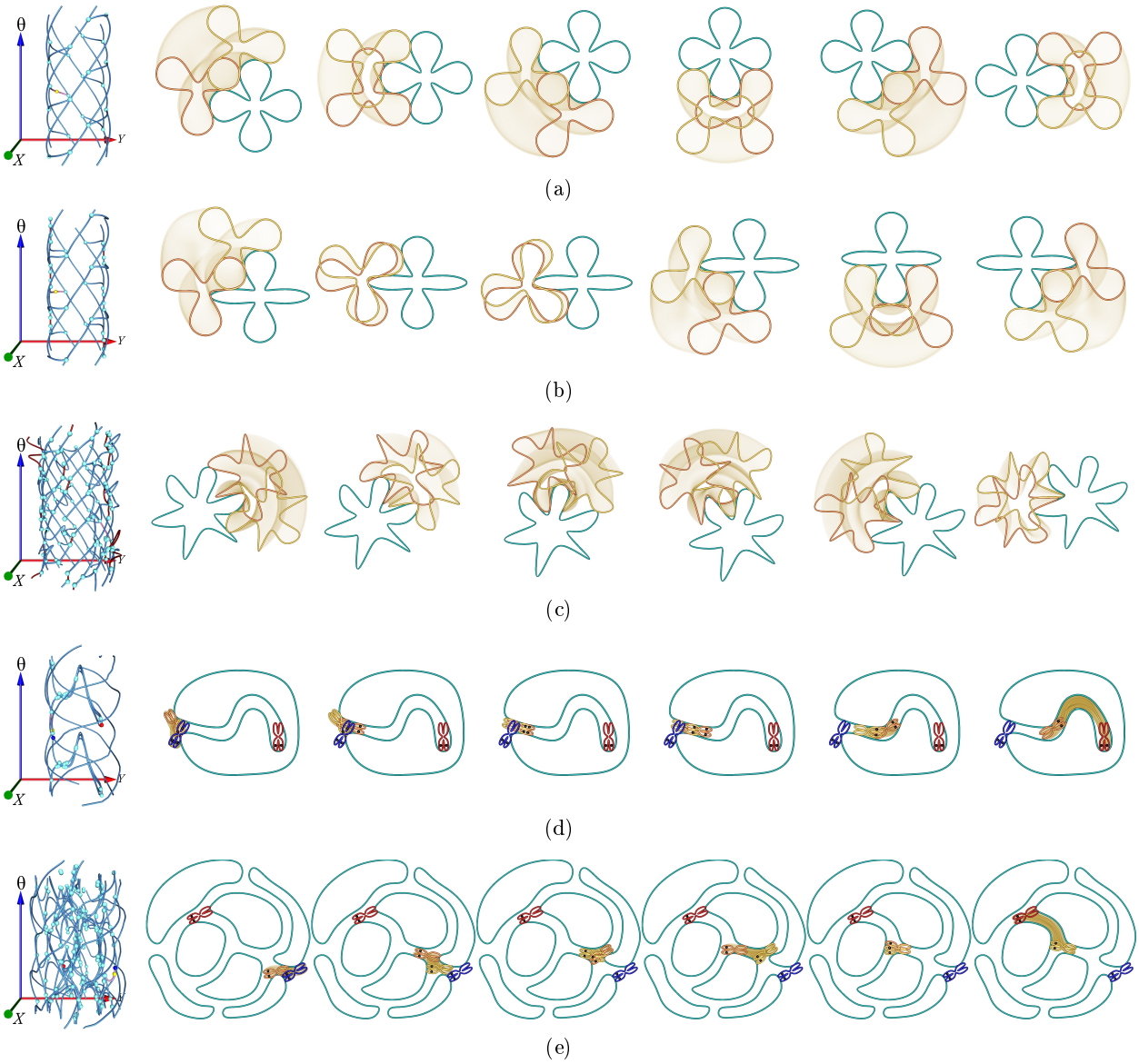

Figure 9: Continuous two-contact motions of the six test examples.

Table 1: Statistics on the six test examples in Figure 9.

Examples	#Ctrl Pts C	#Ctrl Pts D	#Global Triple Contacts	#Traces (Double Contacts)
9 (a)	15	20	24	36
9 (b)	15	20	24	42
9 (c)	18	20	77	140
9 (d)	20	35	28	54
9 (e)	20	94	214	208

Table 2: Timing and memory consumption of examples in Figure 9.

Examples	Pruning Time (s)	Double Contact Computation Time (s)	Triple Contact Computation Time (s)	Total (s)	Memory Usage (MB)
9 (a)	17.41	232.65	0.47	250.53	21.62
9 (b)	29.57	303.44	11.74	344.75	26.92
9 (c)	96.51	989.93	73.38	1159.82	88.48
9 (d)	30.41	1296.86	44.16	1371.43	35.88
9 (e)	178.47	4200.48	1269.69	5648.64	200.01

[6] R.C. Brost. Computing metric and topological properties of configuration space obstacles. *Proc. of 1989 IEEE Conf. on Robotics and Automation*, April 1989, pp. 170–176.

[7] R.C. Brost. Computing the possible rest configuration of two interacting polygons. *Proc. of 1991 IEEE Conf. on Robotics and Automation*, Sacramento, California, April 1991, pp. 686–693.

[8] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, and S. Thrun. *Principles of Robot Motion: Theory, Algorithms, and Implementation*, The MIT Press, Cambridge, MA, 2005.

[9] T. Cormen, C. Leiserson, R. Rivest, and C. Stein. *Introduction to Algorithms*, 3rd Ed., The MIT Press, 2009.

[10] E. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, **1**(1):269–271, 1959.

[11] B.R. Donald. A search algorithm for motion planning with six degrees of freedom. *Artificial Intelligence*, **31**(3):295–353, 1987.

[12] G. Elber and M.-S. Kim. Geometric constraint solver using multivariate rational spline functions. *Proc. of the 6th ACM Symp. on Solid Modeling and Applications*, Ann Arbor, Michigan, USA, June 4–8, 2001, pp. 1–10.

[13] I. Hanniel and G. Elber. Subdivision termination criteria in subdivision multivariate solvers using dual hyperplanes representations. *Computer-Aided Design*, **39**(5):369–378, 2007.

[14] J. Hoschek and D. Lasser. *Fundamentals of Computer Aided Geometric Design*, AK Peters, Wellesley, MA, 1989.

[15] IRIT 11.0 User's Manual, Technion, 2013. http://www.cs.technion.ac.il/ ∼irit.

[16] L. Joskowicz and E. Sacks. Computational kinematics. *Artificial Intelligence*, **51**:381–416, 1991.

[17] L. Joskowicz and E. Sacks. Configuration space computation for mechanical design. *Proc. of 1994 IEEE Conf. on Robotics and Automation*, April 1994, pp. 1080–1087.

[18] E. Larsen, S. Gottschalk, M.C. Lin, D. Manocha. Fast proximity queries using swept sphere volumes. Technical Report TR99-018, Dept. of Computer Science, UNC, 1999.

[19] J.-C. Latombe. *Robot Motion Planning*, Kluwer Academic Pub., Norwell, NY, 1991.

[20] S.M. Lavalle. *Planning Algorithms*, Cambridge University Press, New York, NY, 2006.

[21] I.-K. Lee, M.-S. Kim, and G. Elber. Polynomial/rational approximation of Minkowski sum boundary curves. *CVGIP: Graphical Models and Image Porcessing*, **60**(2):136–165, 1998.

[22] T. Lozano-Pérez. Automatic planning of manipulator transfer movements. *EEE Trans. on System, Man, and Cybernetics*, **11**(10):681–698, 1981.

[23] T. Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Trans. on Computers*, **32**(2):108–120, 1983.

[24] T. Lozano-Pérez and M.A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, **22**(10):560–570, 1979.

[25] V. Milenkovic, E. Sacks and S. Trac. Robust free space computation for curved planar bodies. *IEEE Trans. on Automation Science and Engineering*, **10**(4):875–883, 2013.

[26] V. Milenkovic, E. Sacks and S. Trac. Robust Complete Path Planning in the Plane. *Proc. of the Tenth Workshop on the Algorithmic Foundations of Robotics*, **86**:37–52, 2013.

[27] S. Nelaturi and V. Shapiro. Solving inverse configuration space problems by adaptive sampling. *Computer-Aided Design*, **45**(2):373–382, 2013.

[28] E. Sacks and L. Joskowicz. *The Configuration Space Method for Kinematic Design of Mechanisms*, The MIT Press, 2010.

[29] J.T. Scwhartz, M. Sharir and J. Hopcroft. *Planning, Geometry, and Complexity of Robot Motion*, Ablex Publishing Corporation, 1987.

[30] E.C. Sherbrooke and N.M. Patrikalakis. Computation of solution of non-linear polynomial systems. *Computer Aided Geometric Design*, **5**(10):379–405, 1993.

[31] M. Tang, M. Lee and Y. Kim Interactive Hausdorff distance computation for general polygonla models. *ACM Trans. on Graphics(SIGGRAPH)*, **28**(3), 2009.

[32] The result video http://youtu.be/tuNafC0hgws.

Appendices

Proofs for Algebraic Conditions

Lemma 2. *The algebraic condition for a curvature contact point of two-contact motion (4.1) is characterized as follows:*

$$k_C(u) - k_D(v) = 0, \tag{7.1}$$

$$k'_C(u) - k'_D(v) \cdot \frac{dv}{du} = 0, \tag{7.2}$$

$$\det(R_\theta(C'(u)), D'(v)) = 0, \tag{7.3}$$

where $k_C(u)$ and $k_D(v)$ are the curvature functions of $C(u)$ and $D(v)$, respectively.

Proof. Consider a two-contact case of some motion curve $(u_1(a), v_1(a), u_2(a), v_2(a), t(a))$ that converges to a single curvature contact point $(u^0, v^0, u^0, v^0, t^0) = (u_1(a^0), v_1(a^0), u_2(a^0), v_2(a^0), t(a^0))$. Now consider a point (u_1, v_1, u_2, v_2, t) that is sufficiently close to $(u^0, v^0, u^0, v^0, t^0)$. Without loss of generality, we may assume that the curve segment $C(u)$, $(u \in [u_1, u_2])$, is convex and $D(v)$, $(v \in [v_1, v_2])$, is concave. Obviously, $(u^0, v^0, u^0, v^0, t^0)$ satisfies the tangential condition (7.3). From Lemma 1, $k_C(u_1) > k_D(v_1)$, $k_C(u_2) > k_D(v_2)$ and $\exists u^* \in [u_1, u_2]$ and $v^* \in [v_1, v_2]$ such that $k_C(u^*) < k_D(v^*)$. Therefore, if (u_1, v_1) and (u_2, v_2) converges to (u^0, v^0), we have $k_C(u^0) = k_D(v^0)$.
Now consider a scalar function $f(a) = k_C(u_1(a)) - k_D(v_1(a))$. The function $f(a)$ is positive everywhere except $a = a^0$. Therefore, $f(a)$ has a local minimum at $a = a^0$ and thus $f'(a^0) = 0$.

$$f'(a^0) = k'_C(u_1(a^0)) \cdot \tfrac{\partial u_1}{\partial a} - k'_D(v_1(a^0)) \cdot \tfrac{\partial v_1}{\partial a} = 0,$$

$$\text{or } k'_C(u^0) - k'_D(v^0) \cdot \tfrac{dv}{du} = 0.$$

□