Experimental Results on Planning the Coordinated Paths of two Circular Robots [*]

José María Bañón †, Olmedo Arcila †, and Jaime Arango ‡
Departamento Ciencias de la Computación †
Departamento de Matemáticas ‡
Universidad del Valle, Cali, Colombia.
E-mail : {banon, arcila}@borabora.univalle.edu.co,
jarango@hypatia.univalle.edu.co

Abstract

We present a novel implementation of the Schwartz and Sharir's exact algorithm [1] for planning the free-motion of two circular robots amidst polygonal obstacles. Additionally, some empirical results are reported in order to show the feasibility of the

Key Words: Artifitial Intelligence, Motion Planning, Configuration Space, Exact Cell Decomposition.

1 Introduction

Coordinated motion planning of several independent robots is an important class of robot motion planning problems [2, 3]. Significant work [2] has been directed toward the development of efficient algorithms to solve the multiple-robot path planning problem. Most of these algorithms are based on approximated methods which are not guaranteed to find a collision-free motion. Although, efficient and exact algorithms have been developed for specific multiple-robot motion planning problems [3, 4, 5] very little empirical work [6] has been reported about the implementation of exact algorithms for planning collision-free paths for several robots. This paper is devoted to cover the lack of implementation and empirical experimentation of exact path planning algorithms for several robots. Exact methods are important for robotics systems with few degrees of freedom ($d.o.f. \leq 4$) confined in reduced environments where difficult maneuvers are necessary to achieve the desired goal configuration of the robots.

Experimental work in exact planners for several robots has been reported by Drace-Wright et al. [6]. They implemented an algorithm for planning the motion of two polygonal robots based in an exact cell decomposition method of the configuration

[*]Trabajo parcialmente financiado por *COLCIENCIAS*

space with the use of Minkowsky operations. This algorithm has been used by Alami et al. [7] to describe a method for solving the manipulation planning problem in the case of a polygonal robot and a movable polygonal object moving in translation amidst polygonal obstacles.

In this paper, we present a novel implementation of the Schwartz and Sharir's exact algorithm [1] for planning the free-motion of two circular robots amidst polygonal obstacles. We first overview the algorithm; then we describe its implementation and we report some empirical results. Finally, we conclude and describe remaining problems.

2 Overview of the Algorithm

Let us consider a system composed of two circular robots B_1 and B_2 with radii r_1 and r_2 respectively and $r_1 > r_2$.

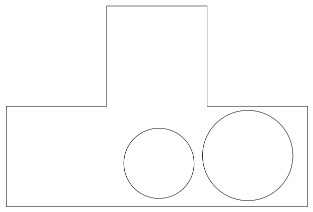


Figure 1: *Workspace.*

The workspace $\mathcal{W} \subset \mathbf{R}^2$ is an open set of the Euclidean plane bounded by a finite set of disjoint polygonal obstacles $\{\mathcal{O}_i\}_{i=1}^n$. We denote by $\mathbf{P}_1$ and $\mathbf{P}_2$ the positions of the centers of B_1 and B_2 respectively. The subset $A_1 \subset \mathcal{W}$ of the positions $\mathbf{P}_1$ such that B_1 does not touch any obstacle is called admissible space for B_1. Analogously, A_2 is the admissible space for B_2. For example, we consider the workspace in Figure 1.

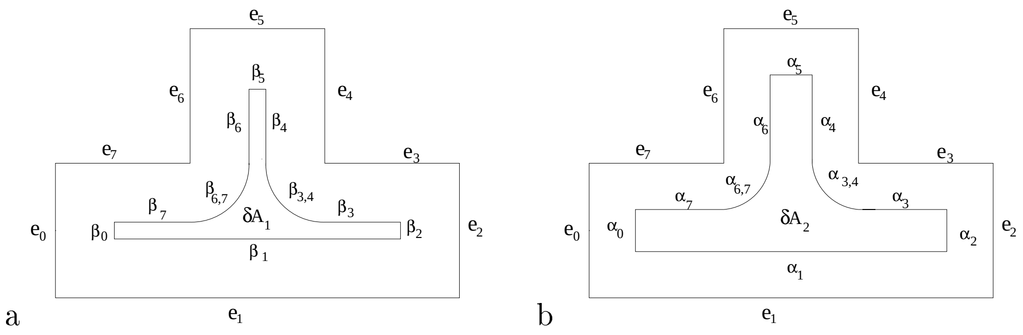


Figure 2: *Admissible Sets A_1 and A_2.*

Figure 2 shows the admissible sets $\mathcal{A}_1$ and $\mathcal{A}_2$. The boundary contour of $\mathcal{A}_1$ is formed by the following sequences of straight segments and circular arcs: $(\beta_0, \beta_1, \beta_2, \beta_3, \beta_{3,4}, \beta_4, \beta_5, \beta_6, \beta_{6,7}, \beta_7)$. The boundary contour of $\mathcal{A}_2$ is formed by the following sequences of straight segments and circular arcs: $(\alpha_0, \alpha_1, \alpha_2, \alpha_3, \alpha_{3,4}, \alpha_4, \alpha_5, \alpha_6, \alpha_{6,7}, \alpha_7)$.

The straight segments $\{\beta_i\}_{i=0}^{7}$ and $\{\alpha_i\}_{i=0}^{7}$ are generated by the edges $\{e_i\}_{i=0}^{7}$ of the obstacle. The circular arcs $\{\beta_{3,4}, \beta_{6,7}\}$ and $\{\alpha_{3,4}, \alpha_{6,7}\}$ are generated by the concave vertex $e_{3,4}$ and $e_{6,7}$ where $e_{i,j}$ is the common vextex of e_i and e_j.

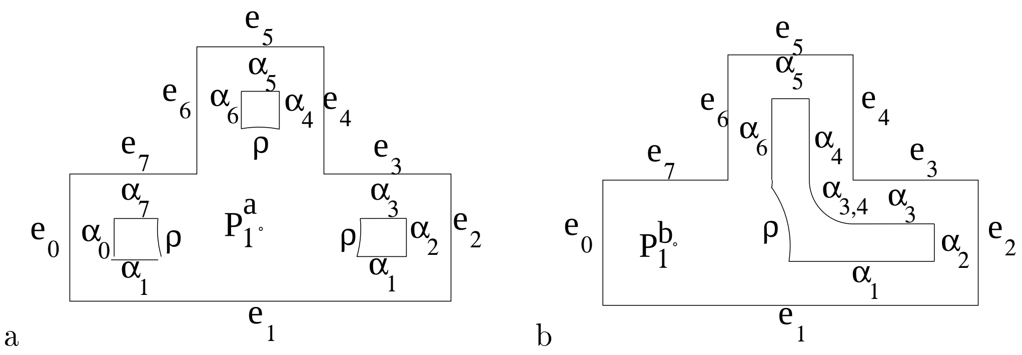


Figure 3: *Components.*

The free configuration space of the system $\mathcal{C}_{free} \subset \mathbf{R}^4$ is the set of configurations $\mathbf{q} = (\mathbf{P}_1, \mathbf{P}_2)$ such that neither B_1 nor B_2 meet each other nor any obstacle. The motion planning problem for the considered system consists in finding a collision-free path linking the initial $\mathbf{q}_{init} = (\mathbf{P}_1^{init}, \mathbf{P}_2^{init})$ and the goal $\mathbf{q}_{goal} = (\mathbf{P}_1^{goal}, \mathbf{P}_2^{goal})$ configurations within the free space $\mathcal{C}_{free}$. A collision-free path solution trajectory is a continuous mapping $u \in [0,1] \mapsto \mathcal{C}_{free}$ such that $u(0) = \mathbf{q}_{init}$ and $u(1) = \mathbf{q}_{goal}$.

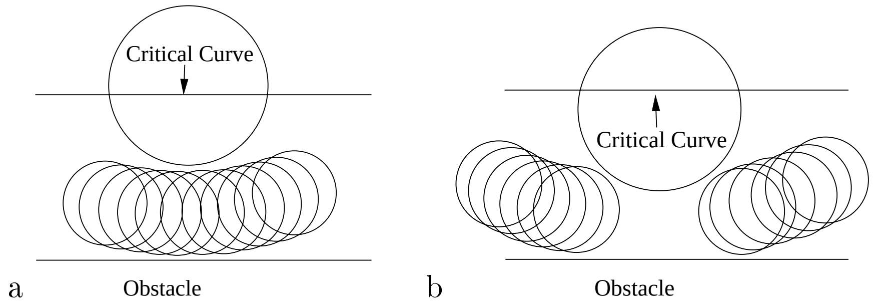


Figure 4: *Crossing a Critical Curve.*

For any position $\mathbf{P}_1$ of B_1 the set $A_2 - \rho(\mathbf{P}_1)$ represents the admissible positions of B_2 in presence of the robot B_1, where $\rho(\mathbf{P})$ is the circle of radius $r_1 + r_2$ centered at $\mathbf{P}$. The set $A_2 - \rho(\mathbf{P}_1)$ is the union $\bigcup_{i \in I_C} \mathcal{K}_i$ of a finite number of disjoint connected open regions $\mathcal{K}_i$ called components. Figures 3.a and 3.b show $A_2 - \rho(\mathbf{P}_1)$ for two different positions, $\mathbf{P}_1^a$ and $\mathbf{P}_1^b$, of B_1; $A_2 - \rho(\mathbf{P}_1^a)$ has three components and $A_2 - \rho(\mathbf{P}_1^b)$ has one component.

The algorithm [1] constructs an exact cell decomposition of the manifold $\mathcal{C}_{free}$ by projecting it onto a two-dimensional space corresponding to fixed positions of the center of B_1. Such a decomposition is based in the fact that the topology of the connected components $\mathcal{K}_i$ of $A_2 - \rho(\mathbf{P}_1)$ changes for specific positions of B_1, called critical points, in which the connected components may appear, disappear, be split or merged. Figures 4.a and 4.b show how the pattern of motion changes as B_1 crosses a critical curve.

A point $\mathbf{P}_1 \in A_1$ is a critical point if either $\rho(\mathbf{P}_1)$ is tangent to the boundary of A_2 or it passes through some vertex of A_2. The locus of critical points consists on the following curves: i) straight line segments lying at a distance $r_1 + 2r_2$ of the edges of $\mathcal{W}$, ii) circular arcs of radius $r_1 + 2r_2$ centered at each convex vertex of $\mathcal{W}$, and iii) circular arcs of radius $r_1 + r_2$ centered at each convex vertex of the boundary of A_2. Figures 5 a) and 5 b) shows all types of critical curves. The critical curves divide A_1 in a finite number of disjoint connected open regions which are called non-critical regions. A non-critical region is a maximal subset of A_1 which intersects no critical curve.

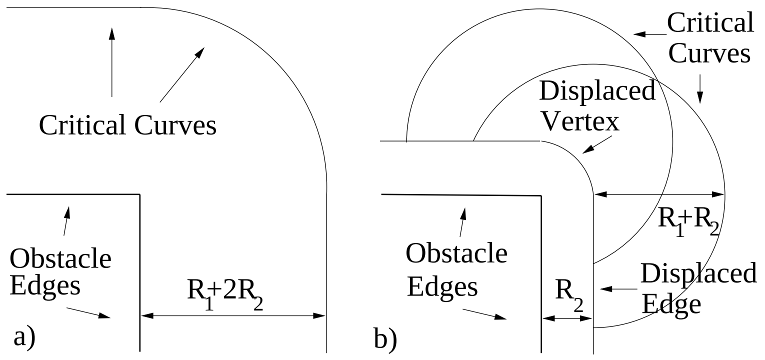


Figure 5: *Critical curves.*

The boundary contours of a component $\mathcal{K}$ of $A_2 - \rho(\mathbf{P}_1)$ is formed by: i) line segments of the boundary of A_2 and ii) circular arcs of the boundary of A_2 and of the circumference $\rho(\mathbf{P}_1)$. We define and denote by $\lambda(\mathcal{K})$ the labeling of a component $\mathcal{K}$ as the circular sequence of the indices of the boundary elements of the exterior boundary of $\mathcal{K}$. An important result lying at the core of the algorithm is that for any point $\mathbf{P}_1 \in \mathcal{R}$ belonging to a fixed non-critical region $\mathcal{R}$, the number of components of $A_2 - \rho(\mathbf{P}_1)$ and its labeling are independent of the position $\mathbf{P}_1$; they depend on the non-critical region $\mathcal{R}$. This result is illustrated in Figure 6.a and 6.b showing the components of two different positions, $\mathbf{P}_1^a$ and $\mathbf{P}_1^b$, of the same non-critical region. Note that both positions have three components $\{\mathcal{K}_1^a, \mathcal{K}_1^a, \mathcal{K}_1^a\}$ and $\{\mathcal{K}_1^b, \mathcal{K}_1^b, \mathcal{K}_1^b\}$; for each $i = 1, .., 3$. $\mathcal{K}_i^a$ and $\mathcal{K}_i^b$ have the same labeling. This result allows to define an equivalence relation of components: two components $\mathcal{K}_1 \subset A_2 - \rho(\mathbf{P}_1)$ and $\mathcal{K}_1' \subset A_2 - \rho(\mathbf{P}_1')$, where $\mathbf{P}_1, \mathbf{P}_1' \in \mathcal{R}$, are equivalent if they have the same labeling, $\lambda(\mathcal{K}_1) = \lambda(\mathcal{K}_1')$. We denote by $\sigma(\mathcal{R})$ the set of equivalence classes of components of $\mathcal{R}$.

A cell is a pair $[\mathcal{R}, \mathcal{T}]$ where $\mathcal{R}$ is a non-critical region and $\mathcal{T} \in \sigma(\mathcal{R})$. The set of all cells is an exact cell decomposition of $\mathcal{C}_{free}$. The connectivity graph CG is the graph whose nodes are the cells and whose arcs connect pairs of adjacent cells. Two cells are adjacent if and only if they satisfied the crossing rules defined in [1]. A sequence of adjacent cells $c_1, c_2, ..., c_n$ is called a channel. The continuous problem of finding a collision-free trajectory linking the configurations $\mathbf{q}_{init}$ and $\mathbf{q}_{goal}$ is reduced to searching in CG for channel $c_{init}, ..., c_{goal}$ which gives a sequence of cells connecting the start and goal configurations along which the robot can move without collisions with obstacles. A solution trajectory is then generated within the found channel.

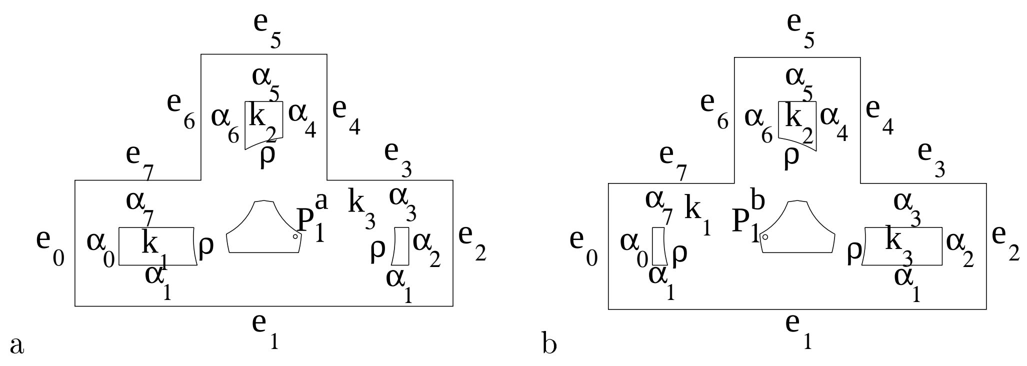


Figure 6: *Components.*

3 Implementation of the Algorithm

In this section, an exact cell decomposition scheme generation for the implementation of the above algorithm is presented. A similar approach has also been proposed by Bañón [9] for the implementation of the ladder algorithm [10].

Calculation of Admissible Spaces: The geometry of the polygonal obstacles is processed to determine the admissible spaces A_1 and A_2. The calculation of A_1 consists of several steps. We first grow isotropically the obstacles $\{\mathcal{O}_i\}_{i=1}^n$, by the circle B_1 by constructing the Minkowski sum with B_1; then we form the union $\mathcal{O}_g$ of the grown obstacles; finally we obtain the admissible space $A_1 = \mathcal{W} - \mathcal{O}_g$. Analogously we calculate A_2. The boundaries, $\partial\mathcal{A}_1$ and $\partial\mathcal{A}_2$, are composed by a finite sequence of straight segments and circular arcs.

Calculation of Critical Curve Sections: Here, we first compute the set of all the critical curves. Then we find the intersection points of all the critical curves with each other and with the boundary $\partial\mathcal{A}_1$. Intersection points that are exterior to $\mathcal{A}_1$ are discarded. The governing equations for obtaining those intersection points reduce to find roots of polynomials with degree less or equal than two, which are easily solved analytically. Since it is necessary to sort the intersections points along each critical curve we have adopted the parametric representation for the curve segments. The parametric equations of line segments joining the points (x_1, y_1) and (x_2, y_2) are: $x(t) = x_1 + t\,(x_2 - x_1)$ and $y(t) = y_1 + t\,(y_2 - y_1)$, where $t \in [0, 1]$. Circular arcs of radius r accept the following rational parametrization:

$$\forall t \in [-1,1): \quad x(t) = \frac{2\,t\,r}{1+t}, \quad y(t) = \frac{r\,(1-t)}{1+t},$$
$$\forall t \in [1,3): \quad x(t) = \frac{-2\,r\,(t-2)}{t-1}, \quad y(t) = \frac{-r\,(3-t)}{t-1}.$$

Once the candidate intersection point (x_i, y_i) is obtained we check for true intersection by determining the parameter values t of the point (x_i, y_i) along the two intersecting curves and checking if they belong to the interval of definition of the curve segments.

For each critical curve we construct the ordered list $\{\mathbf{P}_j\}_{j=1}^N$ of its intersection points according to their order in the curve. This is easily accomplished by sorting the points with respect to their parameter values t in the curve. From the ordered list of intersection points of each critical curve we construct the critical curve sections, $\mathcal{S}(\mathbf{P}_j, \mathbf{P}_{j+1})$, which are defined to be the portion of a critical curve lying between two consecutive intersection points $\mathbf{P}_j$ and $\mathbf{P}_{j+1}$. We indicate by $\mathcal{S}(\mathbf{P}_1, \mathbf{P}_2)$ the critical section with endpoints $\mathbf{P}_1$ and $\mathbf{P}_2$.

Construction of Non-Critical Regions: Non-critical regions are determined by identifying their boundary contours which consist in circular lists of critical curve sections. The identification of adjacent critical sections involves a topological sorting of the critical sections which can be done in two steps: 1) building the map $\mathbf{P} \mapsto \mathcal{L}(\mathbf{P})$, where $\mathbf{P}$ is an intersection point and $\mathcal{L}(\mathbf{P}) = \{\mathcal{S}(\mathbf{P}_a, \mathbf{P}_b) : \mathbf{P} = \mathbf{P}_a \text{ } or \text{ } \mathbf{P} = \mathbf{P}_b\}$ is the set of critical curve sections having endpoint $\mathbf{P}$, 2) sorting the sections $\mathcal{L}(\mathbf{P})$ in clockwise order with respect to the outgoing tangential directions of the curve section at $\mathbf{P}$. Step 1 is done at the same time we build the critical curve sections. Here, it is important to have a precise representation of the orientation of the sections [9]. Each section $\mathcal{S}(\mathbf{P}_1, \mathbf{P}_2)$ is considered as a pair of oppositely oriented sections σ_1 and σ_2; therefore, we consider the pairs section/endpoint as follows: $\sigma_i = (\mathcal{S}, \mathbf{P}_i)$, $i = 1, 2$ which are called outgoing sections. In step 2 we sort the list of outgoing sections $\mathcal{L}(\mathbf{P})$ in clockwise angular order around their common endpoint $\mathbf{P}$. Adjacent outgoing sections are identified as being consecutive sections in the ordered list $\mathcal{L}(\mathbf{P})$. Once all lists of outgoing sections $\mathcal{L}(\mathbf{P})$ are sorted, we proceed to construct the operator $next(,)$ that maps each section $\mathcal{S}$ to its adjacent sections. The idea is the following: starting from one section we get all other boundary sections of the same boundary contour by successive applications of the $next(,)$ operator. We define the $next(,)$ operator as the function that maps the pair $(\mathcal{S}, \mathbf{P})$ to the section that follows $\mathcal{S}$ in the ordered list of curve sections $\mathcal{L}(\mathbf{P})$. Figure 7 illustrates the definition of the $next(,)$ operator. We obtain the sections $\mathcal{S}' = next(\mathcal{S}, \mathbf{P})$ and $\mathcal{S}'' = next(\mathcal{S}, \mathbf{Q})$ by applying respectively the $next(,)$ operator to the section $\mathcal{S}$ at both endpoints, $\mathbf{P}$ and $\mathbf{Q}$. Sections $\mathcal{S}$ and $\mathcal{S}'$ are consecutive sections boundaries of the non-critical region $\mathcal{R}_1$. Sections $\mathcal{S}$ and $\mathcal{S}''$ are consecutive sections boundaries of the non-critical region $\mathcal{R}_2$. Non-critical regions $\mathcal{R}_1$ and $\mathcal{R}_2$ have the common section boundary $\mathcal{S}$.

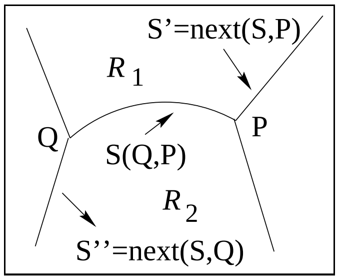


Figure 7: *Definition of $next(,)$.*

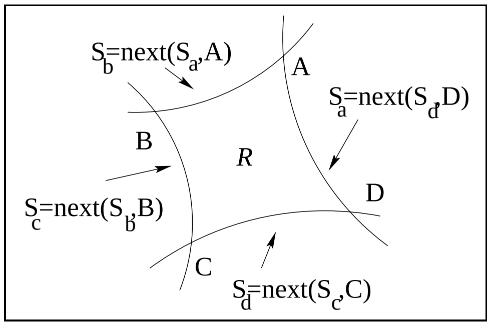


Figure 8: *Building a non-critical region.*

The successive application of the $next(,)$ operator to a section $\mathcal{S}$ allows us to construct the boundary contour of the non-critical regions that have $\mathcal{S}$ as a boundary section. Figure 8 shows an example of the use of the $next(,)$ operator in the determination of a non-critical section; here, the boundary contour of the non-critical region is formed by the boundary sections $\{\mathcal{S}_a, \mathcal{S}_b, \mathcal{S}_c, \mathcal{S}_d\}$. Starting at section $\mathcal{S}_a$, the others sections are obtained as follows: $\mathcal{S}_b = next(\mathcal{S}_a, \mathbf{A})$, $\mathcal{S}_c = next(\mathcal{S}_b, \mathbf{B})$, $\mathcal{S}_d = next(\mathcal{S}_c, \mathbf{C})$. The procedure terminates when in the next iteration we obtain that $next(\mathcal{S}_d, \mathbf{D})$ is the starting section $\mathcal{S}_a$.

Determination of the Components: Having calculated the set of all non-critical regions, we proceed to study the set of equivalence classes $\sigma(\mathcal{R})$ of components that arise in each non-critical region $\mathcal{R}$. Particularly, we need to determine the number of equivalence classes and their labeling. As explained in section 2, this can be accomplished by choosing a particular position $\mathbf{P}_1 \in \mathcal{R}$ and analyzing the components that arise in the set $\mathcal{A}_2 - \rho(\mathbf{P}_1)$. To select the position $\mathbf{P}_1$ we determine a line segment $\overline{\mathbf{FG}} \subset \overline{\mathcal{R}}$ and we take $\mathbf{P}_1$ as the midpoint of the segment $\overline{\mathbf{FG}}$. The point $\mathbf{F}$ is an interior point of a boundary section $\mathcal{S}$ of $\mathcal{R}$. The other point $\mathbf{G}$ is calculated by taking the normal line $\mathcal{N}$ to $\mathcal{S}$ at $\mathbf{F}$, and computing the intersection points of $\mathcal{N}$ with the boundary curve sections of $\mathcal{R}$. Analyzing the intersection points, it is easy to take the point $\mathbf{G}$ such that the segment $\overline{\mathbf{FG}} \subset \overline{\mathcal{R}}$. We have not taken into account the degeneracies that may occur in this procedure. The decomposition of $\mathcal{A}_2 - \rho(\mathbf{P}_1)$ in components is achieved by calculating the intersection points between the boundary of $\mathcal{A}_2$ and the circle $\rho(\mathbf{P}_1)$. The number of intersections points has to be even. From the intersection points we deduce the number of components and their labeling.

Cell Decomposition: We determine the cells by forming all the pairs $[\mathcal{R}, \mathcal{T}]$ where $\mathcal{R}$ is a non-critical region and $\mathcal{T} \in \sigma(\mathcal{R})$. At this point, we have an exact cell decomposition of the free-configuration space $\mathcal{C}_{free}$. It it important to note that the geometrically complex cells are not explicity generated; rather, the simpler two dimensional non-critical regions and its components are constructed, and the cells only implicitly represented.

Building the Connectivity Graph: In order to build the connectivity graph CG we connect two cells c_1 and c_2 with non-critical regions $\mathcal{R}_1$ and $\mathcal{R}_2$ sharing a common boundary if and only if they satisfy the following crossing rules (see [1]):

1. Connect the cell $c_1 = [\mathcal{R}_1, \mathcal{T}]$ to cell $c_2 = [\mathcal{R}_2, \mathcal{T}]$ for each $\mathcal{T} \in \sigma(\mathcal{R}_1)$ and $\mathcal{T} \in \sigma(\mathcal{R}_2)$.

2. Connect the cell $c_1 = [\mathcal{R}_1, \mathcal{T}_1]$ to cell $c_2 = [\mathcal{R}_2, \mathcal{T}_2]$ when $\mathcal{T}_1 \in \sigma(\mathcal{R}_1) - \sigma(\mathcal{R}_2)$ and $\mathcal{T}_2 \in \sigma(\mathcal{R}_2) - \sigma(\mathcal{R}_1)$.

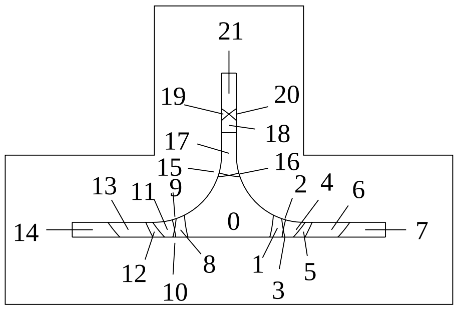
Identification of Initial and Goal Cells: Here, we have to determine the initial $c_{init} = [\mathcal{R}_{init}, \mathcal{T}_{init}]$ and goal cells $c_{goal} = [\mathcal{R}_{goal}, \mathcal{T}_{goal}]$ containing the initial $\mathbf{q}_{init} = (\mathbf{P}_1^{init}, \mathbf{P}_2^{init})$ and goal $\mathbf{q}_{goal} = (\mathbf{P}_1^{goal}, \mathbf{P}_2^{goal})$ free configurations. We do this in two steps:

1. Determination of the initial and goal non-critical regions: $\mathcal{R}_{init}$ and $\mathcal{R}_{goal}$. We consider a line passing through the points $\mathbf{P}_1^{init}$ and $\mathbf{P}_1^{goal}$, we compute the intersections between the line and all the critical curve sections and we sort the intersection points with respect to their order in the line. By inspecting the intersection points closest to $\mathbf{P}_1^{init}$ and $\mathbf{P}_1^{goal}$ it is easy to deduce $\mathcal{R}_{init}$ and $\mathcal{R}_{goal}$.

2. Determination of the initial and goal components: T_{init} and T_{goal}: we calculate the boundary contours of the components of $\mathcal{A}_2 - \rho(\mathbf{P}_1^{init})$, draw a line through the point $\mathbf{P}_2^{init}$ and calculate the intersection points with the boundary contours of the calculated components. By studying the closest intersection points to $\mathbf{P}_2^{init}$ is easy to determine T_{init}. Similarly, we determine T_{goal}.

4 Results

We have implemented the algorithm in C on a Sun-Sparc 20 platform and performed several experiments. The implementation includes the exact cell decomposition scheme and the graph searching method. To illustrate a simulation of our implemented planner we consider the situation in which the two robots are moving in the workspace showed in Figure 1. Figure 9 shows the critical curves of the system and how these critical curves partition the admissible space A_1 into 22 non-critical regions. Table 1 describes the labeling of the components of each non-critical region and the nodes in the connectivity graph. Each node is labeled by the number identifying the corresponding non-critical region, and by one the symbols L, R and U designating respectively a connected component lying on the left, right and upper side of the region. Figure 10 shows the connectivity graph CG which consists in 45 nodes and 52 edges. In this example, the processing time for the calculation of CG amounts 50 milliseconds.



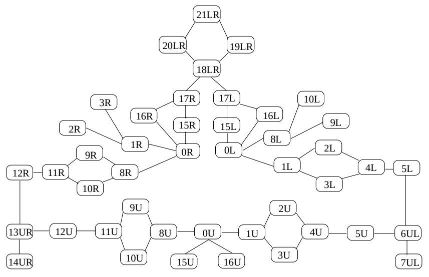


Figure 9: *Critical curves and non-critical regions.* Figure 10: *The connectivity graph.*

Figure 11 shows snapshots along the computed solution trajectory found by our planner to move the robots from an initial position to a goal position. The initial and goal position of robot B_1 are in the non-critical regions 19 and 16 respectively (see Figure 9). Figure 12 illustrates with detail the coordinated motion of both robots as a sequence of alternating motions in order to reach the desired goal. Figure 12 shows more results obtained with our implemented planner.

Finally, we consider the particularly challenging situation in which the two robots are moving through the inside of a regular heptagon. The sizes of the circles have

Región	Componente	Nodo
0	$\rho\alpha_7\alpha_0\alpha_1$	0L
	$\rho\alpha_4\alpha_5\alpha_6$	0U
	$\rho\alpha_1\alpha_2\alpha_3$	0R
1	$\rho\alpha_7\alpha_0\alpha_1$	1L
	$\rho\alpha_4\alpha_5\alpha_6$	1U
	$\rho\alpha_1\alpha_2\alpha_3$	1R
2	$\rho\alpha_7\alpha_0\alpha_1$	2L
	$\rho\alpha_4\alpha_5\alpha_6$	2U
	$\rho\alpha_1\alpha_2$	2R
3	$\rho\alpha_7\alpha_0\alpha_1$	3L
	$\rho\alpha_4\alpha_5\alpha_6$	3U
	$\rho\alpha_2\alpha_3$	3R
4	$\rho\alpha_{6,7}\alpha_7\alpha_0\alpha_1$	4L
	$\rho\alpha_4\alpha_5\alpha_6$	4U
5	$\rho\alpha_{6,7}\alpha_7\alpha_0\alpha_1$	5L
	$\rho\alpha_4\alpha_5\alpha_6$	5U
6	$\rho\alpha_4\alpha_5\alpha_6\alpha_{6,7}\alpha_7\alpha_0\alpha_1$	6UL
7	$\rho\alpha_4\alpha_5\alpha_6\alpha_{6,7}\alpha_7\alpha_0\alpha_1$	7UL
8	$\rho\alpha_7\alpha_0\alpha_1$	8L
	$\rho\alpha_4\alpha_5\alpha_6$	8U
	$\rho\alpha_1\alpha_2\alpha_3$	8R
9	$\rho\alpha_0\alpha_1$	9L
	$\rho\alpha_4\alpha_5\alpha_6$	9U
	$\rho\alpha_1\alpha_2\alpha_3$	9R
10	$\rho\alpha_7\alpha_0$	10L
	$\rho\alpha_4\alpha_5\alpha_6$	10U
	$\rho\alpha_1\alpha_2\alpha_3\alpha_{3,4}$	10R
11	$\rho\alpha_4\alpha_5\alpha_6$	11U
	$\rho\alpha_1\alpha_2\alpha_3\alpha_{3,4}$	11R
12	$\rho\alpha_1\alpha_2\alpha_3$	12R
	$\rho\alpha_4\alpha_5\alpha_6$	12U
13	$\rho\alpha_1\alpha_2\alpha_3\alpha_{3,4}\alpha_4\alpha_5$	13UR
14	$\rho\alpha_1\alpha_2\alpha_3\alpha_{3,4}\alpha_4\alpha_5\alpha_6$	14UR
15	$\rho\alpha_7\alpha_0\alpha_1$	15L
	$\rho\alpha_4\alpha_5$	15U
	$\rho\alpha_1\alpha_2\alpha_3$	15R
16	$\rho\alpha_7\alpha_0\alpha_1$	16L
	$\rho\alpha_5\alpha_6$	16U
	$\rho\alpha_1\alpha_2\alpha_3$	16R
17	$\rho\alpha_7\alpha_0\alpha_1$	17L
	$\rho\alpha_1\alpha_2\alpha_3$	17R
18	$\rho\alpha_7\alpha_0\alpha_1\alpha_2\alpha_3$	18LR
19	$\rho\alpha_7\alpha_0\alpha_1\alpha_2\alpha_3$	19LR
20	$\rho\alpha_7\alpha_0\alpha_1\alpha_2\alpha_3$	20LR
21	$\rho\alpha_7\alpha_0\alpha_1\alpha_2\alpha_3\alpha_{3,4}$	21LR

Table 1: Non-Critical Regions, Components and Nodes.

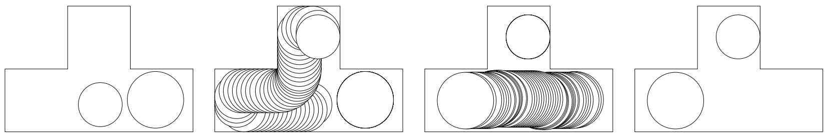


Figure 11: *A coordinated motion path.*

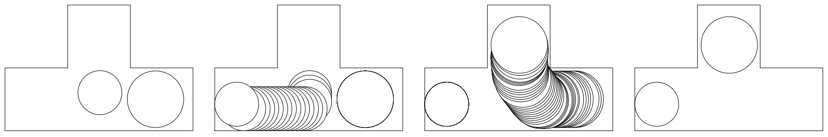


Figure 12: *A coordinated motion path.*

been chosen so that the two robots can barely fit in the interior of the heptagon. This workspace was chosen with the intention of making our planner generate a path consisting in strongly coordinated motions. Figure 13 illustrates with detail the strongly coordinated motion of both robots as a sequence of alternating motions.

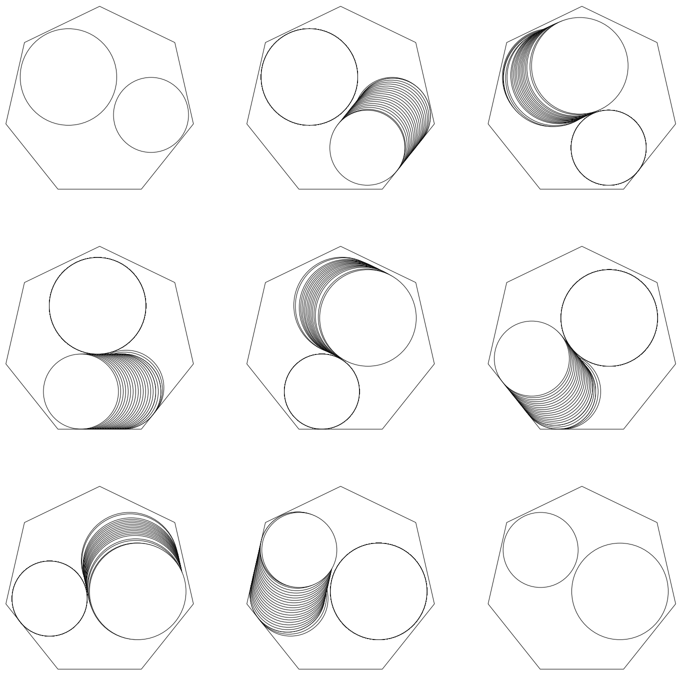


Figure 13: *A Coordinated Motion Path.*

5 Conclusions

In this paper we have considered the difficult problem of the implementation of exact collision path planning algorithms for several robots. In particular, we have presented a comprehensive implementation of the Schwartz and Sharir [1] algorithm for planning the coordinated motion of two circular robots in the plane. An exact cell decomposition of $\mathcal{C}_{free}$ has been implemented by means of efficient tools of computational geometry. Our experimental results confirmed the effectiveness of the exact cell decomposition scheme proposed and demonstrated the applicability of the algorithm we have developed. We have to mention that our approach requires further work to solve in general the problem of trajectory generation within the found channel. This problem represents a challenge to motion planning techniques because of its additional complexity.

Acknowledgments: This work was supported by *COLCIENCIAS*.

References

[1] J. T. Schwartz and M. Sharir, *On the piano movers problem: III. Coordinating the motion of several independent bodies: The special cases of circular bodies moving amidst polygonal barriers*, International Journal of Robotics Research, 2(3), (1983), pp. 46-75.

[2] J. C. Latombe, *Robot Motion planning*, Kluwer Academic Publishers, 1991.

[3] M. Sharir and S. Sifrony, *Coordinated motion planning for two independent robots*, Ann. Math. Artif. Intell., 3, (1991), pp. 107-130.

[4] M. Sharir, *Algorithmic Motion Planning*, ch. 40 of Handbook of Discrete and Computational Geometry, J.E. Goodman and eds., CRC Press, 1997.

[5] C. Yap, *Coordinating the motion of several disks*, Technical Report 14, (1984), Courant Institute, New York.

[6] B. Drace-Wright, J. P. Laumond, and R. Alami, *Motion planning of a robot and a movable object amidst polygonal obstacles*, IEEE Proceedings of the International Conference on Robotics and Automation, 1992, pp. 2474-2480.

[7] R. Alami, J. P. Laumond, and T. Simeon, *Two manipulation planning algorithms*, Algorithmic Foundation of Robotics, Ed. A K Peters Ltda, pp. 109-125, 1995.

[8] F. Avnain, J. D. Boissonnat and B. Faverjon, *A practical exact motion planning algorithm for polygonal objects amidst polygonal obstacles*, Technical report 890, INRIA, Sophia-Antipolis, France, 1988.

[9] J. Bañón, *Implementation and extension of the ladder algorithm*, IEEE Proceedings of the International Conference on Robotics and Automation, (1990), pp. 1548-1553.

[10] J. T. Schwartz and M. Sharir, *On the piano movers problem: I. The case of a two-dimensional rigid polygonal body moving amidst polygonal barriers.* Communication on Pure and Applied Mathematics, 36, (1983), pp. 345-398.