

Motion Planning for Disc-shaped Robots Pushing a Polygonal Object in the Plane

Attawith Sudsang, Fred Rothganger, and Jean Ponce

Abstract—This paper addresses the problem of using three disc-shaped robots to manipulate a polygonal object in the plane in the presence of obstacles. The proposed approach is based on the computation of maximal discs (dubbed *maximum independent capture discs*, or *MICaDs*) where the robots can move independently while preventing the object from escaping their grasp. It is shown that, in the absence of obstacles, it is always possible to bring a polygonal object from any configuration to any other one with robot motions constrained to lie in a set of overlapping MICaDs. This approach is generalized to the case where obstacles are present by decomposing the corresponding motion planning task into (1) the construction of a collision-free path for a modified form of the object, and (2) the execution of this path by a sequence of simultaneous and independent robot motions within overlapping MICaDs. The proposed algorithm is guaranteed to generate a valid plan provided a collision-free path exists for the modified form of the object. It has been implemented and experiments with Nomadic Scout mobile robots are presented.

Keywords—capture regions, object manipulation, pushing, motion planning

I. INTRODUCTION

This paper addresses the problem of using three disc-shaped robots to manipulate a polygonal object in the plane in the presence of obstacles. In practice, the robots may be the fingertips of a gripper or mobile platforms. The proposed approach is based on the computation of maximal discs (dubbed *maximum independent capture discs*, or *MICaDs* for short) of the workspace where the robots can move independently while guaranteeing that the object cannot escape their grasp.

We show that, in the absence of obstacles, there is a neighborhood of any object configuration such that any other configuration within it can be reached using robot motions confined to the associated MICaDs (a property akin to local controllability). This forms the basis for an approach to manipulation planning where an object's trajectory is divided into maximal segments such that the MICaDs associated with the endpoints of each segment overlap. We generalize this approach to the case where obstacles are present by decomposing the corresponding motion planning task into (1) the construction of a collision-free path for a modified form of the object, and (2) the execution of this path by a sequence of simultaneous and independent robot motions within overlapping MICaDs. The proposed algorithm is guaranteed to generate a valid plan provided a collision-free path exists for the modified form of the object. In addition, it does not assume that contact is maintained during the execution of the manipulation task, nor does it rely on detailed and a priori unverifiable models of friction or contact dynamics, but it allows the construction of manipulation plans guaranteed to succeed under the weaker assumption that jamming does not occur during the task execution. We have implemented the proposed approach and present experiments with Nomadic Scout mobile robots.

The proposed approach is based on a few lemmas. Proofs too lengthy to be included in the body of this presentation are

relegated to an appendix, along with the derivation of a few key equations. A preliminary version of this work appeared in [1].

A. Background

When a hand holds an object at rest, the forces and moments exerted by the fingers balance each other to achieve equilibrium. For the hand to hold the object securely, it should also be capable of preventing any motion due to external forces and torques. This is captured by the dual notions of *form and force closure* that constitute the traditional theoretical basis for grasp planning [2], [3], [4]. Recently, Rimon and Burdick have introduced the notion of *second-order immobility* [5], [6] and shown that certain equilibrium grasps of a part which do not achieve form closure effectively prevent any *finite* motion of this part through curvature effects in configuration space. We introduced in [7], [8] the more general notion of *inescapable configuration space* region (or *ICS*): as shown in [5], [6], an object is immobilized when it rests at an isolated point of its free configuration space; moving the fingers in an appropriate way transforms this isolated point into a compact region of free space (the ICS) that cannot be escaped by the object.

ICS regions were introduced in the context of in-hand manipulation with a multi-fingered reconfigurable gripper [7], [8] (see the works of Rimon and Blake [9] and Davidson and Blake [10] for the related notion of a *cage* in two- and three-finger grasping). We showed in [11] that ICS regions can also be used to manipulate an object by pushing it with three disc-shaped robots moving one at a time along straight lines with fixed directions. The object remains at all times in the ICS region associated with the robots, and the manipulation will succeed as long as the friction forces associated with contacts between the robots, the object and its supporting plane are not large enough to cause jamming.

The methods presented in [10], [9], [11] can only be used to generate plans where robots move one by one in a fixed direction. This severely limits the extent of each manipulation step, forcing any complex motion to be decomposed into a large number of atomic elements. We introduced in [12] another technique allowing the robots to move one by one in two-dimensional regions. Here we go further, showing how to construct maximum-radius workspace discs where the robots can move simultaneously and independently while preventing the object from escaping their grasp, and use these discs as the basis for a new approach to motion planning in the presence of obstacles.

The general form of the manipulation planning problem is discussed in [13], [14], and [15]. The proposed approach is related to a number of other techniques for manipulation planning where the uncertainty in the position of the manipulated part is reduced by exploiting the task mechanics. Work in this area

was pioneered by Inoue [16] and Whitney [17], and techniques for re-orienting parts through pushing and grasping operations have been developed by several authors, including Fearing [18], Mason [19], Mani and Wilson [20], Brost [21], Erdmann and Mason [22], Peshkin and Sanderson [23], Goldberg [24], Abell and Erdmann [25], Rao and Goldberg [26], Akella et al. [27], and Leveroni and Salisbury [28]. Unlike these methods, that all assume some predictive model of the contact mechanics, our approach does not require that robot/object contact be maintained during grasping or manipulation, nor does it rely on detailed models of friction and contact dynamics. In addition, by decomposing the construction of manipulation plans into the computation of a collision-free path for (a modified form of) the object, followed by the construction of elementary robot motion sequences for executing this path, we obtain an efficient approach to manipulation planning in the presence of obstacles that is capable of exploiting the effective algorithms that are available for motion planning in low-dimension configuration spaces, such as the exact motion planner of Avnaim *et al.* [29], or the approximate but very efficient planners of Barraquand and Latombe [30] and Kavraki *et al.* [31].

II. MAXIMUM INDEPENDENT CAPTURE DISCS

We characterize in this section the configurations of three disc-shaped robots that capture a polygonal object, i.e., prevent it from escaping to infinity. We will assume that all contacts between the object, the robots and the supporting plane are frictionless. Although this assumption is not verified by most physical systems, robots that capture an object without friction will still capture it when friction is present, and the motion planning techniques presented in latter parts of this article will be guaranteed to succeed as long as the friction forces are not large enough to cause jamming. We assume in this section that the object has been grown by the robots' radius, while the robots have been shrunk into their centers. (This transforms the boundary of the object into a generalized polygon with straight and circular arcs.) We consider the grown object and the point-like robots to be workspace objects. We will restrict our analysis to the study of the constraints imposed on the (point) robots by three arbitrarily chosen straight edges E_1 , E_2 and E_3 of the grown object.

Notation: Given some fixed coordinate system for the plane, we will denote by $\mathbf{q}_i = (x_i, y_i)$ ($i = 1, 2, 3$) the positions of the robots B_1 , B_2 and B_3 , and by $\mathbf{p} = (x, y, \theta)$ the configuration of the polygonal object B being manipulated. The edges E_i are associated with the robots B_i ($i = 1, 2, 3$), and we denote by $E_i(\theta)$ the set of configurations (x, y) for which the robot B_i touches the edge E_i when the polygon B is at orientation θ .

A given configuration of the robots defines the set \mathcal{F} of free configurations of the polygon in its configuration space $\mathbb{R}^2 \times S^1$. For polygonal objects, the robots immobilize the object at equilibrium, which occurs when the edge normals at the three contact points intersect at a single point and positively span the plane [5], [6].

We say that the robots *capture* the object when \mathcal{F} contains a compact connected component \mathcal{I} and \mathbf{p} belongs to \mathcal{I} . In this case the object is free to move inside the *inescapable configuration space* (ICS) region \mathcal{I} , but it cannot escape the robots' grasp.

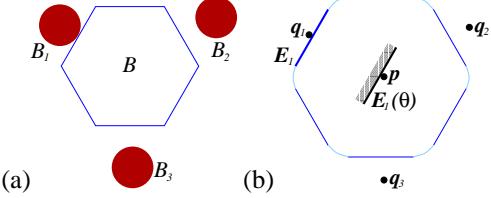


Fig. 1. Contact between an edge and a robot: (a) the original (ungrown) object and the three robots; (b) the grown object, the point-like robots, and the c-space obstacle induced by the contact. Both (a) and (b) are in workspace, and relevant configuration space objects (\mathbf{p} and $E_i(\theta)$) have been projected onto workspace.

It can be shown [11], [12] that the robots capture the polygon when the segments $E_i(\theta)$ ($i = 1, 2, 3$) enclose a triangle for all possible orientations of the polygon (we say that the *enclosure condition* is satisfied, Fig. 2(a)).

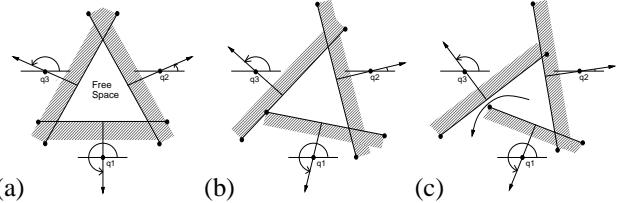


Fig. 2. Enclosure condition in the configuration space of the object: (a) three segments enclosing a triangle; (b) a critical configuration; (c) an opened triangle and an escape path. Each picture is a slice of c-space at a different (decreasing) value of θ . The robots are projected onto each slice to show how their associated ruled surfaces change as θ changes.

Starting from a configuration where the enclosure condition is satisfied, the robots will form an inescapable cage as long as this condition remains satisfied. When it is violated, there must exist some value of the orientation θ at which the segments $E_i(\theta)$ no longer enclose a triangle (Fig. 2(c)), allowing the polygon to escape through the opening. Because of the continuity of the motion of the segments $E_i(\theta)$, there must exist a critical orientation $\theta = \theta_c$ for which an endpoint of segment $E_i(\theta_c)$ lies on segment $E_j(\theta_c)$ right before the condition is violated (Fig. 2(b)). If (x_c, y_c) is the position of this endpoint along $E_j(\theta_c)$, we say that (x_c, y_c, θ_c) is a critical configuration of the polygon. We also say that a configuration of the robots is critical when there exists such a critical object configuration. We have just shown that:

Lemma 1: All paths joining a configuration of the robots that satisfies the enclosure condition to a configuration that violates this condition must contain a critical configuration.

Let $\mathbf{q} = (\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$ be a configuration of the robots for which the enclosure condition is satisfied. Consider the maximal (in the sense of set inclusion) connected subset of the configuration space of the robots that contains the configuration \mathbf{q} and is free of critical configurations. From Lemma 1, it is easy to see that all configurations in this region must satisfy the enclosure condition since there exists a path free of critical configurations joining \mathbf{q} to any configuration in the region. We will not attempt to characterize exactly the maximal region in the joint six-dimensional configuration space of the robots. Instead, we will give simple sufficient conditions for computing disc-shaped regions guaranteed to be free of critical configurations and thus satisfying the enclosure condition.

Definition 1: Three discs Ω_1, Ω_2 and Ω_3 are *independent capture discs* (or *ICaDs* for short) when, for any configuration \mathbf{q} in $\Omega_1 \times \Omega_2 \times \Omega_3$, the robots capture the object. A triple of ICaDs such that the radius of the smallest disc has maximal size is called a triple of *maximum independent capture discs*, or *MICaDs*.

A. Constructing the MICaDs

The rest of this section presents a method for constructing a triple of MICaDs. We assume that an immobilizing configuration \mathbf{q}_0 of the robots exists for the selected triple of edges¹ and characterize the range of motions that robots starting in \mathbf{q}_0 may undergo before reaching a critical configuration.

When the robots and the polygon are at a critical configuration, there exist i and j ($1 \leq i, j \leq 3$ and $i \neq j$) for which an endpoint of the segment $E_j(\theta)$ lies on the segment $E_i(\theta)$. This event occurs when the robot B_i touches the edge E_i and the robot B_j is located at the endpoint V_j of the edge E_j that is farthest from E_i . In such a configuration, the object will either touch two of the robots (Fig. 3(a)) or all three of them (Fig. 3(b)).

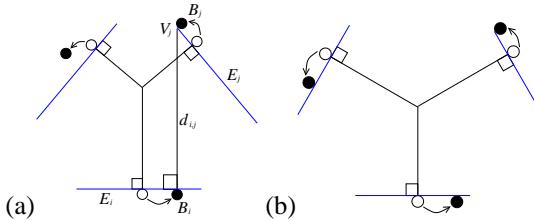


Fig. 3. The two possible types of critical configurations: (a) double contact; (b) triple contact. In each case, we move from an immobilizing configuration (white dots) to a critical one (black dots). The dots here represent the point robots.

The double-contact and triple-contact cases each provides a different set of constraints that form the basis of a method to calculate the positions and radii of the MICaDs. Both methods rely on exact solutions to three equations in three unknowns, and for the double-contact case the solution is optimal in the sense that the radii of the MICaDs are as large as possible for any placement of the discs.

The method we give for computing MICaDs in the triple-contact case is less than optimal. We sacrifice optimality in favor of ease of calculation. Its advantage is that a feasible result always exists, even when a feasible solution for the double-contact equations does not exist.

A.1 Double Contact

Definition 2: We denote by $d_{i,j}$ the distance from the edge E_i to the point V_j (of the edge E_j) that is farthest from E_i , and by $R_{i,j} = R_{j,i}$ the smallest of the $d_{i,j}$ and $d_{j,i}$ distances.

There is no critical configuration when the distance between B_i and B_j is shorter than $R_{i,j}$. More precisely, we have the following result.

Lemma 2: A sufficient condition for the enclosure condition to be satisfied at the configuration \mathbf{q} is that $|\mathbf{q}_i - \mathbf{q}_j| \leq R_{i,j}$ for $1 \leq i < j \leq 3$.

¹A triple of edge-robot pairs immobilizing the object always exists, and it can be found in time linear in the number of edges [32], [33].

This lemma assumes that when the robots are in configuration \mathbf{q} , they are not so close to each other that the space they enclose is empty. Since we start with an immobilizing configuration and then move the robots away from each other, this assumption holds. This lemma gives us a method for constructing a set of ICaDs by imposing appropriate constraints on the sizes of these discs (Fig. 4).

Definition 3: We denote by Ω_i the disc with center $\mathbf{u}_i = (u_i, v_i)$ and radius $r_i > 0$ that touches the edge E_i of the object ($i = 1, 2, 3$), and by $C_{i,j}$ the disc with center \mathbf{u}_i and radius $R_{i,j} - r_i$ for i and j such that $1 \leq i < j \leq 3$.

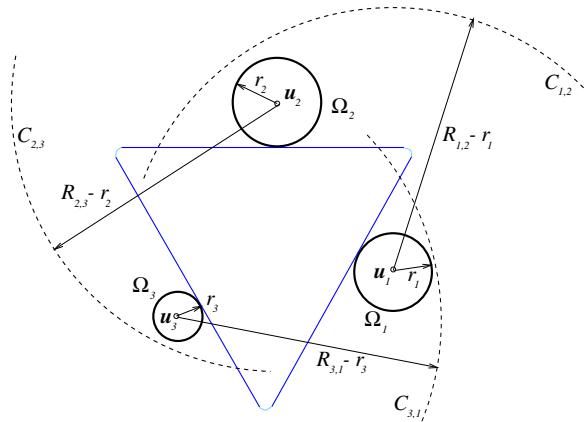


Fig. 4. Construction of independent capture discs.

The following result is an immediate corollary of Lemma 2.

Lemma 3: When the radii r_i ($i = 1, 2, 3$) are chosen so that $\Omega_1 \subset C_{3,1}$, $\Omega_2 \subset C_{1,2}$ and $\Omega_3 \subset C_{2,3}$, the discs Ω_1 , Ω_2 and Ω_3 form independent capture discs.

Positive radii satisfying the hypotheses of Lemma 3 may not exist, in which case the solution is not feasible and we must use the triple-contact method. Assuming that they do exist, and let us address the problem of maximizing the smallest of the three radii.

Lemma 4: The independent capture discs maximizing the smallest of their radii satisfy the following conditions: (a) Ω_1 is tangent to $C_{3,1}$, Ω_2 is tangent to $C_{1,2}$ and Ω_3 is tangent to $C_{2,3}$; (b) the robot configuration where the discs Ω_i , $i = 1, 2, 3$ touch the edge E_i must immobilize the object; and (c) the three discs must have the same radius.

Proof:

(a) Ω_1 is tangent to $C_{3,1}$, Ω_2 is tangent to $C_{1,2}$ and Ω_3 is tangent to $C_{2,3}$. It is easy to see from Figure 4 that when this condition is not satisfied, there is space for the smallest disc to grow. Figure 5(a) shows an example after the radius r_3 has been increased until this condition is satisfied.

(b) *The robot configuration where the discs Ω_i , $i = 1, 2, 3$ touch the edges E_i must immobilize the object.* Even when condition 1 is satisfied, it is still possible for the radii to increase. Figure 5(a) illustrates this: condition 1 is satisfied but the three normals at the contact between the independent discs and the object do not intersect. This implies that the object is not immobilized by the discs, therefore it can move to a configuration where it does not touch the discs (Fig. 5(b)), providing space for them to get larger.

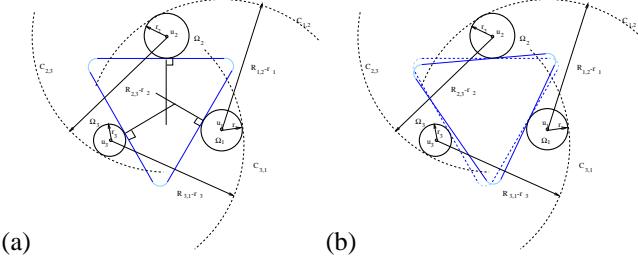


Fig. 5. Proving Lemma 4: (a) independent capture discs where condition 1 satisfied; (b) when the discs do not immobilize the object, there is some space for the smallest disc to grow.

(c) *The three radii are equal: $r_1 = r_2 = r_3$.* Consider Figure 6. We denote by $L_{i,j}$ the line segment joining two points in the disc Ω_i and the disc Ω_j that are farthest from each other. Clearly, $L_{i,j}$ passes through the centers of both discs. For condition 1 to be satisfied, the length of $L_{i,j}$ must be maintained constant at $R_{i,j}$. In Figure 6(a), we assume that the first two conditions are satisfied and r_2 is the smallest radius. To increase the radius r_2 , we change the intersection point between $L_{1,2}$ and $L_{2,3}$ (which is the location of the center of Ω_2). For the discs not to intersect with the object and to maintain the constant length of each $L_{i,j}$, as r_2 increases, the object must move toward Ω_1 and Ω_3 and the radii r_1 and r_3 must decrease (Fig. 6(b)). Since this process can always be applied as long as the radius of the smallest disc is strictly smaller than the other ones, this radius is maximized when all three radii are equal.

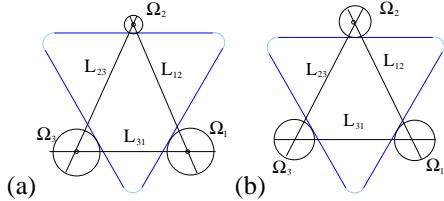


Fig. 6. Growing the smallest disc.

Lemma 4 allows us to compute the MICaDs. For a given configuration of the object, these discs are defined by three parameters: the position (s, t) of the intersection of the three normals to the object edges at the points where the discs touch them, and the common radius r of the discs. Lemma 4 yields a quadratic equation in r, s and t for each disc (Appendix A), and the maximum capture discs can be found by solving the corresponding system of three equations in three unknowns using homotopy continuation [34] and picking the solution that yields the maximum value of r . The centers of the corresponding discs are easily determined by the condition that they must be tangent to the selected object edges. \blacksquare

A.2 Triple Contact

For the double-contact solution to be feasible: 1) r, s and t must be real numbers (imaginary component equal to zero), 2) r must be positive, and 3) (s, t) must be positioned so that each disc is tangent to its respective edge at a point that is actually on the line segment that is part of the object. Failing that, we

cannot determine the MICaDs using the double-contact equations. (The reason why this occurs has to do with the shape of the configuration space obstacles, and is beyond the scope of this paper.) However, we can use the following method, which is based on constraints derived from the triple-contact case.

Let $\mathbf{q}_0 = (\mathbf{q}_{01}, \mathbf{q}_{02}, \mathbf{q}_{03})$ be an immobilizing configuration for the object. There must exist some $\delta > 0$ such that a critical configuration occurs when the distance between each pair of robots is increased by δ . This critical configuration is necessarily associated with three contacts (Fig. 3(b)) since there is no two-contact solution.

We calculate the maximum distance the robots can move away from each other before a critical configuration occurs. More precisely, if $d_{i,j} = |\mathbf{q}_{0i} - \mathbf{q}_{0j}|$ ($1 \leq i < j \leq 3$), we seek $\delta > 0$ such that

$$|\mathbf{q}_i - \mathbf{q}_j| = d_{i,j} + \delta, \quad \text{for } 1 \leq i < j \leq 3, \quad (1)$$

and, say, \mathbf{q}_1 coincides with the vertex V_1 of E_1 farthest from the edge E_2 while \mathbf{q}_2 and \mathbf{q}_3 lie on the edges E_2 and E_3 . This critical configuration is defined by the three equations (1) in three unknowns δ, t_2 and t_3 , where t_2 and t_3 define the positions of \mathbf{q}_2 and \mathbf{q}_3 along the corresponding edges. The solution yielding the minimum positive value of δ is picked. The process is repeated for each pair E_i, E_j ($1 \leq i, j \leq 3$ and $i \neq j$) and the solution corresponding to the minimum overall δ value is retained. The discs of radius $\delta/4$ touching the edges of the object at $\mathbf{q}_{01}, \mathbf{q}_{02}, \mathbf{q}_{03}$ are the desired MICaDs.

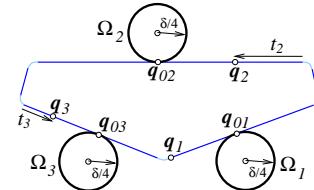


Fig. 7. Construction of MICaDs in three-contact case.

The existence of a physically-realizable solution is guaranteed by the fact that \mathbf{q}_0 is an immobilizing configuration and there must exist a critical configuration at some $\delta > 0$. The choice of $\delta/4$ for the radius of the discs guarantees that the distance between any pair of robots cannot be increased by more than δ .

III. MOTION PLANNING WITHOUT OBSTACLES

Without loss of generality, we will continue to assume in this section that the object has been grown by the radius of the robots and the robots have been shrunk into points. The MICaDs are rigidly attached to the boundary of this object, and we consider them as a single (if disconnected) rigid workspace object Ω , whose reference frame coincides with the frame attached to the object. There is a unique point on the boundary of each MICaD where the disc touches the edge of the grown object.

Notation: From now on, we will denote by Ω_i the MICaD associated with E_i ($i = 1, 2, 3$), by $\Omega_i(\mathbf{p})$ the region of the workspace occupied by Ω_i at configuration \mathbf{p} . We will denote by \mathbf{g}_i the point on the boundary of Ω_i where it touches the grown

object, and by $\mathbf{g}_i(\mathbf{p})$ the point in workspace occupied by \mathbf{g}_i at configuration \mathbf{p} .

The three robots form an inescapable cage if there exists a configuration \mathbf{p} of Ω such that \mathbf{q}_i belongs to $\Omega_i(\mathbf{p})$ for $i = 1, 2, 3$. According to Lemma 4, when the positions of the robots are $\mathbf{q}_i = \mathbf{g}_i(\mathbf{p})$ ($i = 1, 2, 3$), the object is immobilized in the corresponding configuration \mathbf{p} .

Lemma 5: Given some object configuration \mathbf{p} and assuming that the radius r of the capture discs is positive, there always exists a neighborhood U of \mathbf{p} for which $\Omega_i(\mathbf{p}) \cap \Omega_i(\mathbf{p}') \neq \emptyset$ for $i = 1, 2, 3$ and any configuration \mathbf{p}' in U .

Figure 8 illustrates this lemma, which follows directly from the fact that the mapping $\mathbf{u}_i : \mathbb{R}^2 \times S^1 \rightarrow \mathbb{R}^2$ that associates with a configuration \mathbf{p} of the object the center of the corresponding disc $\Omega_i(\mathbf{p})$ is continuous. Thus the pre-image of the open ball centered in $\mathbf{u}_i(\mathbf{p})$ with radius $2r$ is also an open set $U_i(\mathbf{p})$, not empty since it contains \mathbf{p} . The set $U = \cap_{i=1}^3 U_i(\mathbf{p})$ is also open and nonempty, and it defines a neighborhood of \mathbf{p} where $\Omega_i(\mathbf{p}) \cap \Omega_i(\mathbf{p}') \neq \emptyset$ for $i = 1, 2, 3$ and any \mathbf{p}' in U .

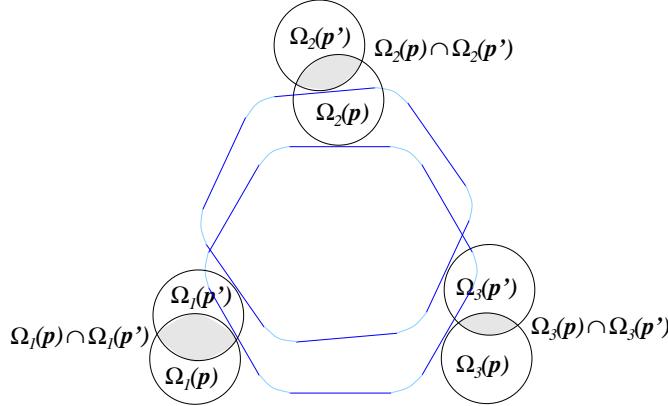


Fig. 8. An object in two different configurations \mathbf{p} and \mathbf{p}' , with overlapping MICaDs due to the fact that \mathbf{p}' is in the neighborhood U of \mathbf{p} .

Lemma 6: Given an object configuration \mathbf{p} and any other configuration \mathbf{p}' in the neighborhood U defined in Lemma 5, and some configuration \mathbf{q} in $\Omega_1(\mathbf{p}) \times \Omega_2(\mathbf{p}) \times \Omega_3(\mathbf{p})$, there exists a trajectory of the three robots starting in \mathbf{q} that will bring the object to the configuration \mathbf{p}' .

The proof of this corollary is constructive: if $\mathbf{q} = (\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$, move each robot from \mathbf{q}_i to some point in $\Omega_i(\mathbf{p}) \cap \Omega_i(\mathbf{p}')$ for $i = 1, 2, 3$, then move the robots again to the immobilizing configuration $\mathbf{q}' = \mathbf{g}(\mathbf{p}')$ in $\Omega_1(\mathbf{p}') \times \Omega_2(\mathbf{p}') \times \Omega_3(\mathbf{p}')$ (Fig. 9). Of course, one should make sure that the robots never leave their corresponding discs. Since the discs are convex, we can easily achieve this by limiting the motions of the robots to translations. Note that Lemma 6 states a property akin to *local controllability* in manipulation planning [35]. Since, in the absence of obstacles, the configuration space of the object is connected and the configuration \mathbf{p} can be arbitrarily chosen, we can iteratively apply this procedure to bring the object from any initial configuration to any final one.

Implicit in moving the object from \mathbf{p} to \mathbf{p}' is that one or more of the robots will make contact with and push the object. However, we do not need to explicitly plan any push. In essence, we are moving a bubble (the ICS) around in configuration space,

and the object must remain in that bubble. A wall of the bubble represents contact between a robot and an edge of the object. If the object's configuration lies on that wall as it moves, the robot is pushing the object.

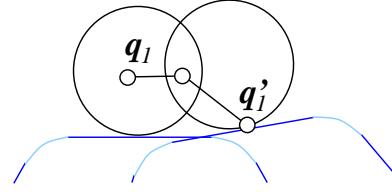


Fig. 9. Close-up of a robot moving from some point \mathbf{q}_1 in $\Omega_1(\mathbf{p})$ to some point in $\Omega_1(\mathbf{p}) \cap \Omega_1(\mathbf{p}')$ and finally to $\mathbf{q}'_1 = \mathbf{g}_1(\mathbf{p}')$.

A. Generating Paths for the Robots

Let us now address the problem of commanding the robots so that the manipulated object follows some pre-specified path \mathcal{P} parameterized by $\mathbf{p} : [0, 1] \rightarrow \mathbb{R}^2 \times S^1$. To reduce the possibility of jamming due to (unmodeled) friction, we will not attempt to immobilize the object except at its goal configuration. The three robots can be thought of as a moving cage along the path. To simplify the notation, we will use $\Omega_i(t)$ for $\Omega_i(\mathbf{p}(t))$ and $E_i(t)$ for the location of the edge E_i associated with the object configuration $\mathbf{p}(t)$. Suppose that the robots B_i are at the configurations \mathbf{q}_i in $\Omega_i(t)$ ($i = 1, 2, 3$) and thus capture the object. From Lemma 5, there always exists some $\delta > 0$ for which $\Omega_i(t) \cap \Omega_i(t + \delta)$ is not empty (we will see in a minute how to compute the maximum possible value for δ). To progress by δ along the path, we command each robot to move from \mathbf{q}_i to the point $\mathbf{q}_i^*(t, \delta)$ in $\Omega_i(t) \cap \Omega_i(t + \delta)$ that is closest to the line $E_i(t + \delta)$ (Fig. 10). This approach allows the robots to move independently during a single step, which greatly reduces the precision of control required to execute a motion plan.

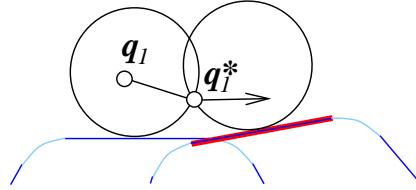


Fig. 10. Close-up of a robot moving from some point \mathbf{q}_1 in $\Omega_1(t)$ to the point \mathbf{q}_1^* closest to the line $E_1(t + \delta)$ (drawn here using a thick brush) in $\Omega_1(t) \cap \Omega_1(t + \delta)$.

Any point in $\Omega_i(t) \cap \Omega_i(t + \delta)$ could have been chosen for $\mathbf{q}_i^*(t, \delta)$. However, the proposed choice keeps each robot as close to the object as possible, which, as discussed in Section IV-A, will allow larger step sizes in the presence of obstacles. It is also easy to compute: Figure 11 shows the three possible cases. They involve four points: the two intersections of the circles, and the point on each circle that is closest to the line $E_i(t + \delta)$. We only consider those points that are contained in both discs, and choose the one that is closest to the line.

To move the object along the chosen path, we first command the robots B_i to move to $\Omega_i(0)$ for $i = 1, 2, 3$, so that they can capture the object at the initial configuration $\mathbf{p}(0)$. The step described above is then repeated until the robots B_i are in $\Omega_i(1)$

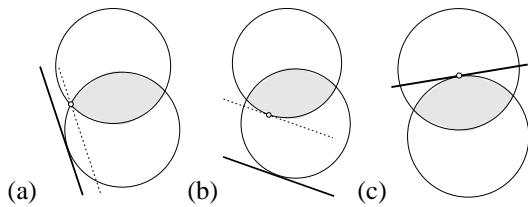


Fig. 11. The point in the intersection of two discs that is closest to a given line.

and the object can be immobilized and brought to the goal configuration $\mathbf{p}(1)$. To find an efficient plan, we must determine the maximum value δ^* of δ for each step in the sequence. This problem is addressed in the next section.

B. Computing Maximum Steps Along a Path

Let us fix some value t_0 for t . When $\delta = 0$, the two MI-CaDs $\Omega(t_0)$ and $\Omega(t_0 + \delta)$ obviously coincide. As δ increases, there must exist some critical value δ_i^* of δ for which the two discs $\Omega_i(t_0)$ and $\Omega_i(t_0 + \delta)$ ($i = 1, 2, 3$) are tangent to each other before their intersection becomes empty. Let δ^* denote the smallest of these three values; it is then clear that the MI-CaDs will overlap for $0 \leq \delta \leq \delta^*$ and that at least one of the corresponding pairs of discs will get disconnected when $\delta > \delta^*$.

Given an object configuration $\mathbf{p}_0 = \mathbf{p}(t_0)$, computing the maximum allowable step δ^* along the path \mathcal{P} is thus equivalent to computing the three values δ_i^* ($i = 1, 2, 3$), i.e., solving three instances of the following problem: Given a fixed disc A_0 with center \mathbf{a}_0 and radius r , find all object configurations \mathbf{p} such that a disc A with center \mathbf{a} and radius r rigidly attached to the object remains tangent to A_0 (Fig. 12). Here A_0 and A stand for a pair of maximum inescapable discs associated with the same edge of the object in configurations \mathbf{p}_0 and \mathbf{p} .

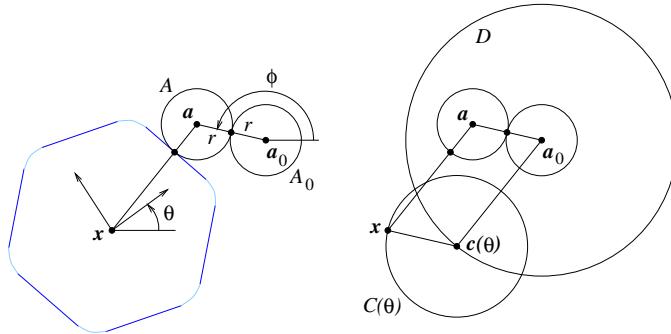


Fig. 12. Object configurations such that the associated MI-CaD is tangent to a fixed MI-CaD. Left: the two discs and the object. Right: the circle $C(\theta)$ swept by the position of the reference point x as ϕ varies while θ remains fixed, and the circle D swept by the center $c(\theta)$ of $C(\theta)$ as θ varies.

We parameterize this problem by two angles: the orientation ϕ of the line joining \mathbf{a}_0 and \mathbf{a} , and the orientation θ of the object at configuration \mathbf{p} . If \mathbf{x} denotes the position of the reference point of the object in this configuration, it is easy to see that, for a fixed value of θ , \mathbf{x} describes a circle $C(\theta)$ with center $\mathbf{c}(\theta)$ (Fig. 12), such that for any value of ϕ , we have $\mathbf{c}(\theta) - \mathbf{a}_0 = \mathbf{x} - \mathbf{a}$. As θ varies, the point $\mathbf{c}(\theta)$ describes a circle D centered in \mathbf{a}_0 with radius $|\mathbf{x} - \mathbf{a}|$.

In configuration space, this means that the configurations \mathbf{p}

such that A_0 and A remain tangent to each other form the helicoidal surface \mathcal{H} swept by the circles $C(\theta)$ as θ varies between 0 and 2π . This surface acts as a boundary that limits the largest single step that we can take along the path \mathcal{P} . Its shape is independent of the object and robot configurations, but its position in the configuration space of the object depends on \mathbf{p}_0 .

To determine δ_i^* ($i = 1, 2, 3$), we search for the first intersection between the path \mathcal{P} and the corresponding helicoid \mathcal{H}_i , starting in $t = t_0$. This is a simple one-dimensional numerical search that can be implemented efficiently. The value t_i^* of t corresponding to the first intersection determines $\delta_i^* = t_i^* - t_0$. The parameter δ^* is then determined as the smallest of the three values found.

IV. MOTION PLANNING WITH OBSTACLES

We can adapt our approach to the case where obstacles are present by adding the constraint that the object and the robots should never collide with an obstacle during any step of the path. To achieve this, we must first find a collision-free path \mathcal{P} for a modified form of the object (Fig. 13). We will then modify the strategy outlined in the previous section to follow that path in steps that are as large as possible yet are guaranteed to prevent any collision.

The modified form of the object is essentially the union of the object with the three robots (having their natural non-zero radius this time) located at the reference immobilizing positions. The robots must be taken into account by the motion planning algorithm because they must pass through free workspace along with the object. Furthermore, there must be enough room around the object for Lemma 6 to apply and for the robots to be able to push it in arbitrary directions.

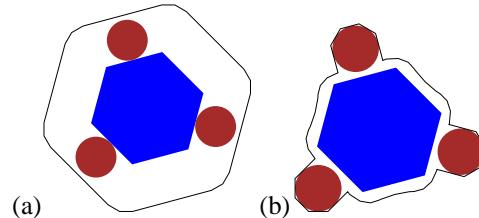


Fig. 13. Two different objects that may be passed to the motion planner: (a) the polygonal object is grown by $2r + \varepsilon$, where r is the robots' radius; (b) outer polygonal approximations of the robots are swept to a distance ε from the relevant edges, and the union of the swept region with the corresponding object envelope is computed.

Various forms can be chosen (Figure 13). Smaller ones will result in paths that tend to bring the object closer to obstacles, which forces a reduction in step size to avoid collisions. Larger ones may prevent the planner from finding feasible paths. We have conducted experiments with two approaches guaranteed to yield valid plans. The first one, discussed in [1], is to grow the object by the diameter of the robots plus some small clearance ε (Fig. 13(a)). However, this creates an unnecessarily large object, which keeps the planner from finding otherwise feasible paths. The second approach, which is currently used in our implementation, is to construct the polygonal region swept by (outer) polygonal approximations of the robots as they are moved to a distance of ε from the relevant edges and compute its union with the corresponding “envelope” of the object (Fig.

13(b)). This envelope is itself the union of all the regions of workspace that the object may occupy while it is captured by the robots in a given configuration. We will define envelopes more formally in Section IV-B.

We modify the strategy outlined in the previous section to follow the path output by the planner in steps that are as large as possible but at the same time small enough to avoid collisions. More precisely, we use Algorithm 1, which segments the path found by the planner and outputs a sequence of steps for each of the three robots to follow.

Algorithm 1

Input: The object, the obstacles, and the path of the object output by the motion planner.

Output: The path P . Each step of P is a configuration $\mathbf{q} = (\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$ of the three robots.

```

 $\mathbf{q} = \mathbf{g}(0); // \mathbf{g}(0) = (\mathbf{g}_1(\mathbf{p}(0)), \mathbf{g}_2(\mathbf{p}(0)), \mathbf{g}_3(\mathbf{p}(0)))$ 
 $P \leftarrow \{\mathbf{q}\};$ 
 $t \leftarrow 0;$ 
repeat
   $\delta^* \leftarrow \text{MaxDelta}(t);$ 
  Use bisection to find the maximum value  $\delta$  in  $[0, \delta^*]$  such
  that  $\text{CLEAR}(\mathbf{q}, t, \delta)$  returns TRUE;
   $\mathbf{q} \leftarrow \text{Closest}(t, \delta);$ 
   $P \leftarrow P \cup \{\mathbf{q}\};$ 
   $t \leftarrow t + \delta;$ 
until  $t = 1;$ 
 $P \leftarrow P \cup \{\mathbf{g}(1)\}.$ 
```

The function $\text{MaxDelta}(t)$ implements the calculation of the maximum step size in the absence of obstacles, as described in the previous section. The function $\text{Closest}(t, \delta)$ implements the method of selecting the point in the intersection of two MICaDs discussed in Section III-A and depicted in Figure 11. The predicate $\text{CLEAR}(\mathbf{q}, t, \delta)$ returns TRUE if it is not possible for the object to collide with the obstacles when the robots move from \mathbf{q} to $\text{Closest}(t, \delta)$ and FALSE otherwise. The rest of this section shows how to efficiently implement this predicate.

A. Implementing CLEAR

At each step in the plan, the three robots move simultaneously and independently from each other along line segments. Thus, implementing CLEAR requires the ability to test whether the object may collide with the obstacles for any position of each robot along the corresponding line segment. The following lemma simplifies the calculations.

Lemma 7: If $\text{ICS}(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$ denotes the set of free configurations of the object associated with the robot configurations \mathbf{q}_i ($i = 1, 2, 3$), then

$$\bigcup_{\substack{\mathbf{q}_i \in \overline{\mathbf{q}'_i \mathbf{q}''_i} \\ i = 1, 2, 3}} \text{ICS}(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3) = \bigcup_{\substack{\mathbf{q}_i \in \{\mathbf{q}'_i, \mathbf{q}''_i\} \\ i = 1, 2, 3}} \text{ICS}(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$$

for any two configurations $\mathbf{q}' = (\mathbf{q}'_1, \mathbf{q}'_2, \mathbf{q}'_3)$ and $\mathbf{q}'' = (\mathbf{q}''_1, \mathbf{q}''_2, \mathbf{q}''_3)$ in the same triple of MICaDs.

The proof of this lemma is elementary but lengthy, and it is relegated to Appendix B. Lemma 7 shows that despite the fact

that each robot can be anywhere on a line segment in the associated MICaD, testing whether the object can collide with the obstacles reduces to performing the collision test at the eight configurations where each robot is fixed at one of the ends of the associated line segment.

Given one of these configurations, we want to test for possible collisions of the captured object with an obstacle. Directly testing the intersection of the ICS with a configuration space obstacle is complicated and computationally expensive. Instead, we reformulate our problem in the workspace of the robots. If we define the *envelope* of an object as the region of the plane that it sweeps as its configuration varies within the ICS, testing for potential collisions reduces to testing whether the object envelope intersects these obstacles. Since we deal with polygonal obstacles, this only requires knowing how to test the intersection between the envelope and a line segment.

B. Collision Checking

To simplify the discussion, we will assume for the time being that our robots have zero radius, so there is no need for a distinction between the original object and the grown one. We will come back to the general case at the end of this section.

Rather than explicitly construct the envelope for each configuration of the robots, we directly test whether it is possible for any edge or vertex of the object to collide with a line segment bounding one of the obstacles. Suppose the object is in the ICS determined by a configuration of the three robots. Each robot is associated with one of the edges of the object, and the object can only touch the robots along those edges. The boundary of the envelope of an object corresponds to its configurations for which two of the edges are in contact with the corresponding robots, and it is therefore composed of three parts associated with the corresponding double contacts (Fig. 14). When a line segment intersects the envelope, it must intersect at least one of these parts, or lie inside the envelope.

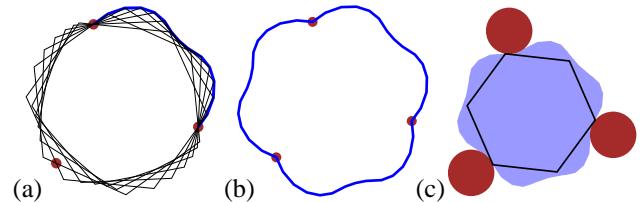


Fig. 14. The envelope of a hexagon: (a) the first of the three arcs formed by the first double contact between the hexagon and point robots; (b) the complete envelope; (c) the true envelope when the non-zero radius of the robots is taken into account as explained at the end of this section (the hexagon is shown at one of its extreme orientations in this case). Note that the envelope's boundary is composed of smooth curved segments.

Let us first focus on testing the intersection of a line segment with the part of the envelope boundary associated with the *infinite* line supporting one of the object edges. More concretely, let us consider the object configurations for which the robots B_1 and B_2 touch the edges E_1 and E_2 . Let L_a be the line supporting some other edge E_a of the polygon. We have the following lemma, which is proved in Appendix C.

Lemma 8: As the object rotates while maintaining contact with B_1 and B_2 , L_a rotates as well and remains tangent to a fixed circle C_a .

The object configurations that maintain contact with the robots B_1 and B_2 have a single degree of freedom and they can be parameterized by the object orientation θ . But since the object cannot escape, θ must be in some range $[\theta_L, \theta_H]$ where θ_L and θ_H correspond to two configurations for which the object touches the three edges. Figure 15(a) shows the region S_a swept by L_a as the object rotates in the range $[\theta_L, \theta_H]$.

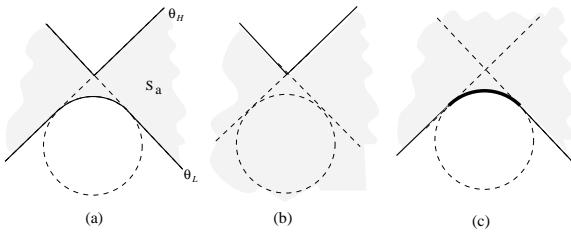


Fig. 15. Regions swept by (a) the line L_a , (b)-(c) the half spaces bounded by L_a .

Let us pretend for a moment that L_a is the only edge of the object. In this case, the object envelope is the region swept by one of the half-planes bounded by L_a as this line rotates while remaining tangent to the circle C_a . There are two possibilities, illustrated by Fig. 15(b)-(c): the envelope's boundary is either composed of two line segments (Fig. 15(b)), or of two line segments and a circular arc (Fig. 15(c)). In the first case, testing the intersection of a line segment with the boundary reduces to testing its intersection with the line L_a at orientations θ_L and θ_H . In the second case, an additional intersection test with the circular part of the envelope boundary must also be performed.

Of course, the edges bounding the object only have a finite extent. To take this into account, we now consider the edge E_a itself instead of its supporting line L_a . The region swept by this edge as the object rotates under the double-contact constraint is a subset of the region S_a swept by the line L_a . It is bounded by quartic curves traced by the endpoints of the edge E_a as it rotates around the circle C_a . Computing the intersection of the curve traced by an object vertex and a given line segment reduces to solving a trigonometric equation in the object orientation θ (which can be done in closed form), then testing whether this orientation is in the range $[\theta_L, \theta_H]$, and finally checking whether the corresponding intersection point is within the extent of the line segment (see Appendix D for details).

To summarize, we test the intersection of the envelope's boundary with a given line segment by testing, for each pair of contacts with two robots, the intersection of this segment with (1) the edges of the object at orientations θ_L and θ_H , (2) the circular arcs of the envelope boundary, and (3) the curves swept by the object vertices (Fig. 16). This process takes time linear in the size of the polygon, and it must be repeated for every obstacle edge (or until some intersection is found). It is of course also possible for an obstacle to be entirely contained in the envelope. This can be tested by checking whether the line segment either lies inside (1) the object when $\theta = \theta_L$ or $\theta = \theta_H$, or (2) the region swept by the edges as they rotate from θ_L to θ_H . The second test is similar to the intersection test between a segment and the curve swept by an object vertex, and it also reduces to solving a trigonometric equation. The envelope may also lie inside the obstacle, which can be tested by checking whether the

three robots themselves are inside the obstacle.

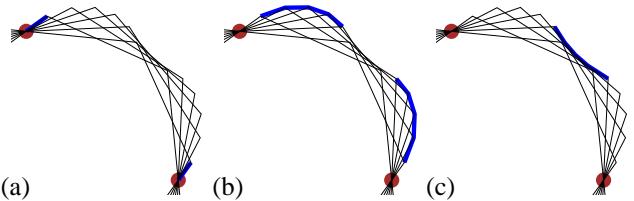


Fig. 16. The various types of boundary arcs of an envelope: (a) edges of the object at the extreme orientations; (b) curves traced by vertices of the object; (c) a circular arc traced out by an edge tangent to it.

As noted earlier, we have assumed in this section that the three robots have zero radius. It is easily shown that collision checking for robots with non-zero radius can be carried out exactly as described above, except that (1) the angles θ_L and θ_H are computed using the grown object and the shrunk robots, (2) the straight and circular parts of the envelope's boundary are of course constructed using the edges of the *original* object, not of its grown version, and (3) the intersections between the curves swept by object vertices and the obstacle edges are found (again in closed form) by solving a modified version of the trigonometric equation derived in the zero-radius case (see Appendix D for details).

C. Optimization

We can improve the efficiency of the approach proposed in the previous section by limiting collision checking to obstacle edges in the vicinity of the envelope: we compute a bounding rectangle for the envelope and test the intersection between this rectangle and each obstacle edge. To construct a bounding rectangle for the envelope, we first compute the radius of the smallest circle that contains the object (Fig. 17(a)): this is known as the *smallest enclosing circle* problem in computational geometry, and the radius can be found in time linear in the size of the polygon by a randomized algorithm [36]. We place three circles with this radius in the workspace so that each one touches a pair of the robots and its center is outside the triangle formed by the three robots (Fig. 17(b)). The bounding rectangle is computed as the smallest *isothetic* rectangle (i.e., with sides parallel to the coordinate axes) that contains the three circles. It is easy to show that the envelope is contained in this rectangle.

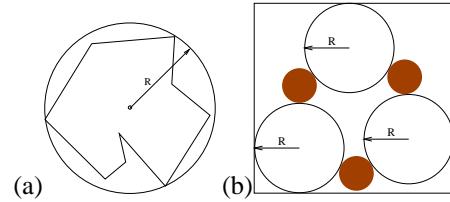


Fig. 17. Constructing a box bounding the envelope: (a) the smallest circle containing the original object, and (b) a simple bounding box of the envelope.

We use a quadtree to avoid testing edges that are far away from the rectangle bounding the object. The obstacle edges intersecting a quadtree leaf are stored along with this leaf, which allows the retrieval of the edges intersecting the quadrant of the workspace associated with any leaf in logarithmic time. An alternative would be to enclose every obstacle edge into an iso-

thetic rectangle, and retrieve all rectangles intersecting the rectangle bounding the object. In computational geometry, this problem is referred to as *orthogonal intersection searching* and it can be solved with query time $O(m + \log n)$ and preprocessing time $O(n \log n)$, where m is the number of edge-enclosing rectangles intersecting the object-enclosing rectangle, and n is the number of obstacle edges [37], [38].

V. IMPLEMENTATION AND RESULTS

We have implemented the construction of MICaDs for polygonal objects. The implementation does not include the triple-contact case (discussed in Section II-A.2), but is still capable of handling most polygons in which we are interested. Figure 18 shows some examples of polygons and their associated MICaDs.

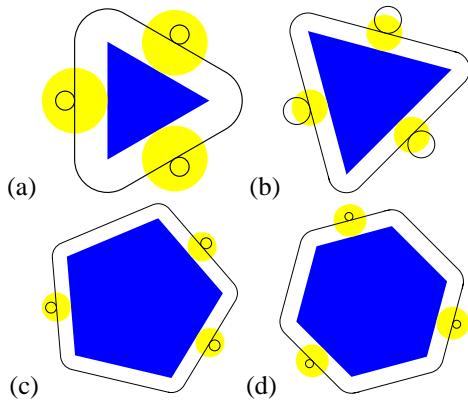


Fig. 18. Examples of MICaDs: (a) the object used in the experiments, (b)-(d) other examples. In each example, the polygonal object and the three robots are portrayed with shaded regions, while the grown generalized polygon and the corresponding MICaDs are portrayed with black lines.

We have also implemented the proposed motion planning algorithm, and conducted experiments using three Nomadic Scout robots and a styrofoam triangle on casters (Fig. 19). We use the motion planning library contained in the Move2D package from LAAS to compute collision-free paths for our objects. Move2D is a graphical motion planning package which contains a potential-field planner based on the navigation function method discussed in [39].² For obstacles we have used naturally occurring objects in our building, such as chairs and pillars, modeled as simple polygons.

Figure 20 shows still images of two experiments and the corresponding motion plans. Animations of these plans and actual video footage are available from our web site, http://www-cvr.ai.uiuc.edu/ponce_grp/demos/micad.

A. Closed-Loop Control

As mentioned above, we have conducted experiments with both open- and closed-loop control, using dead reckoning in the first case, and visual feedback from overhead cameras in the second one. The open-loop approach has proven insufficient for paths whose length is greater than about 5 meters, due to imperfect odometry and drift due to friction and slipping. For closed-loop control, we have implemented a visual localization system

²See the web site [40] for details on the software and its implementation.



Fig. 19. The robots and the polygonal object used in the experiments.

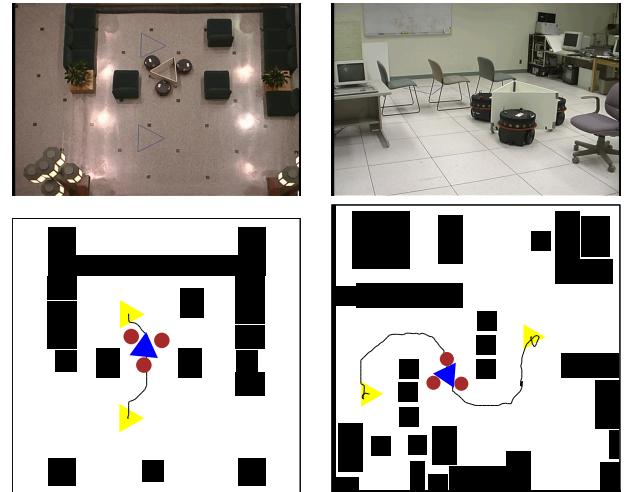


Fig. 20. Snapshots of two experiments using (left) dead reckoning and (right) closed-loop control.

based on the *Locext* software from LAAS [41]. Our localization system uses three overhead cameras (Fig. 21(a)), fitted with IR pass filters, and the robots are fitted with circular patterns of IR LEDs (Fig. 21(b)-(c)). Locext produces very accurate position information (within 5mm and 0.02 degrees) by finding the centroid of the image of a given LED to sub-pixel accuracy, and combining the information from all the visible points to localize a given robot. The use of multiple LEDs on each robot also serves to distinguish that robot from the others.

Each robot runs a server program that is responsible for both executing motion commands and determining the current location of that robot. Robots determine their position by combining two unreliable sources of information: odometry and visual localization. The accuracy of odometry degrades in proportion to the distance the robots travel. The accuracy of the visual localization system is quite good, and it is independent of how far the robots travel. However, when one robot is partially hidden from the cameras by one of the obstacles or the object, the visual localization system may get confused and return an incorrect value. Such errors are transitory, but the corresponding readings should of course never be used to guide a motion. This has prompted us to use a very simple model of uncertainty: As it moves, each robot maintains a moving disc D_i ($i = 1, 2, 3$) whose center represents its nominal position and whose radius

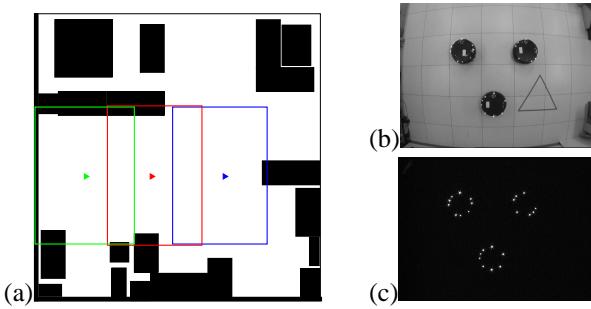


Fig. 21. Robot localization: (a) the fields of view of the three overhead cameras, overlaid on the floor plan of our laboratory, minus the chairs used as obstacles in Figure 20; (b) an image of the three robots with the IR LEDs turned on; and (c) the corresponding IR image.

is an upper bound on the current localization error. After each motion step, the position of D_i is updated using the robot's odometry, and its radius is grown using bounds on the odometry error. As long as this radius is below some preset threshold, the motion proceeds using odometry alone. When the threshold is exceeded, a position and orientation reading (x_i, y_i, θ_i) is requested from the visual localization system. If (x_i, y_i) falls within D_i , this reading is accepted, the center of D_i is set to (x_i, y_i) , and its radius is reset to zero. Otherwise, the reading is ignored, and one more step is taken using odometry before requesting a new reading. As mentioned earlier, the errors of the visual localization system are normally transitory, and in practice this simple approach works quite well, allowing the robots to move freely without being limited by the speed or coverage of the visual localization system, while maintaining accurate position information during extended motion sequences.

B. Computational Cost

We can measure the size and difficulty of a problem indirectly by observing the path length and number of steps output from the planner. We have generated plans with over 400 steps in fairly complicated simulated environments. The plan depicted in the right portion of Figure 20 has 129 steps. We computed the plan on a 500MHz Celeron PC system running Linux, and the time cost for various phases are: generate MICaDs = 2.02s; generate modified object = 293.44s; run Move2D motion planner = 2.71s; run Algorithm 1 = 23.31s.

Generating the modified object takes up an inordinate amount of time. The focus of our implementation efforts has been on the efficient construction of MICaDs and motion plans, so no particular care has been given to this part of the process. We have implemented the construction of the modified object using the LEDA package [42], specifically the rational geometry kernel. Because the size of the representation increases with each rational operation, this particular step effectively requires exponential space. It could be solved in much less time with a floating point implementation. LEDA provides such an implementation, but our objects are too complex for it to handle.

We can use the ratio of path length to serial step count as a measure of the efficiency of this planning method. Our previous work [43] produced a plan with an efficiency of $\frac{5\text{meters}}{85\text{steps}} = 0.0588$. The plan depicted in the right portion of Figure 20 has an efficiency of $\frac{13.0806\text{meters}}{129\text{steps}} = 0.1014$, or about twice as ef-

ficient as our previous approach. There exists work [44] where three mobile robots cage an object and move continuously, for an efficiency of $\frac{\text{path length}}{1}$. The difference is that [44] works on-line as a control law, while the method presented here works off-line, does explicit motion planning, and determines a whole class of motions that can achieve the same plan.

VI. DISCUSSION AND CONCLUSIONS

We have introduced in this paper the concept of maximum independent capture discs (MICaDs), where three robots can move independently while guaranteeing that an object cannot escape their grasp. We have given a simple method for constructing the MICaDs associated with a polygonal object, and shown that they can be used as a basis for motion planning in the presence of obstacles. We have presented an implementation of the proposed approach, along with experimental results.

Friction may in principle cause the robots to jam during the execution of a motion plan. We have tried to minimize the risk of jamming by avoiding a complete grasp of the object until the end of the manipulation task: indeed, grasping requires establishing three simultaneous contacts, with a much higher chance for jamming than when only one or two contacts occur at a given time [43]. Nonetheless, we feel that friction should be dealt with explicitly, and plan to address the more difficult problem of constructing jam-free plans in the near future.

The approach proposed in this article only considers contacts between the robots and three fixed edges of the object. This limits the size of the corresponding MICaDs. We are currently working on an extension that will allow multiple edges to be considered when determining the capture region of each robot.

Acknowledgments

This research was supported in part by the National Science Foundation under grants IRI-9634393 and IRI-9907009, and by an equipment grant from the Beckman Institute for Advanced Science and Technology.

We wish to thank Sara Fleury, Michel Devy, and Thierry Siméon from LAAS for kindly providing the localization and motion planning software. Thanks to Jeff Erickson, Shripad Thite, Peter Leven, Svetlana Lazebnik, and Sean Sitter for helpful discussions and comments.

APPENDICES

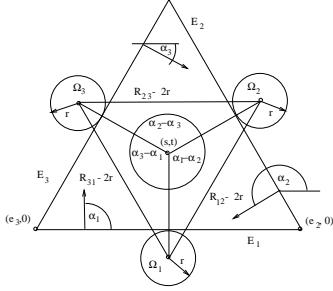
I. DOUBLE-CONTACT EQUATIONS

Let us denote by r the common radius of the three discs and by (s, t) the intersection of the three normals to the object edges (Fig. 22). Without loss of generality, assume that edge E_1 is on the x -axis. Let $(e_3, 0)$ be the intersection of the line supporting E_3 and the x -axis, and let $(e_2, 0)$ be the intersection of the line supporting E_2 and the x -axis. Denote by $\alpha_i, i = 1, 2, 3$ the orientations of the normals relative to the x -axis.

We write that the distance between the centers of Ω_1 and Ω_2 is $R_{1,2} - 2r$ and use the law of cosines to obtain

$$(R_{1,2} - 2r)^2 = (t + r)^2 + ((s - e_2) \cos \alpha_2 + t \sin \alpha_2 + r)^2 - 2(t + r)((s - e_2) \cos \alpha_2 + t \sin \alpha_2 + r) \cos(\alpha_2 - \alpha_1).$$

The remaining two cases yield two similar equations. The MICaDs can be computed by solving these three equations in the unknowns r, s and t using homotopy continuation [34].

Fig. 22. Computing r, s and t .

II. PROOF OF LEMMA 7

The set $\text{ICS}(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$ is a volume bounded by three ruled surfaces, each of which corresponds to the set of configurations for which the robot B_i at \mathbf{q}_i is in contact with edge E_i of the object.

When B_i is at the point \mathbf{q}'_i , the corresponding ruled surface is:

$$V'_i = (x - q'_i) \cos(\theta + \alpha_i) + (y - r'_i) \sin(\theta + \alpha_i) + d_i \geq 0.$$

Likewise, when B_i is at the point \mathbf{q}''_i , the corresponding ruled surface is:

$$V''_i = (x - q''_i) \cos(\theta + \alpha_i) + (y - r''_i) \sin(\theta + \alpha_i) + d_i \geq 0.$$

For $\eta_i + \mu_i = 1$, $\eta_i, \mu_i \geq 0$, we have $V(\eta_i, \mu_i) = \eta_i V'_i + \mu_i V''_i = 0$.

The equation $V(\eta_i, \mu_i) = 0$ defines the ruled surface associated with having robot B_i at $\eta_i \mathbf{q}'_i + \mu_i \mathbf{q}''_i$ (which is a point on the line segment $\overline{\mathbf{q}'_i \mathbf{q}''_i}$). Clearly, this ruled surface is always between the ruled surfaces $V'_i = 0$ and $V''_i = 0$. Therefore, we can write $\mathcal{V}_i(\eta_i, \mu_i) \subset \mathcal{V}'_i \cup \mathcal{V}''_i$, where we denote by $\mathcal{V}_i(\eta_i, \mu_i)$ the volume $V_i(\eta_i, \mu_i) \geq 0$, by \mathcal{V}'_i the volume $V'_i \geq 0$, and by \mathcal{V}''_i the volume $V''_i \geq 0$.

Using the above statement and considering $i = 1, 2, 3$, we can write:

$$\begin{aligned} \mathcal{V}_1(\eta_1, \mu_1) \cap \mathcal{V}_2(\eta_2, \mu_2) \cap \mathcal{V}_3(\eta_3, \mu_3) \\ \subset (\mathcal{V}'_1 \cup \mathcal{V}''_1) \cap (\mathcal{V}'_2 \cup \mathcal{V}''_2) \cap (\mathcal{V}'_3 \cup \mathcal{V}''_3). \end{aligned}$$

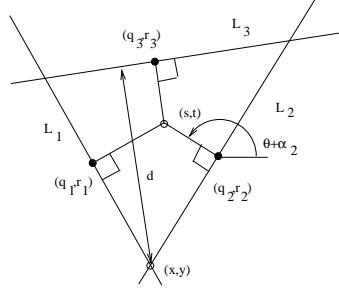
The left hand side of the statement above defines $\text{ICS}(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$ where \mathbf{q}_i can be anywhere on the line segment $\overline{\mathbf{q}'_i \mathbf{q}''_i}$ (depending on η_i and μ_i). That the above statement is true independent of the values of η_i and μ_i implies:

$$\begin{aligned} & \left(\bigcup_{\substack{\eta_i + \mu_i = 1 \\ i=1,2,3}} \mathcal{V}_i(\eta_i, \mu_i) \cup \mathcal{V}'_i \cup \mathcal{V}''_i \right) \\ & \subset (\mathcal{V}'_1 \cup \mathcal{V}''_1) \cap (\mathcal{V}'_2 \cup \mathcal{V}''_2) \cap (\mathcal{V}'_3 \cup \mathcal{V}''_3). \end{aligned} \quad (2)$$

The right-hand side of the statement above can be expanded as the union of eight sets. Each set is the ICS for the configuration having the robot B_i located at either end of the line segment $\overline{\mathbf{q}'_i \mathbf{q}''_i}$. For example, the set $\mathcal{V}'_1 \cap \mathcal{V}'_2 \cap \mathcal{V}'_3$ is $\text{ICS}(\mathbf{q}_1 = \mathbf{q}'_1, \mathbf{q}_2 = \mathbf{q}'_2, \mathbf{q}_3 = \mathbf{q}'_3)$.

Because the term \mathcal{V}'_i can be written as $\mathcal{V}_i(\eta = 1, \mu = 0)$ and the term \mathcal{V}''_i can be written as $\mathcal{V}_i(\eta = 0, \mu = 1)$, the reverse of the inclusion in statement (2) holds as well.

III. PROOF OF LEMMA 8

Fig. 23. The line L_3 as the object rotates.

Consider Figure 23. The three lines L_1 , L_2 and L_3 support edges E_1 , E_2 and E_3 that belong to the object; the points $\mathbf{q}_1 = (q_1, r_1)$ and $\mathbf{q}_2 = (q_2, r_2)$ are fixed, and $\mathbf{q}_3 = (q_3, r_3)$ is a non-fixed point on L_3 . The normals at \mathbf{q}_1 , \mathbf{q}_2 and \mathbf{q}_3 intersect at (s, t) . We want to show that when the object rotates while maintaining simultaneous contacts between \mathbf{q}_1 and L_1 and between \mathbf{q}_2 and L_2 , the point \mathbf{q}_3 will trace a circle C and the line L_3 is always tangent to this circle.

We give only a brief sketch of the proof here. We denote by θ the orientation of the object and by \mathbf{n}_1 , \mathbf{n}_2 and \mathbf{n}_3 the internal normals of the three edges. We start by listing six equations. The first three equations define the lines L_1 , L_2 and L_3 : $\mathbf{n}_1 \cdot (\mathbf{q}_1 - (x, y)^T) = 0$; $\mathbf{n}_2 \cdot (\mathbf{q}_2 - (x, y)^T) = 0$; $\mathbf{n}_3 \cdot (\mathbf{q}_3 - (x, y)^T) = d$. The other three equations are of similar form, but express the fact that the three internal normals at the contacts intersect at the point (s, t) (immobilizing the object).

We apply some algebraic manipulation to eliminate x , y , s , t and θ from the six equations, and obtain an equation of the form:

$$(K_1 q_3 + K_2)^2 + (K_1 r_3 + K_3)^2 = K_1^2 d^2,$$

where the K_i summarize constant terms. This shows that \mathbf{q}_3 traces out a circle of radius d .

To show that the line L_3 is always tangent to the circle for a given value of θ , we perform further algebraic manipulation to determine the center \mathbf{c} of the circle and to derive an equation for \mathbf{q}_3 as a function of θ . We also express the unit tangent \mathbf{t}_3 of the line L_3 as a function of θ . It then follows that $(\mathbf{q}_3 - \mathbf{c}) \cdot \mathbf{t}_3 = 0$.

IV. INTERSECTIONS BETWEEN THE CURVE TRACED BY A VERTEX AND A LINE SEGMENT

Let us consider the line defined by:

$$u \cos \alpha + v \sin \alpha + d = 0, \quad (3)$$

where α and d are some constants, and define the curve $U = \{(u(\theta), v(\theta))^T, \theta_L \leq \theta \leq \theta_H\}$ where $(u(\theta), v(\theta))^T$ is an endpoint of the edge E_a when the object is at orientation θ (Figure 24).

Writing that the distances from $(u(\theta), v(\theta))^T$ to the lines L_1 and L_2 (the lines supporting the edges E_1 and E_2) are the constants d_1 and d_2 yields:

$$\begin{aligned} (u(\theta) - q_1) \cos(\theta + \alpha_1) + (v(\theta) - r_1) \sin(\theta + \alpha_1) &= d_1, \\ (u(\theta) - q_2) \cos(\theta + \alpha_2) + (v(\theta) - r_2) \sin(\theta + \alpha_2) &= d_2. \end{aligned} \quad (4)$$

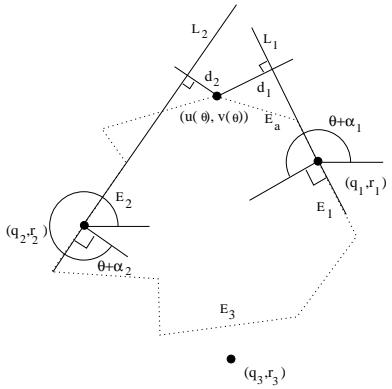


Fig. 24. Parameterizing the endpoint $(u(\theta), v(\theta))^T$. The object is drawn as dotted polygon here.

We want to compute the intersection of the curve U and the line L . Suppose the intersection occurs at $(u, v)^T$. We can then set $u(\theta) = u$ and $v(\theta) = v$ in (4), and use the resulting equations to eliminate u and v in (3). We obtain after some simplification:

$$(r_1 - r_2) \cos(\alpha - \alpha_1 - \alpha_2 - 2\theta) + (q_1 - q_2) \sin(\alpha - \alpha_1 - \alpha_2 - 2\theta) - 2d_2 \sin(\alpha - \alpha_1 - \theta) + 2d_1 \sin(\alpha - \alpha_2 - \theta) + K = 0, \quad (5)$$

where

$$K = \begin{cases} r_2 \cos(\alpha - \alpha_1 + \alpha_2) - r_1 \cos(\alpha + \alpha_1 - \alpha_2) \\ + 2d \sin(\alpha_1 - \alpha_2) + q_1 \sin(\alpha + \alpha_1 - \alpha_2) \\ - q_2 \sin(\alpha - \alpha_1 + \alpha_2). \end{cases}$$

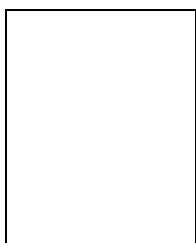
Finding the roots of Eq. (5) amounts to solving a trigonometric equation in θ which can be done in closed form. The intersection of U and L are the points $(u, v)^T$ at the root values of θ that are within the range $[\theta_L, \theta_H]$. To compute the intersection of a line segment supported by the given line L , an extra step is needed for testing whether any of the obtained intersection points is between the endpoints of the line segment.

Note that the derivation of (5) assumes that the robots have zero radius. To handle non-zero radii, we simply use the lines L_1 and L_2 supporting the edges of the *grown* object in the derivation, and use for d_1 and d_2 the distances between the vertex of interest of the *original*, ungrown object and these lines. The rest of the derivation remains unchanged.

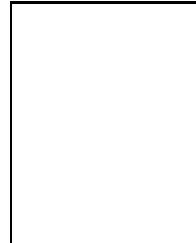
REFERENCES

- [1] A. Sudsang and J. Ponce, "A new approach to motion planning for disc-shaped robots manipulating a polygonal object in the plane," in *IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, 2000, pp. 1068–1075.
- [2] R.S. Ball, *A treatise on the theory of screws*, Cambridge University Press, 1900.
- [3] M.S. Ohwovorile, *An extension of screw theory and its application to the automation of industrial assemblies*, Ph.D. thesis, Stanford University, Stanford, CA, 1980.
- [4] F. Reulaux, *The kinematics of machinery*, MacMillan, NY, 1876, Reprint, Dover, NY, 1963.
- [5] E. Rimon and J. W. Burdick, "Mobility of bodies in contact—i: A new 2nd order mobility index for multiple-finger grasps," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 5, pp. 696–708, 1998.
- [6] E. Rimon and J. W. Burdick, "Mobility of bodies in contact—ii: How forces are generated by curvature effects," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 5, pp. 709–717, 1998.
- [7] A. Sudsang, J. Ponce, and N. Srinivasa, "Grasping and in-hand manipulation: Experiments with a reconfigurable gripper," *Advanced Robotics*, vol. 12, no. 5, pp. 509–533, December 1998.
- [8] A. Sudsang, J. Ponce, and N. Srinivasa, "Grasping and in-hand manipulation: Geometry and algorithms," *Algorithmica*, vol. 26, pp. 466–493, January 2000.
- [9] E. Rimon and A. Blake, "Caging 2D bodies by one-parameter two-fingered gripping systems," in *IEEE Int. Conf. on Robotics and Automation*, Minneapolis, MN, April 1996, pp. 1458–1464.
- [10] C. Davidson and A. Blake, "Caging planar objects with a three-finger one-parameter gripper," in *IEEE Int. Conf. on Robotics and Automation*, Leuven, Belgium, June 1998, pp. 2722–2727.
- [11] A. Sudsang and J. Ponce, "On grasping and manipulating polygonal objects with disc-shaped robots in the plane," in *IEEE Int. Conf. on Robotics and Automation*, Leuven, Belgium, June 1998, pp. 2740–2746.
- [12] A. Sudsang and J. Ponce, "On manipulating polygonal objects with three 2-dof robots in the plane," in *IEEE Int. Conf. on Robotics and Automation*, Detroit, MI, 1999, pp. 2227–2233.
- [13] Y. Koga, *On Computing Multi-Arm Manipulation Trajectories*, Ph.D. thesis, Stanford University, 1994.
- [14] R. Alami, J.P. Laumond, and T. Siméon, "Two maipulation planning algorithms," in *Proceedings of Workshop on Algorithmic Foundations of Robotics*, Latombe Goldberg, Halperin and Wilson, Eds. 1995, A K Peters.
- [15] Juan Manuel Ahuactzin, Kamal Gupta, and Emmanuel Mazer, "Manipulation planning for redundant robots: A practical approa ch," *International Journal of Robotics Research*, vol. 17, no. 7, pp. 731–747, July 1998.
- [16] H. Inoue, "Force feedback in precise assembly tasks," AI Lab. AIM-308, MIT, 1974.
- [17] D.E. Whitney, "Quasi-static assembly of compliantly supported rigid parts," *J. Dyn. Sys., Meas. Contr.*, vol. 104, pp. 65–77, 1982.
- [18] R. S. Fearing, "Simplified grasping and manipulation with dexterous robot hands," *IEEE Journal of Robotics and Automation*, vol. RA-2, no. 4, pp. 188–195, Dec. 1986.
- [19] M.T. Mason, "Mechanics and planning of manipulator pushing operations," *International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.
- [20] M. Mani and E.D.W. Wilson, "A programmable orienting system for flat parts," in *North American Mfg. Research Conf. XIII*, 1985.
- [21] R.C. Brost, "Automatic grasp planning in the presence of uncertainty," *International Journal of Robotics Research*, vol. 7, no. 1, pp. 3–17, 1988.
- [22] M. Erdmann and M. T. Mason, "An exploration of sensorless manipulation," *IEEE Journal of Robotics and Automation*, vol. 4, no. 4, pp. 369–379, Aug. 1988.
- [23] M.A. Peshkin and A.C. Sanderson, "Planning robotic manipulation strategies for workpieces that slide," *IEEE Journal of Robotics and Automation*, vol. 4, no. 5, 1988.
- [24] K.Y. Goldberg, "Orienting polygonal parts without sensors," *Algorithmica*, vol. 10, no. 2, pp. 201–225, 1993.
- [25] T. Abell and M. A. Erdmann, "Stably supported rotations of a planar polygon with two frictionless contacts," *IEEE/RSJ International Conference on Robots and Systems*, vol. 3, pp. 411–418, August 1995.
- [26] A.S. Rao and K.Y. Goldberg, "Manipulating algebraic parts in the plane," *IEEE Transactions on Robotics and Automation*, pp. 598–602, 1995.
- [27] S. Akella, W. H. Huang, K. M. Lynch, and M. T. Mason, "Planar manipulation on a conveyor with a one joint robot," in *Robotics Research: The Seventh International Symposium*, G. Giralt and G. Hirzinger, Eds., London, UK, 1996, pp. 265–276, Springer.
- [28] S. Leveroni and K. Salisbury, "Reorienting objects with a robot hand using grasp gaits," in *Robotics Research: The Seventh International Symposium*, G. Giralt and G. Hirzinger, Eds., London, UK, 1996, pp. 39–51, Springer.
- [29] F. Avnaim, J.-D. Boissonnat, and B. Faverjont, "A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles," Tech. Rep. 890, INRIA, 1988.
- [30] J. Barraquand and J.-C. Latombe, "Robot motion planning: a distributed representation approach," *International Journal of Robotics Research*, vol. 10, pp. 628–649, 1991.
- [31] Lydia E. Kavraki, Petr Švestka, Jean-Claude Latombe, and Mark H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug. 1996.
- [32] X. Markenscoff, L. Ni, and C.H. Papadimitriou, "The geometry of grasping," *International Journal of Robotics Research*, vol. 9, no. 1, pp. 61–74, February 1990.
- [33] B. Mishra, J.T. Schwartz, and M. Sharir, "On the existence and synthesis of multifinger positive grips," *Algorithmica, Special Issue: Robotics*, vol. 2, no. 4, pp. 541–558, November 1987.

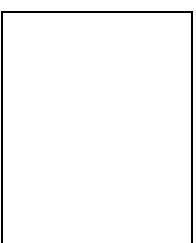
- [34] A.P. Morgan, *Solving Polynomial Systems using Continuation for Engineering and Scientific Problems*, Prentice Hall, Englewood Cliffs, NJ, 1987.
- [35] K.M. Lynch and M.T. Mason, "Stable pushing: mechanics, controllability, and planning," in *Algorithmic Foundations of Robotics*, K.Y. Goldberg, D. Halperin, J.-C. Latombe, and R. Wilson, Eds., pp. 239–262. A.K. Peters, 1995.
- [36] E. Welzl, "Smallest enclosing disks (balls and ellipsoids)," in *New Results and New Trends in Computer Science, LNCS 555*, H. Maurer, Ed., pp. 359–370. Springer Verlag, 1991.
- [37] V. Vaishnavi and D. Wood, "Data structures for the rectangle containment and enclosure problems," *Computer Graphics and Image Processing*, vol. 13, pp. 372–384, 1980.
- [38] V. Vaishnavi and D. Wood, "Rectilinear line segment intersection, layered segment trees, and dynamization," *J. Algorithms*, vol. 3, pp. 160–176, 1982.
- [39] J.-C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, 1991.
- [40] Nicola Simeon, "Move2D (software)," <http://www.laas.fr/~nic/G2D/RIA-g2d.html>.
- [41] S. Fleury, T. Baron, and M. Herrb, "Monocular localization of a mobile robot. LAAS Report 92451," in *International Conference on Intelligent Autonomous Systems (IAS-3)*, Pittsburgh (USA), February 15-18, 1993, pp. 470–479.
- [42] LEDA, "Library of Efficient Data types and Algorithms," Distributed by Algorithmic Solutions, GmbH. <http://www.algorithmic-solutions.com>.
- [43] A. Sudsang, *Geometry and algorithms for part fixturing, grasping and manipulation with modular fixturing elements, a new reconfigurable gripper and mobile robots*, Ph.D. thesis, University of Illinois at Urbana-Champaign, 1999.
- [44] J.Spletzer, A.K.Das, R. Fierro, C.J. Taylor, V. Kumar, and J.P. Ostrowski, "Cooperative localization and control for multi-robot manipulation," in *IROS01*, 2001.



Attawith Sudsang received the B.Eng. degree in computer engineering from Chulalongkorn University, Thailand, in 1991, the M.S. and Ph.D. degrees in computer science from the University of Illinois at Urbana-Champaign in 1994 and 1999 respectively. Part of the work presented in this paper was done while he was completing his doctoral research at UIUC. He was a research associate at the Department of Computer Science, Rice University from 1999 to 2000. Since January, 2001, he has been a Lecturer at the Department of Computer Engineering, Chulalongkorn University. His research interests include distributed manipulation, part fixturing and grasping.



Fred Rothganger received his M.S. degree in Computer Science from University of Massachusetts, Boston, in 1997. He is currently a doctoral student working under Jean Ponce at the University of Illinois, Urbana-Champaign. His research interests are in vision and mobile robotics. His current work is in automatic construction of models for recognition.



Jean Ponce is a Professor in the Department of Computer Science and the Beckman Institute at the University of Illinois at Urbana-Champaign. Dr. Ponce's main research interests are in Computer Vision and Robotics. His current Computer Vision research focuses on automatic 3D model construction from image data, interactive image synthesis from video streams, motion analysis, 3D shape representation and object recognition. His Robotics work focuses on fixture, grasp and manipulation planning, and he has recently developed a novel reconfigurable gripper and a part feeder that automatically adapts to new part geometry. Dr. Ponce is the author

of over a hundred technical publications and he is finishing, in collaboration with Dr. David Forsyth from UC Berkeley, a Computer Vision textbook that will be out in the Summer of 2002. He is an associate editor of the International Journal of Computer Vision and was an area editor of Computer Vision and Image Understanding (1994-2000) and an associate editor of the IEEE Transactions on Robotics and Automation (1996-2001). He was program co-chair of the 1997 IEEE Conference on Computer Vision and Pattern Recognition and served as general co-chair of the year 2000 edition of this conference.