Precise Contact Motion Planning for Deformable Planar Curved Shapes

Yong-Joon Kim[a], Gershon Elber[a], Myung-Soo Kim[b]

[a]*Department of Computer Science, Technion, Israel*
[b]*School of Computer Science and Eng, Seoul National University, Korea*

Abstract

We present a precise contact motion planning algorithm for a deformable robot in a planar environment with stationary obstacles. The robot and obstacles are both represented with C^1-continuous implicit or parametric curves. The robot is changing its shape using a single degree of freedom (via a one-parameter family of deformable curves). In order to reduce the dimensionality of the configuration space, geometrically constrained yet collision free contact motions are sought, that have $K(= 2, 3)$ simultaneous tangential contact points between the robot and the obstacles. The K-contact motion analysis effectively reduces the degrees of freedom of the robot, which enables a more efficient motion planning. The geometric conditions for the K-contact motions can be formulated as a system of non-linear polynomial equations, which can be solved precisely using a multivariate equation solver. The solutions for K-contact motions are represented as curves in a 4-dimensional (x, y, θ, t) space, where x, y, θ are the degrees of freedom of the rigid motion and t is the deformation's parameter. Using the graph structure of the solution curves for the K-contact motions, our algorithm efficiently finds a feasible path connecting two configurations via a graph searching algorithm, whenever available. We demonstrate the effectiveness of the proposed approach using several examples.

Keywords: Configuration spaces, multivariate algebraic constraints, freeform geometric models, B-spline curves, deformable robots.

1. Introduction

The problem of collision-avoidance in robot motion planning has been an active research area over the last several decades. Remarkable progress has been achieved in numerous practical applications. However, relatively few results were introduced for the case of deformable robots, even though potentially many applications can benefit from the added degrees of freedom. The deformation ability of the robot provides more flexibility in the motion planning and enables the successful navigation of more challenging tasks that cannot be accomplished by rigid robots. Actual designs of deformable robots already exist, i.e. [1]. On the other hand, more degrees of freedom in the robot exponentially increase the complexity of motion planning algorithms. As a consequence, it is very difficult to find feasible motions in a reasonable amount of time.

The geometry of deformable robots and their environments are often designed with Non-Uniform Rational B-spline (NURBS) curves and surfaces, which is the de facto standard representation for industrial objects. Yet, the majority of contemporary algorithms for motion planning first tessellate the freeform NURBS curves and surfaces, as they can handle only piecewise linear discrete objects. The errors caused by the polygonal approximation are difficult to control. Moreover, it is non-trivial to precisely manage the collision detections (in particular when dealing with contact motions) using these polygonal approximations. Thus, it is highly desirable to directly process the NURBS curves and surfaces for applications requiring high accuracy.

In this paper, we consider the precise (up to machine precision) contact motion planning for a deformable planar parametric or implicit freeform C^1-continuous robot $\Phi_t(C(u))$, with respect to a C^1-continuous stationary (set of) parametric obstacle(s) $D(v)$, in the plane. The transformation Φ_t represents a one-parameter smooth freeform deformation of the C^1-continuous robot $C(u)$ and we assume that Φ_t is pre-defined algebraically. The deformable robot $\Phi_t(C(u))$ has two rigid motion degrees of freedom in translation, (x, y), and one rigid motion degree of freedom in rotation, θ. Finally, the last degree of freedom t provides the shape control over the robot's deformation function Φ_t. The configuration space (C-space) is thus a 4-dimensional space, in (x, y, θ, t).

A naive approach for planning the motion of such a deformable robot is to compute the entire boundary of the C-space's obstacle which can be represented as an implicit 3-manifold, $f(x, y, \theta, t) = 0$, and use this 3-manifold for the motion planning. The optimal motion path may then be computed by considering all possible motion paths over the entire boundary of the obstacle's C-space. Clearly, computing and even representing the entire 3-manifold solution is expected to be highly challenging. It is indeed inefficient and in fact unnecessary to compute the entire boundary of the C-space, because the robot typically follows a univariate motion path and thus only a small portion of the C-space is used for the motion planning.

Therefore, instead of computing the entire C-space, we focus on analyzing contact motions that satisfy additional geometric constraints so that the dimension of the computed solution can be significantly reduced. Toward this end, we seek the collision free motion of a (deformable) robot $\Phi_t(C(u))$ while it maintains

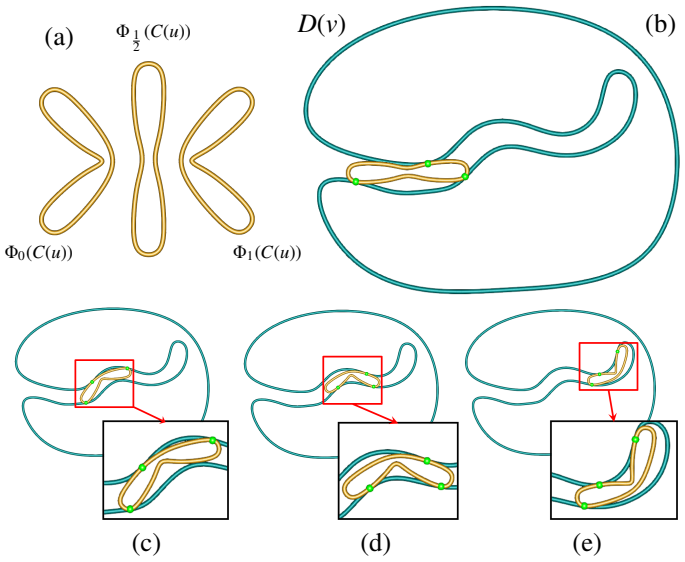


Figure 1: K-contact motion planning for a deformable robot: (a) a deformable robot $\Phi_t(C(u))$ and the shape of $\Phi_t(C(u))$ for $t = 0, \frac{1}{2}, 1$. (b) The robot with constant deformation parameter ($t = \frac{1}{2}$) cannot go deeper into obstacle $D(v)$ without gouging at the 3-contact configuration. (c)-(e) A 3-contact motion analysis between robot $\Phi_t(C(u))$ and obstacle $D(v)$ enlarges the accessible region of the, now deformable, robot.

multiple tangential contacts, typically two or three, with obstacles $D(v)$. We denote such motions $K(= 2, 3)$-contact motions. The reasons for using K-contact motions, in the motion planning, are twofold:

1. We can now effectively reduce the degrees of freedom in the contact motion analysis. Now, every K-contact motion will be represented as a curve in a 4-dimensional C-space. As a result, the entire set of K-contact motions forms a graph structure in the (x, y, θ, t) space and the motion planning can be addressed via well-known graph searching algorithms.
2. A K-contact motion analysis often provides a good solution for narrow passage problems, because there is a high probability for the robot to have multiple contact points in such narrow passages. Figure 1 shows an example of a deformable robot with K-contact motion that follows a narrow passage.

The rest of this paper is organized as follows. In Section 2, we briefly review previous related work. Section 3 describes the different ways one can prescribe the deformation of the robot, algebraically. Section 4 introduces the algebraic conditions for the K-contact motion between $\Phi_t(C(u))$ and $D(v)$. Section 5 addresses the construction of the K-contact motion graph and the motion planning algorithm using this K-contact graph. Several experimental results are reported in Section 6 and the paper is finally concluded with discussions on future work in Section 7.

2. Related Work

Algorithms for motion planning of deformable robots are typically based on the probabilistic road-map (PRM) approach [2]. A PRM planner samples random points in the C-space and generates an approximated C-space graph by connecting adjacent sample points. Then, the planner finds a feasible motion by connecting these points on the graph via a graph searching algorithm. The performance and quality of PRM based algorithms is heavily dependent on the sampling strategy. Guibas et al. [3] proposed an efficient motion planning algorithm for flexible objects. By deforming the robot as closely as possible to the medial axis of the workspace, the algorithm successfully finds critical deformations and effectively reduces the deformation space. However, the medial axis computation for 3D objects is computationally expensive and fitting the object to the 3D medial axis is non-trivial. Bayazit et al. [4] sampled configurations that might cause inter-penetration into the obstacles and then generated a collision-free path by locally deforming the robot. The deformation of the robot is employed only for preventing the inter-penetration of the robot and thus limits the exploration of the deformation space. Gayle et al. [5] developed a practical algorithm for motion planning of deformable robots in complex environments, which can take into account geometric and physical constraints. The deformation of the robot should satisfy some imposed constraints that are formulated as an energy minimization problem and be solved via an optimization algorithm. This optimization can be inefficient if the geometry of the robot and the environments are not similar. Mahoney et al. [6] tackled the problem of motion planning of deformable robot by reducing the dimension of the deformation space via principal component analysis over the deformation space. The entire deformation space is reduced to the subspace spanned by a small number of basis elements and the degrees of freedom in C-space are significantly reduced. The problem is that, in many cases, the reduced deformation space suffers from narrow passages of the obstacles.

The above results focus on polygonal representation and typically produce approximate solutions. The previous work on precise motion planning for NURBS curves and surfaces is limited. Bajaj and Kim [7, 8, 9] considered the generation of C-space obstacles for translational motions of rigid algebraic curves and surfaces. For the case of a rigid planar freeform shapes moving (with translation only) among similar static freeform curves in the plane, Lee et al. [10] presented a high-precision algorithm that can approximate the boundary of the obstacle's C-space using B-spline planar curves. Holleman et al. [11] and Lamiraux et al. [12] applied a PRM planner for path planning of flexible surface patches modeled with low degree Bezier surfaces. Holleman et al. [11] enforced a desired deformation of the surface by formulating the energy function and Lamiraux et al. [12] handled a specialized deformation having only one degree of freedom. Milenkovic et al. [13, 14] developed a robust algorithm for constructing the C-space obstacles for a rigid planar moving object and obstacles bounded by circular arcs. Recently, Kim et al. [15] presented an algorithm for computing the precise contact motion between rigid planar freeform curves. In this work, we extend the approach of Kim et al. [15] to the case of deformable robot.

Designing the proper deformation of the robot is crucial for the generation of feasible motions. We employ shape morph-

ing techniques (also known as metamorphosis) for designing the deformation of the robot. The algorithms for shape morphing can be classified into two main categories: parametric correspondence and interpolation [16, 17] and implicit function interpolation [18, 19]. The parametric methods directly interpolate two (or more) shapes using the correspondence between them and are therefore relatively faster and more efficient than methods for implicit functions. On the other hand, implicit methods interpolate the implicit representations of shapes and have no constraints on the topological similarity of the deformed shapes (see Figure 2 for an example). The motion planning algorithm for deformable robots that we present in this work can handle both types of shape deformation interpolations.

The presented algorithms for motion planning of continuous NURBS curves and surfaces involve solving a system of non-linear equations. Based on the analytical representation of NURBS, the desired geometric constraints can be converted to a set of implicit multivariate equations via symbolic computation. Sherbrooke and Patrikalakis [20] proposed an early subdivision-based approach for solving a system of multivariate equations. Elber and Kim [21] and Hanniel and Elber [22] improved the subdivision based method by introducing a simple termination condition into the subdivision process. Barton et al. [23] presented an efficient algorithm for handling the special case of univariate solution spaces. In this work, we employ the approach of [23] to solve the presented systems of non-linear constraints.

3. Deformable Robots

We consider two general ways of prescribing a planar deforming shape, algebraically. Parametric forms are blended to generate a parametric deformed shape, and implicit forms are also blended to generate implicits under deformation. We briefly discuss both approaches below.

3.1. Parametric Deformable Robots

Let $C_i(u_i)$, $(i = 1, 2)$, $u_i \in [0, 1]$, be two regular smooth parametric curves. Then, let

$$\Phi_t(C(u)) = (1 - t)C_1(u) + tC_2(u). \tag{1}$$

In essence, $\Phi_t(C(u))$ is a ruled surface between $C_1(u)$ and $C_2(u)$. However, any one-parameter family of curves can be equally used. Specifically, any bivariate surface $S(u, t)$ can serve as the shape deformation function:

$$\Phi_t(C(u)) = S(u, t), \tag{2}$$

where Equation (1) is just one possible (ruled surface) instance of Equation (2).

3.2. Implicit Deformable Robots

Let $C_i(x, y) = 0$, $(i = 1, 2)$, be two smooth implicit curves. Then,

$$\Phi_t(C(x, y)) = (1 - t)C_1(x, y) + tC_2(x, y) = 0, \tag{3}$$

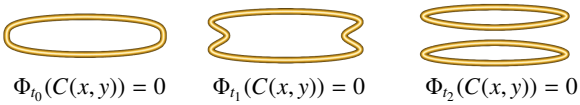


Figure 2: Topological changes of an implicitly defined deformable robot.

prescribes an implicit curve that identifies with $C_1(x, y)$, for $t = 0$, and with $C_2(x, y)$, for $t = 1$.

In essence, $\Phi_t(C(x, y))$ is a ruled volume between bivariates $C_1(x, y)$ and $C_2(x, y)$. However, any one-parameter family of bivariates can be equally used. Specifically, any trivariate volume $V(x, y, t)$ can serve as the shape deformation function:

$$V(x, y, t) = 0, \tag{4}$$

where Equation (3) is just one possible (ruled volume) instance of Equation (4).

Implicit blends have the advantage that they may change the topology as a part of the deformation; thus, the two shapes $C_i(x, y) = 0$, $(i = 1, 2)$, are not constrained to have the same topology (see Figure 2). On the other hand, the implicit form is typically more difficult to handle and approximated solutions are typically employed.

Both the parametric and implicit deformation forms yield algebraic constraints that can be exploited in this work. This ability will be demonstrated in the coming sections. In the ensuing discussion, we assume the deformable robots are always smooth and regular.

4. Algebraic Conditions for K-Contact Motion

In this section, we present the algebraic conditions for K-contact configurations between a deformable robot $\Phi_t(C(u))$ and stationary obstacle $D(v)$. Consider an orientable planar C^1-continuous regular parametric deformable curve $\Phi_t(C(u)) = (\Phi_t(C(u))_x, \Phi_t(C(u))_y)$, where t is the deformation parameter, and a stationary obstacle given as a C^1-continuous parametric curve $D(v) = (D(v)_x, D(v)_y)$, $0 \le u, v, t \le 1$. Now consider the rigid transformation of $\Phi_t(C(u))$, $T[\Phi_t(C(u))] = R_\theta[\Phi_t(C(u))] + (x, y)$, $T = T(x, y, \theta)$, where (x, y) and θ represent the planar translation and rotation degrees of freedom, respectively. When $T[\Phi_t(C(u))]$ and $D(v)$ have K contact points at $T[\Phi_t(C(u_i))]$ and $D(v_i)$, $i = 1, \cdots, K$, these conditions can be formulated by the following $3K$ equations:

$$\begin{aligned}
0 &= R_\theta[\Phi_t(C(u_i))]_x + x - D(v_i)_x, \\
0 &= R_\theta[\Phi_t(C(u_i))]_y + y - D(v_i)_y, \\
0 &= F_i(u_i, v_i, \theta, t) \\
&= R_\theta[\Phi_t(C'(u_i))] \times D'(v_i), \qquad \text{for } i = 1, \cdots, K.
\end{aligned} \tag{5}$$

The first two constraints in Equations (5) ensure the two curves share a contact location in the plane. The last constraint makes sure this contact is being tangential.

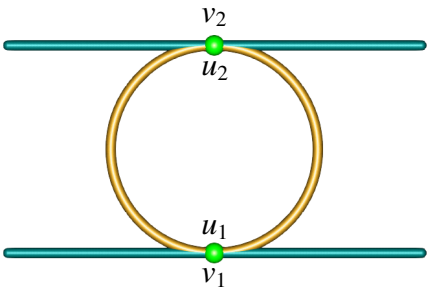


Figure 3: Singular case for the constraint system of K-contact motion.

Isolating x and y in the first two constraints in Equations (5), we get:

$$\begin{aligned} x &= G_i(u_i, v_i, \theta, t) \\ &= D(v_i)_x - R_\theta[\Phi_t(C(u_i))]_x, \\ y &= H_i(u_i, v_i, \theta, t) \\ &= D(v_i)_y - R_\theta[\Phi_t(C(u_i))]_y. \end{aligned} \tag{6}$$

The linear terms, x and y, in Equations (6), can be eliminated from Equations (5), for $i = 2, \cdots, K$, by replacing x and y with $G_1(u_1, v_1, \theta, t)$ and $H_1(u_1, v_1, \theta, t)$, respectively. Consequently, we get the following $3K - 2$ constraints in $2K + 2$ variables (u_i, v_i, θ, t):

$$\begin{aligned} 0 &= G_1(u_1, v_1, \theta, t) - G_i(u_i, v_i, \theta, t), \text{ for } 2 \le i \le K, \\ 0 &= H_1(u_1, v_1, \theta, t) - H_i(u_i, v_i, \theta, t), \text{ for } 2 \le i \le K, \\ 0 &= F_i(u_i, v_i, \theta, t), \qquad \text{for } 1 \le i \le K. \end{aligned} \tag{7}$$

In this work, we consider the analysis for two types of univariate contact motions, 2-contact and 3-contact univariate motions:

- For a rigid robot, the 2-contact ($K = 2$) motion analysis has a univariate solution. Because the deformation parameter t is fixed, we have four constraints in five variables $(u_1, v_1, u_2, v_2, \theta)$.
- If the robot is under a one-parameter family of deformations, the 3-contact ($K = 3$) motion analysis has a univariate solution. Here, we have seven constraints in eight variables $(u_1, v_1, u_2, v_2, u_3, v_3, \theta, t)$.

There exist special cases where the constraints for the K-contact configuration are dependent on each other. Consider a 2-contact motion between a rigid unit circle and two parallel line segments (Figure 3). Then, we have the following four constraint equations to represent this $K = 2$ condition:

$$\begin{aligned} 0 &= G_1(u_1, v_1, \theta, t^*) - G_2(u_2, v_2, \theta, t^*), \\ 0 &= H_1(u_1, v_1, \theta, t^*) - H_2(u_2, v_2, \theta, t^*), \\ 0 &= F_1(u_1, v_1, \theta, t^*), \\ 0 &= F_2(u_2, v_2, \theta, t^*), \end{aligned}$$

where t^* is a fixed deformation parameter. This system of algebraic equations has four constraints in five variables $(u_1, v_1, u_2, v_2, \theta)$ and thus it is supposed to have a univariate solution set. However, in this case, the robot can rotate in place or slide left/right while maintaining a 2-contact configuration. As a result, this system of algebraic equations has a bivariate solution space. Special cases, with dependent constraint equations near/at zeros, are referred to as singularities. In the rest of this paper, we assume that there are no singularities in the constraint systems for the K-contact motions.

Before concluding this section, let us also comment on the required change if the deformable robot is represented implicitly. Having an implicit robot $\Phi_t(C(\mathcal{X}, \mathcal{Y})) = 0$, and assuming the obstacle $D(v)$ is parametric as before, Equations (5) now becomes, for the i-th contact:

$$\begin{aligned} 0 &= \Phi_t(C(\mathcal{X}_i, \mathcal{Y}_i)), \\ 0 &= \mathcal{X}_i + R_\theta^{-1}[(x, y) - D(v_i)]_x, \\ 0 &= \mathcal{Y}_i + R_\theta^{-1}[(x, y) - D(v_i)]_y, \\ 0 &= L_i(\mathcal{X}_i, \mathcal{Y}_i, \theta, t), \\ &= \left\langle \nabla\Phi_t(C(\mathcal{X}_i, \mathcal{Y}_i)), R_\theta^{-1}[D'(v_i)] \right\rangle, \end{aligned} \tag{8}$$

prescribing a tangential contact at a location $(\mathcal{X}_i, \mathcal{Y}_i)$, where ∇ denotes the gradient operator. The first constraint in Equations (8) ensures that the (deformed) implicit is satisfied at $(\mathcal{X}_i, \mathcal{Y}_i)$. The second and third constraints ensure that the translation (x, y) and the (inverse) rotation θ applied to $D(v_i)$ will also equate with $(\mathcal{X}_i, \mathcal{Y}_i)$. Finally, the last constraint in Equations (8) guarantees that the contact is, again, tangential.

We can further reduce the number of variables and equations by isolating the variables as we did for the parametric robot:

$$\begin{aligned} x &= M_i(\mathcal{X}_i, \mathcal{Y}_i, \theta), \\ &= D(v_i)_x - R_\theta[(\mathcal{X}_i, \mathcal{Y}_i)]_x, \\ y &= N_i(\mathcal{X}_i, \mathcal{Y}_i, \theta), \\ &= D(v_i)_y - R_\theta[(\mathcal{X}_i, \mathcal{Y}_i)]_y. \end{aligned} \tag{9}$$

If the implicit robot is defined by a linear interpolation of two implicit curves as in Equation (3), we can also isolate parameter t as follows:

$$\begin{aligned} t &= P_i(\mathcal{X}_i, \mathcal{Y}_i), \\ &= \frac{C_1(\mathcal{X}_i, \mathcal{Y}_i)}{C_1(\mathcal{X}_i, \mathcal{Y}_i) - C_2(\mathcal{X}_i, \mathcal{Y}_i)}. \end{aligned} \tag{10}$$

The isolated variables, x, y, and t, can be eliminated from Equations (8) by replacing x, y, t with $M_1(\mathcal{X}_1, \mathcal{Y}_1, v_1, \theta)$, $N_1(\mathcal{X}_1, \mathcal{Y}_1, v_1, \theta)$, $P_1(\mathcal{X}_1, \mathcal{Y}_1)$, respectively. Now we have the following $4K - 3$ constraints in $3K + 1$ variables $(\mathcal{X}_i, \mathcal{Y}_i, v_i, \theta)$:

$$\begin{aligned} 0 &= M_1(\mathcal{X}_1, \mathcal{Y}_1, v_1, \theta) - M_i(\mathcal{X}_i, \mathcal{Y}_i, v_i, \theta), \text{ for } 2 \le i \le K, \\ 0 &= N_1(\mathcal{X}_1, \mathcal{Y}_1, v_1, \theta) - N_i(\mathcal{X}_i, \mathcal{Y}_i, v_i, \theta), \text{ for } 2 \le i \le K, \\ 0 &= P_1(\mathcal{X}_1, \mathcal{Y}_1) - P_i(\mathcal{X}_i, \mathcal{Y}_i), \qquad \text{for } 2 \le i \le K, \\ 0 &= L_i(\mathcal{X}_i, \mathcal{Y}_i, v_i, \theta), \qquad \text{for } 1 \le i \le K. \end{aligned} \tag{11}$$

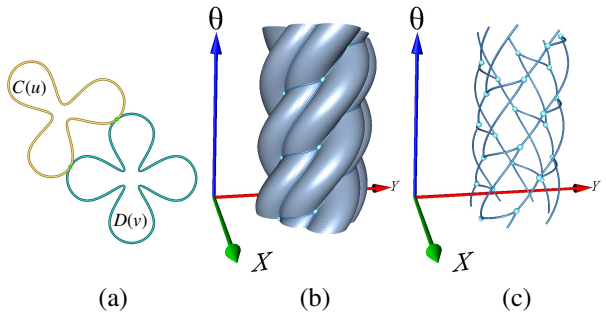


Figure 4: C-space obstacle boundary and 2-contact motion graph: (a) the rigid robot $C(u)$ and the obstacle $D(v)$, (b) C-Space obstacle boundary between $C(u)$ and $D(v)$, (c) 2-contact motion graph embedded on the boundary of the obstacle's C-space.

In the following sections, we present an efficient algorithm for computing these 2- and 3-contact motions.

5. The Generation of the *K*-contact Motion Graph

The graph we build consists of 2- and 3-contact motions. We first compute the 2-contact motion curves for a rigid robot, fixing the deformation parameter as $t = t^*$. For the sake of simplicity, we assume $\Phi_{t^*}(C(u)) = C(u)$. The 2-contact motion curves are derived using the algorithm described in Kim et al. [15]. We then compute the necessary 3-contact motion curves based on the results of 2-contact motion analysis. In this section, we briefly review the algorithm of Kim et al. [15] only to discuss how to extend this algorithm to the case of 3-contact motion, for a deformable robot.

5.1. 2-Contact Motion Curves

Consider the boundary of the C-space between a rigid robot $C(u)$ and an obstacle $D(v)$, which is a 2-manifold in the three-dimensional space (x, y, θ). Each point on the boundary of the C-space obstacle corresponds to a collision-free contact configuration. The 2-contact configuration between $C(u)$ and $D(v)$ can be characterized as the self-intersections of the 2-manifold boundary of the C-space obstacle. Each of these self-intersection curve segments has two end points that correspond to 3-contact configurations between $C(u)$ and $D(v)$. The 2-contact motion curves thus have a layout which can be represented in a graph structure, where each vertex corresponds to a 3-contact configuration and each edge corresponds to a 2-contact motion curve. (See Figure 4).

5.2. 3-Contact Motion Curves

If the 2-contact motion graph of a rigid robot has disconnected components (see the leftmost column, in Figure 8), there is no valid 2-contact motion that connects these isolated components. In such a case, it is impossible for a rigid robot to move between the disconnected components. However, we may connect these disconnected components by allowing the robot to deform. Then, the univariate solution space for the 2-contact motion expands to a bivariate solution space, having t as a new degree of freedom.

To take advantage of the possibility of robot deformation while aiming to keep the univariate graph structure of the motion space, we employ a 3-contact motion analysis for deformable robots. As pointed out in Section 4, the 3-contact motion analysis for a robot with a one-parameter family of deformations also has a univariate solution space. Hence, this analysis can provide a new type of collision-free contact motion which involves simultaneous deformations of the robot. Indeed and as will be shown later on, in many cases, this deformable robot's 3-contact motion is able to connect disconnected components of the rigid robot 2-contact motion graph (see the second column from the left, in Figure 8). In this section, we present the algorithm for computing these deformable robot's 3-contact motions, based on the result of the 2-contact motion graph of Kim et al. [15].

5.2.1. Computing the 3-Contact Motion Curves

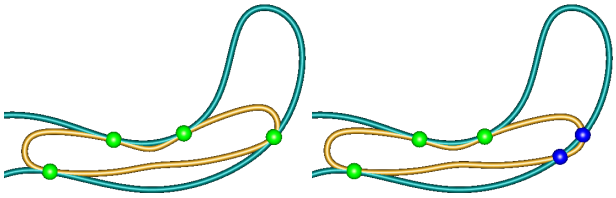
As discussed in Sections 4 and 5, the algebraic constraints for 3-contact planar motion of a deformable robot have seven equations with eight variables $(u_1, v_1, u_2, v_2, u_3, v_3, \theta, t)$, for a parametrically deformed robot. Due to its high dimensionality and the corresponding large solution space, we strive to exploit the 2-contact motion analysis for a rigid robot as much as possible, and resort to a 3-contact analysis over a deformable robot only when necessary, achieving a better performance. The 2-contact motion graph for a rigid robot includes 3-contact configurations as vertices of the graph. These 3-contact configurations are locations where the rigid robot is tangential to the obstacle, in three different places. We will take these 3-contact configurations as the locations where the deformable robot is tangential to the obstacle, in three different places, when $t = t^*$, the fixed deformation parameter. In other words, we use these 3-contact configurations as seed points for tracing the 3-contact deformable robot motion, by solving the algebraic constraint equations for 3-contact motions of the deformable robot, in the local neighborhood of these seed points, for which $t = t^*$.

For each 3-contact $(u_1, v_1, u_2, v_2, u_3, v_3, \theta, t^*)$ and a user specified threshold δ, we extract a sub-domain $\mathcal{D}$:

$$\begin{aligned}\mathcal{D} = &[u_1 - \delta, u_1 + \delta] \times [v_1 - \delta, v_1 + \delta] \times \\ &[u_2 - \delta, u_2 + \delta] \times [v_2 - \delta, v_2 + \delta] \times \\ &[u_3 - \delta, u_3 + \delta] \times [v_3 - \delta, v_3 + \delta] \times \\ &[\theta - \delta, \theta + \delta] \times [t^* - \delta, t^* + \delta],\end{aligned}$$

only to solve the algebraic equations for the sub-domain $\mathcal{D}$, which is much smaller than the entire domain and thus can be computed more efficiently. The computed univariate solution for the sub-domain $\mathcal{D}$ can be parameterized as $(u_1(a), v_1(a), u_2(a), v_2(a), u_3(a), v_3(a), \theta(a), t(a))$ for $0 \le a \le 1$.

The two end points of the solution are typically on the boundary of the domain $\mathcal{D}$. Let one of the end points be $E = ({u_1}^e, {v_1}^e, {u_2}^e, {v_2}^e, {u_3}^e, {v_3}^e, \theta^e, s^e)$. Since E is on the boundary of $\mathcal{D}$, at least one coordinate of E is a minimum or a maximum value of the domain of $\mathcal{D}$. Assume, without loss of generality, the coordinate is ${u_1}^e = u_1 + \delta$. We then continue the solution tracing and march to the next domain, $\mathcal{D}_2$, which can be set as



(a) (b)

Figure 5: (a) A 4-contact configuration, (b) inter-penetration of $\Phi_t(C(u))$ into $D(v)$. Of the geometry in Figure 8 (a).

follows:

$$\begin{aligned}\mathcal{D}_2 = &[u_1, u_1 + 2\delta] \times [v_1 - \delta, v_1 + \delta] \times \\ &[u_2 - \delta, u_2 + \delta] \times [v_2 - \delta, v_2 + \delta] \times \\ &[u_3 - \delta, u_3 + \delta] \times [v_3 - \delta, v_3 + \delta] \times \\ &[\theta - \delta, \theta + \delta] \times [t^* - \delta, t^* + \delta].\end{aligned}$$

We repeat this tracing procedure, solving algebraic equations for one sub-domain and then deciding the next sub-domain to solve, until the traced 3-contact motion curve meets one of several terminal points which are described in the next section.

5.2.2. Terminal Points for 3-Contact Motion

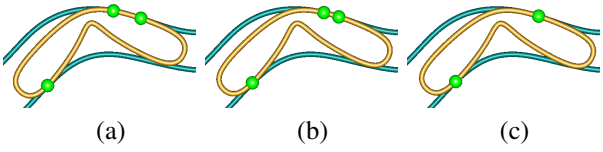
Typical terminal points for the 3-contact motion curve of a deformable robot is a 4-contact configuration location (see Figure 5(a)). The 4-contact configuration is realized when the robot undergoing a 3-contact motion gets in a tangential contact with the obstacle in a fourth location. The algebraic conditions for this 4-contact configuration can be formulated using Equations (7), for $K = 4$ with $3K-2 = 10$ constraints in $2K+2 = 10$ variables, $(u_1, v_1, u_2, v_2, u_3, v_3, u_4, v_4, \theta, t)$.

During the 3-contact motion computation, we keep checking whether the robot has any collision with the obstacle. The inter-penetration of the robot can be checked by computing the intersection point between the robot and the obstacle. Since the robot already has three tangential contacts with the obstacle, we check if the tangent directions at the intersection point are parallel or not, to distinguish the current three tangent points from the new intersection point.

Once we detect an inter-penetration at $(u_1(a), v_1(a), u_2(a), v_2(a), u_3(a), v_3(a), \theta(a), t(a))$, we extract a domain $\mathcal{D}^F$ around this location as follows:

$$\begin{aligned}\mathcal{D}^F = &[u_1(a) - \delta, u_1(a) + \delta] \times [v_1(a) - \delta, v_1(a) + \delta] \times \\ &[u_2(a) - \delta, u_2(a) + \delta] \times [v_2(a) - \delta, v_2 + \delta] \times \\ &[u_3(a) - \delta, u_3(a) + \delta] \times [v_3(a) - \delta, v_3 + \delta] \times \\ &[u_s, u_e] \times [v_s, v_e] \times \\ &[\theta(a) - \delta, \theta(a) + \delta] \times [t(a) - \delta, t(a) + \delta],\end{aligned}$$

where $[u_s, v_s]$ and $[u_e, v_e]$ are the parameters of the inter-penetration location (see the blue points in Figure 5(b)). We then solve the algebraic equations for the 4-contact configuration in the domain $\mathcal{D}^F$.



(a) (b) (c)

Figure 6: Curvature contact point: the two different contact points merges to a single point. Of the geometric configuration in Figure 8 (a).

Similarly to the 2-contact motion curve of a rigid robot, the 3-contact motion curve of a deformable robot has singular terminal points in the form of order-four (curvature derivative) contact points. The curvature derivative contact configuration is realized when two different contact points merge to a single location. Figure 6 shows one example of an order-four terminal contact. Two order-two (tangential) contact locations coalesce into one order-four (curvature derivative) contact. We can formulate this event by the following three equations in three variables (u, v, t):

$$\begin{aligned}\kappa_C(u,t) - \kappa_D(v) &= 0, \\ \frac{\partial \kappa_C(u,t)}{\partial u} - \frac{\partial \kappa_D(v)}{\partial v}\frac{\partial v}{\partial u} &= 0, \\ \frac{\partial \kappa_C(u,t)}{\partial t} - \frac{\partial \kappa_D(v)}{\partial v}\frac{\partial v}{\partial t} &= 0,\end{aligned} \tag{12}$$

where $\kappa_C(u,t)$ and $\kappa_D(v)$ are the curvature fields of $\Phi_t(C(u))$ and $D(v)$, respectively. Stated differently, Equations (12) identify locations on $\Phi_t(C(u))$ and $D(v)$ that share the same curvature and curvature derivative. Using a rigid motion, one can always bring these locations together so that they coalesce in their position and tangent.

Finally, a 3-contact motion curve may also terminate when it reaches its extreme deformation, i.e. for $t = 0$ or $t = 1$. While the other parameters $u_1, v_1, ..., \theta$ are all periodic for C^1 closed curves, here caution must also be taken to handle these extreme deformations cases.

5.2.3. Constructing the 3-Contact Motion Graph

We start from the seed points which are 3-contact vertices in the graph of the 2-contact motion of rigid robot, solve the algebraic constraints for the 3-contact motion curve of a deformable robot in the local neighborhood of these seed points and trace the 3-contact motion. When a terminal 4-contact configuration is reached during this tracing, we must fork out multiple branches for new traces. We now explain this forking-out process.

When a rigid planar robot is in a 3-contact with its obstacle(s), there are three different 2-contact motion routes that the robot can get out without collision. This is because there are $3 = \binom{3}{2}$ different ways to select 2-contact points out of the three (without repetitions). When we trace a 2-contact motion to a terminal 3-contact configuration, we already exhausted one such path (where we came from) and so two new paths are to be forked and traced out. Similarly, a deformable planar robot in a 4-contact configuration with its obstacle(s) has $4 = \binom{4}{3}$ different ways to select a 3-contact. One path (where we came from) is

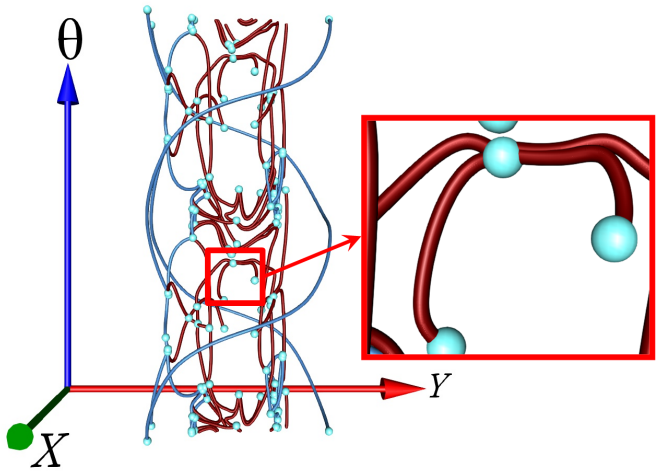


Figure 7: Four 3-contact curves forking out from 4-contact configuration and zoom in of four 3-contact curves from 4-contact configuration of the geometry in Figure 8 (d).

again already exhausted and so three new paths are to be forked out and traced from every 4-contact location of a deformable robot. As is shown in Figure 7, typically four 3-contact motion curves branch out from the 4-contact configuration location.

Finally, when the 3-contact traced motion curve of a deformable robot terminates at the curvature derivative contact or at an extreme deformation point (i.e., $t = 0$ or 1), we simply stop the tracing process.

When solving the entire constraint system via a multivariate equation solver [23], one can guarantee to find all solutions up to some prescribed tolerance. On the other hand, when we locally trace the 3-contact motion, we examine a considerably smaller domain space while we are not guaranteed to find all possible 3-contact motions. However, it effectively finds 3-contact motions that connect the fully disclosed but possibly disconnected 2-contact univariate components in the C-space (solved using [23]).

6. Experimental Results

We have implemented the presented contact motion planning algorithm for deformable robots in C++ and using the IRIT solid modeling system [24] on an Intel Core i7 3.4 GHz PC with 32 GB main memory. To demonstrate the effectiveness of our algorithm, we performed the motion planning for obstacles having narrow passages.

Figure 8 (a)-(d) shows four examples. The first three examples employ a parametrically define robot while the last example, in Figure 8 (d), shows an example of an implicitly defined robot. In the leftmost column, the 2-contact motion graph of a rigid robot is shown. This graph has disconnected components. The second column from the left augments the 2-contact graph with 3-contact motion paths of a deforming robot (in red) which connects the isolated components. The three columns on the right show samples of the computed motion using the combined 2- and 3-contact motion. Using the graph structure of 2- and 3-contact motions and Dijkstra's algorithm [25], non-trivial motion paths can be efficiently computed.

Table 1 presents some statistics for these four examples. To compare our local tracing algorithm for 3-contact motions with a general algorithm (that is based on all possible constraints), we solved the entire system of equations for Example 2 (shown in Fig. 8(b)) which is the fastest among all test examples considered in this work. Because the construction of an entire system of equations requires a huge amount of memory space, which is far beyond the capability of modern workstations, we had to subdivide input curves into Bézier pieces and then constructed a system of equations for each sub-domain. It took about 295 hours to solve the entire domain and it is more than 70 times slower than our local tracing algorithm. We report the average fraction of examined domain for each variable. The domain for each variable including θ is scaled to $[0, 1]$ and thus the entire domain space can be represented as $[0, 1]^8$ for the parametric robot and $[0, 1]^{10}$ for the implicit robot. We added the sizes of the sub-domains that were examined by our 3-contact motion computation algorithm and then computed the d-root (d is the dimension of the space of the domain) of the summation. As can be seen from the results, our algorithm examined less than 5% of each dimension of the domain on average, which means that the domain examined by our algorithm is less than 0.05^8 (0.05^{10} for the implicit robot) of the entire domain. Even though our algorithm examined only a tiny fraction of the entire domain, enabling a significant performance improvement, the computation times are still in the range of many hours. Nevertheless, the algorithm effectively and precisely produced 3-contact motions that connect disconnected components of the rigid robot 2-contact motion graph.

6.1. A Video

The results are also presented in a video [26] that shows the animated version of our four examples (Figure 8 (a)-(d)). Again, the first three examples employ parametrically defined deformable robots whereas the last example (d) employs an implicitly defined deformable robot. The pink curve on the right shows the computed continuous motion path connecting the start (in blue) and goal (in red) configurations employing Dijkstra's algorithm [25]. The deformable robot (in yellow) on the left undergoes the computed continuous contact motion that consists of 2- and 3-contact motions (and hence includes deformations) from the start configuration to the goal configuration.

The motion of our deformable robot is fully determined by Dijkstra's algorithm. The algorithm does not distinguish between 2- and 3-contact motions and it simply finds a sequence of edges connecting the start and goal configurations from the graph structure of 2/3 contact motions. By following the sequence of 2- and 3-contact motions, the robot undergoes either rigid transformations or deformations, respectively.

7. Conclusions and Future Work

In this work, we have extended previous results on motion planning for rigid robots to a generalized approach that sup-

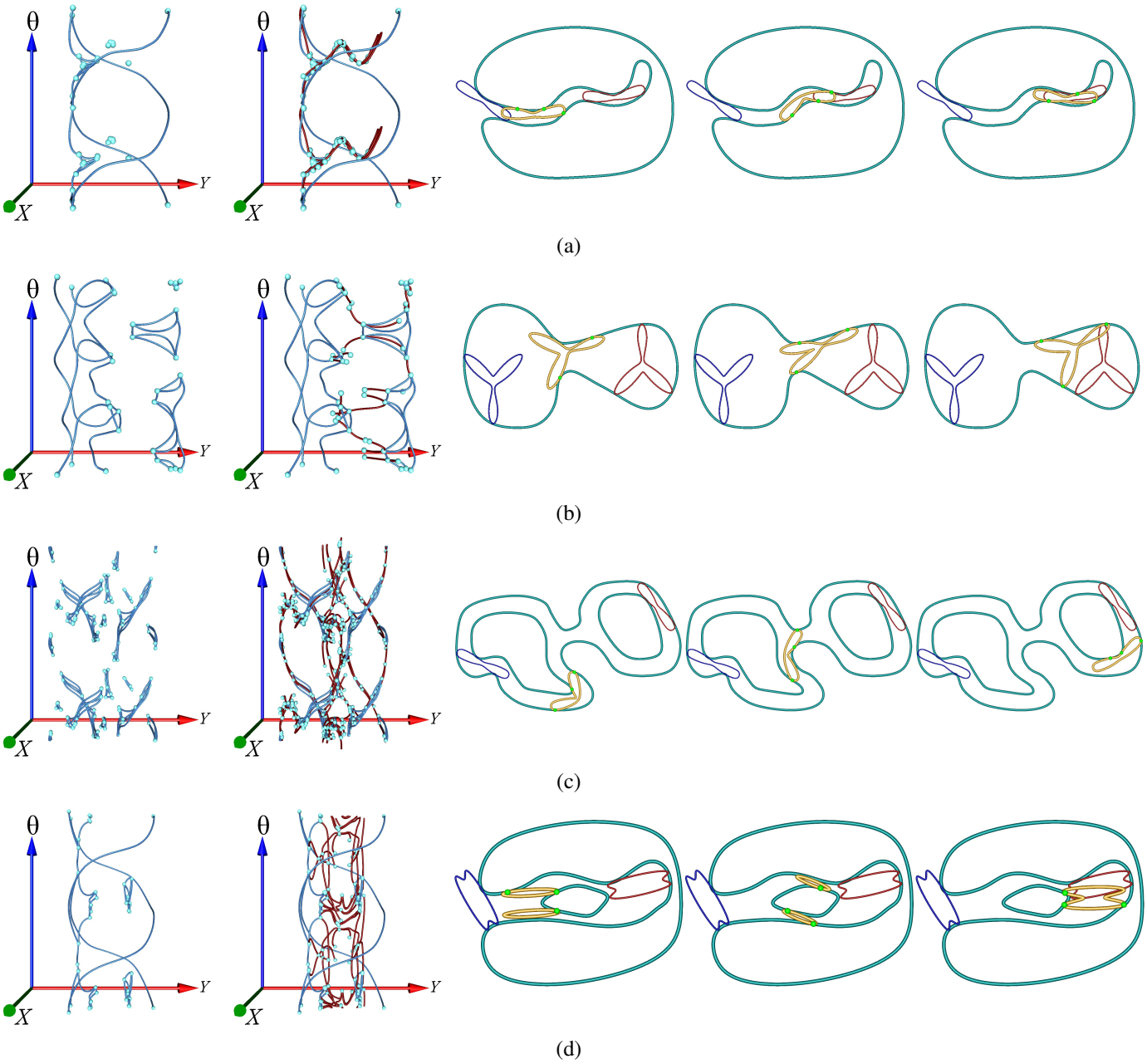


Figure 8: Continuous K-contact motion analyses of four examples. (a)-(c) are examples of a parametrically defined robot while (d) shows an example of an implicitly defined robot (See also Figure 2).

Example	2-contact	3-contact	4-contact	Total	Memory (MB)	Examined Domain (%) (in each dimension)
Fig. 8(a)	00:07:54	16:00:56	00:48:30	17:14:01	43.07	2.85
Fig. 8(b)	00:01:35	04:30:10	00:00:00	04:37:48	17.86	2.65
Fig. 8(c)	00:17:10	58:06:56	07:34:12	66:39:01	70.39	3.35
Fig. 8(d)	04:43:16	187:22:05	34:20:47	232:36:36	54.85	4.02

Table 1: Timing (in HH:MM:SS), memory consumption (in Megabytes) and average fraction of examined domain in each dimension (in %).

ports deformable robots. This extension is based on the representation of deformable robots using one-parameter family of deformations. Algebraic methods played a major role in this work, an approach that yields high accuracy in the order of machine precision. More importantly, based on the ability of the exploited multivariate solver [21, 22, 23] that ensures finding all real solutions, we can guarantee that the configuration space will be exhaustively analyzed, and thus always finds a feasible motion if one exists, up to the prescribed tolerance. While the presented algorithms are slow and requires hours and days of computations, they all used a single thread. Parallelizing the solver is possible and can clearly reduce the computing time. Another possible consideration for optimization is the optimal δ value to be used in the local tracing. One can also consider further optimizations via the use of different δ values for each variable, which potentially reduce the domain size of the system of equations.

Our algorithm effectively addresses a narrow passage problem which is typically the most challenging part of the motion planning. On the other hand, it cannot handle open regions where there exist no 2/3 contact motions. These open regions can be effectively handled by using other well-known algorithms such as a probabilistic road map (PRM) [2] or a rapidly exploring random tree (RRT) [27]. However, there are several technical issues that must be resolved before applying these sampling based algorithms to C^1 continuous implicit or parametric curves. For example, collision test for the straight line in the C-space connecting two collision free configurations is non-trivial and it is even more difficult to check the condition efficiently. Therefore, combining our work with other sampling based method requires a further in-depth research and analysis.

Clearly one can consider more than one degree of freedom in the deformation function of the robot. In fact, the tangible robot of [1] can have several independent degrees of freedom. Let $C_i(u)$, $(i = 1, \cdots, n)$, $u \in [0, 1]$, be n regular parametric curves that can be convex-blended using $(n-1)$ independent parameters. Differently stated, consider the multivariate $M(u, t_1, ..., t_{n-1})$. By setting the $(n-1)$ independent parameters, t_j, fixed in M, a univariate is prescribed. While the complexity of solvers [21, 22, 23] is exponential in the dimension, the algebraic structure of the problem is clear. One can hope that this algebraic configuration can be exploited, as in the local behaviors scheme exploited in this work, reducing the dimensionality and exploiting the deformations' full degrees of freedom only when they are really needed.

Similarly, one should also consider handling piecewise C^1-continuous shapes, handling curve-point contacts, where the point is a discontinuity point. While conceptually simpler than the curve-curve contacts, it does require the management of some book-keeping of all the different contact events' cases.

Acknowledgments

This work was supported in part by the People Programme (Marie Curie Actions) of the European Union's Seventh Framework Programme FP7/2007-2013/ under REA grant agreement PIAP-GA-2011-286426, in part by the ISRAEL SCIENCE FOUNDATION (grant No.278/13). and also in part by the Korean MCST and KOCCA in the CT R&D Program 2014 (No. R2014060001), and in part by NRF Research Grants (No. 2013R1A1A2010085).

References

[1] O. Salomon, A. Wolf, Inclined links hyper-redundant elephant trunk-like robot, J. Mechanisms Robotics 4 (4) (2012) 045001–045001–6.

[2] L. E. Kavraki, P. Svestka, J. C. Latombe, M. H. Overmars, Probabilistic roadmaps for path planning in high-dimensional configuration spaces, IEEE Transactions on Robotics and Automation 12 (4) (1996) 566–580.

[3] L. Guibas, C. Holleman, L. Kavraki, A probabilistic roadmap planner for flexible objects with a workspace medial-axis-based sampling approach, in: Proc. IROS, 1999, pp. 254–259.

[4] O. B. Bayazit, H. Lien, N. Amato, Probabilistic roadmap motion planning for deformable objects, in: Proc. IEEE Trans. on Robotics and Automation, 2002, pp. 2126–2133.

[5] R. Gayle, P. Segars, M. C. Lin, D. Manocha, Path planning for deformable robots in complex environments, in: In Robotics: Systems and Science, 2005, pp. 225–232.

[6] A. Mahoney, J. Bross, D. Lin, D. Johnson, Deformable robot motion planning in a reduced-dimension configuration space, in: Proc. IEEE Trans. on Robotics and Automation, 2010, pp. 5133–5138.

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

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

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

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

[11] C. Holleman, L. Kavraki, J. Warren, Planning paths for a flexible surface patch, in: Proc. IEEE Trans. on Robotics and Automation, 1998, pp. 21–26.

[12] F. Lamiraux, L. Kavraki, Path planning for elastic object under manipulation constraints, Int'l J of Robotics Research 20 (3) (2001) 151–156.

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

[14] V. Milenkovic, E. Sacks, S. Trac, Robust complete path planning in the plane, in: Proc. of the Tenth Workshop on the Algorithmic Foundations of Robotics, 2013, pp. 37–52.

[15] Y. J. Kim, G. Elber, M. S. Kim, Precise continuous contact motion for planar freeform geometric curves, Graphical Models 76 (5) (2014) 580–592.

[16] G. W. T. Sederberg, P. Gao, H. Mu, 2dd shape blending: An intrinsic solution to the vertex path problem, in: Proc. SIGGRAPH '93, 1993, pp. 15–18.

[17] S. Cohen, G. Elber, R. B. Yehuda, Matching of freeform curves, Computer Aided Design 29 (5) (1997) 369 – 378.

[18] J. Gomes, L. Darsa, B. Costa, L. Velho, Warping and Morphing of Graphical Objects, The Morgan Kaufmann, 1998.

[19] G. Turk, J. O'Brien, Shape transformation using variational implicit functions, in: Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, 1999, pp. 335–342.

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

[21] G. Elber, M. S. Kim, Geometric constraint solver using multivariate rational spline functions, in: Proc. of the 6th ACM Symp. on Solid Modeling and Applications, ACM, 2001, pp. 1–10.

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

[23] I. H. M. Barton, G. Elber, Topologically guaranteed univariate solutions of underconstrained polynomial systems via no-loop and single-component tests, Computer Aided Design 43 (8) (2011) 1035–1044.
[24] IRIT 11.0 Users Manual, `http://www.cs.technion.ac.il/~irit` (2013).
[25] E. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1) (1959) 269–271.
[26] The result video, `https://youtu.be/_Aj-J_KE6ec`.
[27] S. M. Lavalle, J. J. Kuffner, Rapidly-exploring random trees: Progress and prospects, in: Algorithmic and Computational Robotics: New Directions, 2000, pp. 293–308.