Proceedings of the 1992 IEEE
International Conference on Robotics and Automation
Nice, France - May 1992

Motion planning for a robot and a movable object amidst polygonal obstacles.

Benoît Dacre-Wright, Jean-Paul Laumond, Rachid Alami
LAAS / CNRS
7, Avenue du Colonel Roche
31077 Toulouse - France

Abstract

This paper addresses the motion planning problem for a robot and a movable object amidst polygonal obstacles. Motion planning in this context appears as a constrained instance of the coordinated motion planning problem for two robots. Indeed, the object cannot move by itself, it only does when the robot grasps it.

We first show a topological property that characterizes the existence of solutions in the subspace of configurations where the robot "touches" the object. This property then provides a general resolution scheme which is applied to the special case of a convex polygonal robot moving in translation amidst a convex polygonal object and polygonal obstacles. The algorithm is complete and exact. It extends a first pioneering study provided by Wilfong in [12].

1 Introduction

Robot motion planning usually consists in planning collision-free paths for a robot moving amidst fixed obstacles. Nevertheless a robot may have to perform tasks which are more difficult than planning motions only for itself. In some situations, a robot may be able to move objects and to change the structure of its environment. In such a context, the robot moves amidst obstacles but also movable objects. A movable object cannot move by itself; it can move only if it is grasped by the robot. According to the standard terminology, considering movable objects appears as a constrained instance of the *coordinated motion planning problem.* We propose to call such a problem the *manipulation planning problem.* Indeed, the system will have to plan how the robot must manipulate some objects in the environment in order to reach a given situation (see Figure 1).

While the problem is crucial in practical applications, motion planning in the presence of movable objects has attracted only few researchers at this time. An introduction to the problem can be found in [6]. Wilfong [12] gave the first results on the complexity of the problem : he proved that the problem is $PSPACE$-hard (resp. NP-hard) in two dimensional environments where only translations are allowed and when the final configuration specifies (resp. does not specify) the final positions of all the movable objects. In the same reference, Wilfong gives a solution in $O(n^3 \log^2 n)$ (where n is the number of vertices of a polygonal environment) for the case of a convex polygonal robot moving in translation amidst one polygonal object and obtacles; however the robot is able to grasp the object only from a *finite* number of relative position[1].

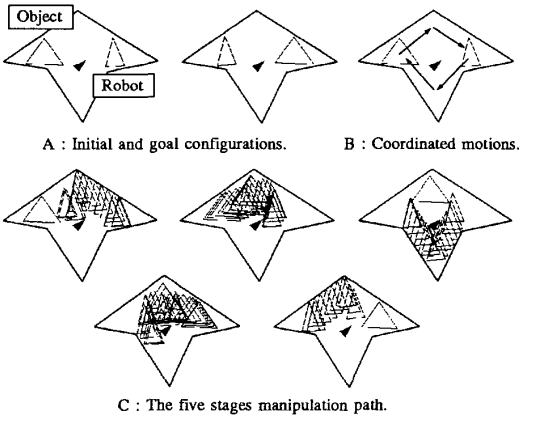

Figure 1: An illustration of manipulation constraints.

When the number of grasping positions is no more finite, the problem becomes more difficult. In [8] we presented a general solution when the grasping positions are defined as configurations where the robot and the movable object are in *contact.* In this case a preliminary study shows how to discretize the space of grasping configurations into a finite number of cells in order to make the problem tractable. However this study used the general tools from algebraic geometry, leading the method to be unefficient in practice.

In this paper we apply this last study to the special case of a convex robot moving in translation amidst a convex polygonal object and polygonal obstacles. The algorithm we propose is complete and exact. It solves

[1] Let us pinpoint the reference [1] that gives a general algorithm for the manipulation problem in the case of a *finite* number of grasping positions and a *finite* number of object placements. [2] gives a heuristic solution to the problem by using a generate-and-test paradigm.

0-8186-2720-4/92 $3.00 ©1992 IEEE

the case where grasping conditions are defined from the contact relationship between the robot and the object. It extends the first solution given by Wilfong.

The paper is organized as follows : we first state the problem and give a resolution scheme from a topological property proving the existence of solutions when both the robot and the object are in a same connected component of contact configurations (Section 2). We then apply this general result to our special case : we show how to use a cell decomposition of the 4-dimensional configuration space (solving the coordinated motion problem) in order to produce a cell decomposition of the contact space (solving the contact motion problem); this last structure is then completed into a *manipulation* graph whose connected components characterize the existence of solutions to the manipulation problem (Section 3). The algorithm has been implemented and Section 4 discusses experimental results.

2 Problem statement and resolution scheme

2.1 An informal statement

In a manipulation problem, we consider three kinds of bodies :

- a robot
- a movable object
- fixed obstacles

A movable object can be displaced only while grasped by the robot. Possible motions are then restricted to two classes :

- A motion of the robot alone, the object being then an additional obstacle.
- A displacement of the object by the robot, which requires to grasp the object. The robot and the object are then rigidly linked and the motion must be planned for the composed body.

In the following, we consider that the robot may grasp the object when it touches it (i.e. when the robot and the object are in contact). Moreover, we consider the case where both robot and object move in translation in the plane[2].

2.2 The geometrical structure of admissible paths

Let $CSR = \mathbf{R}^2$ and $CSO = \mathbf{R}^2$ be the configuration spaces of the robot and the object, and let $CS = CSO \times CSR$ be the configuration space of the complete system. A configuration $c \in CS$ is the pair $c = (co, cr)$ where co and cr are the position coordinates of the object and the robot.

We call ACS the subset in CS of all admissible configurations, i.e. configurations where the bodies do not overlap. Let $ACSR$ be the collision-free configuration space of the robot without considering the object ($ACSR \subset CSR$). Similarily, we define $ACSO \subset CSO$ as the set of admissible configurations of the object without the robot. See figure 2. Note that ACS is not the set of $(co, cr) \in CS$ such that $co \in ACSO$ and $cr \in ACSR$; all configuration where object and robot collide must be excluded.

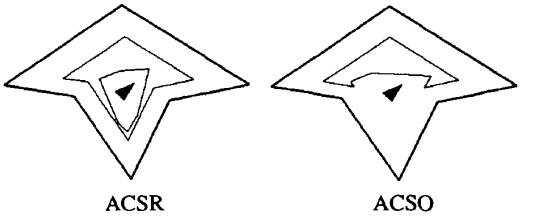

Figure 2: Admissible configurations subsets of the illustration example.

The solution of the coordinated motion problem relies in finding a path in the connected components of ACS. Any path in ACS corresponds to coordinated motion where the robot and the object move independently (see Figure 1B).

However, such a path does not necessarily verify the specific constraints of a manipulation problem. These constraints can be formulated as follows. The only valid paths in ACS are :

Transit paths along which the robot moves alone and the object remains at the same place.

Transfer paths corresponding to motions of the robot and the object together, rigidly linked in a contact relative configuration. Let $c = (co, cr)$ be a configuration where the robot and the object are in contact : we define $co - cr$ as the relative grasp configuration[3]. Along a transfer path, the relative grasp configuration is constant.

We call **manipulation path** a sequence of transit and transfer paths. Solving a manipulation problem consists then in finding a manipulation path.

Figure 1 gives an illustration of this. The initial and goal configurations are given in Figure 1A. Figure 1B illustrates the coordinated motion, where the robot and the object move independantly. The associated manipulation problem requires a transit path to reach the object, a transfer path to bring it to the center of the environment, a second transit path to regrasp it on the other side in order to move it to its final place with a last transfer path, and finally a transit path to make the robot reach its goal position (see Figure 1C)[4].

We determine now the topological properties of transit and transfer paths in ACS. In fact, they take place in specific submanifolds of CS.

[2]Nevertheless, the results presented in this section hold in a more general statement (see [8]).

[3]Note that co, cr and $co - cr$ are in $\mathbf{R}^2$.

[4]Note that the two transfer stages require several regrasps. This point is discussed below with the reduction property.

Let $ACSR(co)$ be the set of all admissible configurations of the robot when the object is placed in co. All transit paths, when the object position is co, belong to the connected components of $ACSR(co)$.

Changing co requires to "grasp" the object in order to move it. Let $GRASP$ be the set of all grasp configurations. $GRASP$ is a subset of ACS boundary, corresponding to all contacts between the robot and the object. We assume that the robot avoids all the contacts with fixed obstacles : this means that cr stays in an open set of $ACSR$ (hypothesis H). We define $GRASP_H$ as the subset of $GRASP$ of all configurations verifying hypothesis H. In $GRASP_H$, the robot touches the object but none of the obstacles[5].

Any transfer path takes place in $GRASP_H$. However, any path in $GRASP_H$ is not necessarily a transfer path : it may not keep the very same grasp of the robot on the object, but may perform a continuous change of the grasp. Basically, transfer paths are constrained to stay in 2-dimensional submanifolds of $\mathbf{R}^4$ defined by a constant grasp $g = co - cr$. We prove in the next section that any path in $GRASP$ is equivalent to a finite sequence of transit and transfer paths, i.e. a manipulation path.

2.3 Reduction property

Property 1 *Two configurations of a same connected component of $GRASP_H$ are connected by a manipulation path.*

Proof : Let a and b be two configurations in a connected component of $GRASP_H$. There exists a path $p : [0,1] \mapsto GRASP_H$ linking these two configurations[6] ($p(0) = a$, $p(1) = b$). We define p_r and p_o as the projections of p onto CSR and CSO respectively.

Let $c = p(t)$ be any configuration on the path. Thanks to the hypothesis H, $p_r(t)$ lies in an open set of $ACSR$. We then can find an open disc $D_\epsilon \subset ACSR$ centered on $p_r(t)$ and with a radius $\epsilon > 0$.

Since p is continuous, there exists $\eta_1 > 0$ such that :

$$\forall \tau \in]t - \eta_1, t + \eta_1[, p_r(\tau) \in D_{\epsilon/2}.$$

Similarily, $p_r - p_o$ is a continuous function. Then there exists $\eta_2 > 0$ such that :

$$\forall \tau \in]t - \eta_2, t + \eta_2[,$$
$$\| (p_r(\tau) - p_o(\tau)) - (p_r(t) - p_o(t)) \|_{\mathbf{R}^2} < \epsilon/2.$$

This last assertion means that the relative grasp configuration does not vary more than $\epsilon/2$ along the path p between $p(t - \eta_2)$ and $p(t + \eta_2)$.

Let us consider $\eta = min\{\eta_1, \eta_2\}$:

$$\forall \tau, \sigma \in]t-\eta, t+\eta[, \quad p_o(\sigma)+(p_r(\tau)-p_o(\tau)) \in D_\epsilon. \quad (1)$$

Let $c_1 = p(\tau_1)$ and $c_2 = p(\tau_2)$ be any two configurations on p, with τ_1 and τ_2 in $]t-\eta, t+\eta[$ (we assume that $\tau_1 < \tau_2$). We prove now that c_1 and c_2 can be linked by one transfer path followed by one transit path.

Let us consider the path $(p_o(\tau), p_o(\tau) + (p_r(\tau_1) - p_o(\tau_1)))$, with $\tau \in [\tau_1, \tau_2]$. This path is clearly a transfer path with constant grasp $(p_r(\tau_1) - p_o(\tau_1))$, between $p(\tau_1)$ and $(p_o(\tau_2), p_o(\tau_2) + (p_r(\tau_1) - p_o(\tau_1)))$. According to relation 1, this path is admissible. Let us consider the path $(p_o(\tau_2), p_o(\tau_2) + (p_r(\tau) - p_o(\tau)))$, with $\tau \in [\tau_1, \tau_2]$. This path is clearly a transit path between $(p_o(\tau_2), p_o(\tau_2) + (p_r(\tau_1) - p_o(\tau_1)))$ and $p(\tau_2)$. Again, according to relation 1, this path is admissible. The concatenation of both paths constitutes a manipulation between $p(\tau_1)$ and $p(\tau_2)$.

As path p_r is a compact set included in an open set of ACSR, we can apply this local transformation on a *finite* covering of $[0, 1]$. We have then a finite number of elementary manipulation paths[7] which constitutes a manipulation path linking a and b. □

Note that the reduction property does not hold when hypothesis H is not satisfied. Figure 3 illustrates this fact. From now on, and in order to keep simple notations, we will consider $GRASP$ as being $GRASP_H$.

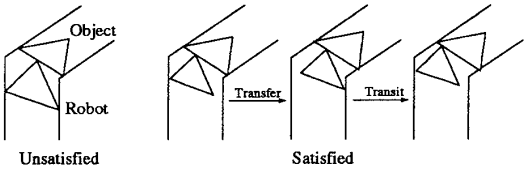

Figure 3: An illustration of hypothesis H.

2.4 Computational consequences

Let $\mathcal{C}$ be the set of connected components of $GRASP$. Let us consider the graph whose nodes are the elements of $\mathcal{C}$ and whose edges correspond to the existence of a transit path between two configurations of the associated nodes. Thanks to the property above, two configurations in $GRASP$ are connected by a manipulation path if and only if they belong to two elements of $\mathcal{C}$ which are in the same connected component of the graph.

Therefore, in order to solve the manipulation problem, we have to :

1. compute the connected components of $GRASP$,
2. and link them by transit paths.

A general method consists in using a cell decomposition of $GRASP$ (see [7, 8]). In next section, we detail a specific solution for the case of two translating polygons.

[5] Note that $GRASP_H$ is obtained by excluding from $GRASP$ only some of its frontiers.

[6] p designates a path as well as the associated function.

[7] It is clear that this number varies as a linear function of ACSR size.

3 The case of a convex polygonal robot and a convex polygonal object moving in translation amidst polygonal obstacles

This section describes a method for solving the manipulation problem in the case of a polygonal robot and a polygonal object moving in translation amidst polygonal obstacles. It has been implemented in C, in the case where both polygons are convex. In order to illustrate the different steps, we will rely on the simple example of Figure 1.

According to the resolution scheme stated in Section 2 :

- The exploration of $GRASP$ is provided by a cell decomposition : we first compute a cell decomposition of ACS (that solves the coordinated motion problem); then a retraction on the boundary of ACS gives a cell decomposition of $GRASP$.
- The connectivity of $GRASP$ cells by transit paths is given by the various connected components of $ACSR(co)$, whose structure can be extracted from ACS cells.

3.1 ACS cell decomposition and coordinated motions

To compute the cell decomposition of ASC, we have choosen to use an adaptation of the projection method developed by Schwartz and Sharir in [10] for the case of two discs[8].

Let us consider an object position co. $ACSR(co)$ is obtained by removing, from $ACSR$ (robot admissible configurations without the object), the set $COL(co)$ of all configurations where the robot and the object collide [9] (see figure 4).

Let us observe now the evolution of $ACSR(co)$ with respect to co. In a neighbourhood of most object positions, $ACSR(co)$ keeps the same structure. However, at some object positions, some $ACSR(co)$ connected components may appear, disappear, be split or merged. More precisely, the geometrical structure of the connected components of $ACSR(co)$ are modified when some vertices appear or disappear. These changes correspond to specific values of co which constitute a set of *critical curves* (see [10] for a proof). The figure 5 gives an example of a critical curve along which a connected component of $ACSR(co)$ is divided into two separate components. The critical curves provide a decomposition of $ACSO$ into *non-critical regions* (Figure 6).

Let us consider a non-critical region R. $\{\{co\} \times ACSR(co) \mid co \in R\}$ constitutes cells of ACS (there are as many cells as the number of connected components of $ACSR(co)$). The set of all such cells is the expected cell decomposition.

[8] Any other, and perhaps better ([5, 9, 11]) method could have been used. However, the purpose of this paper is not to give some optimal algorithm, but to prove the feasibility of our approach.

[9] $COL(co)$ is the polygon obtained by Minkowski difference between the robot and the object, and placed in co.

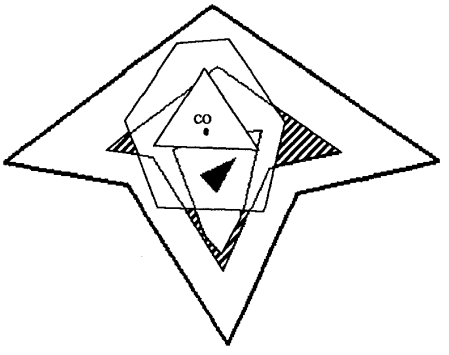

Figure 4: The hatched areas represent the connected components of $ACSR(co)$.

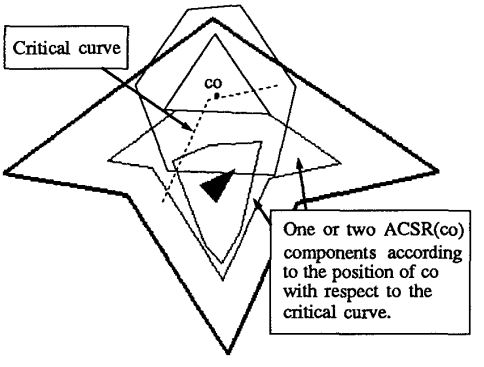

Figure 5: A critical curve

In order to compute the critical curves, we introduce a symbolic description of $ACSR(co)$. Let us recall that the boundary of $ACSR(co)$ is constituted by $ACSR$ edges and $COL(co)$ edges. We put a numerical label on all the vertices of $ACSR$ and a letter on the edges of $COL(co)$. We denote by $b[8,9]$ the intersection between the edge b of $COL(co)$ and the segment $[8,9]$ of $ACSR$. Therefore, the bottom connected component in the example of Figure 7 is labeled by the sequence $(3, 4, [4,5]b, b[8,9], 9, [9,10]b, b[1,2], 2)$.

To compute the critical curves, we do not consider all the possible changes in this sequence. We just need to consider the changes *on the letters* (i.e. the changes induced by $COL(co)$ and not by $ACSR$ vertices). For instance, in Figure 7, when co moves to bottom, the disappearance of vertex 2 in the sequence above does not induce a critical curve, while the fusion of the two b labels (when edge b meets vertex 3) does.

Due to space constraints, we do not give details here on the computation of the critical curves (see [3] for details). The critical curves of our example are shown

in Figure 6, together with the graph of non-critical regions of $ACSO$.

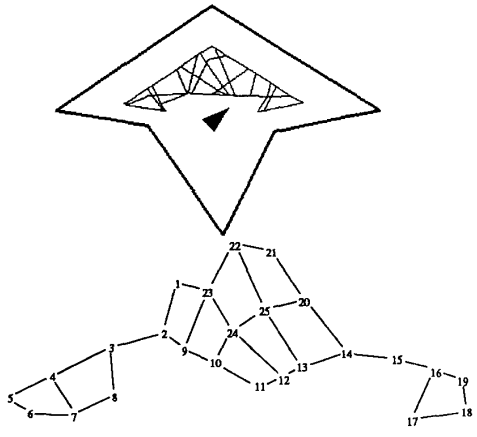

Figure 6: Non-critical regions and the associated graph.

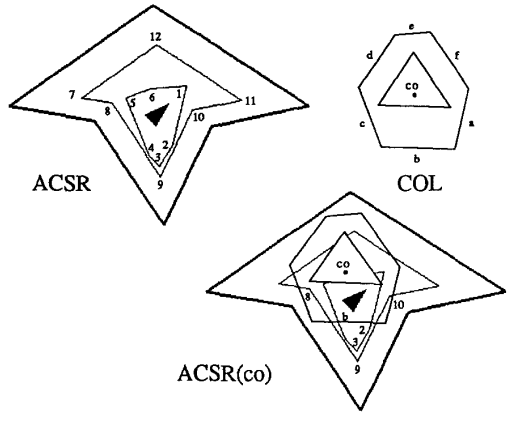

Figure 7: An illustration of the labels used to characterize the connected components of ACSR(co).

Let us recall that each non-critical region induces as many cells in ACS as the number of connected components in $ACSR(co)$ when co belong to the region. Now we structure all these cells into a *coordinated motion graph*. Two cells are adjacent in this graph if and only if :

- the associated non-critical regions are adjacent in $ACSO$, and
- the symbolic descriptions of the associated $ACSR(co)$ components just differ by a letter.

Figure 8 shows the coordinated motion graph of our example. The position co in Figure 7 belong to the non-critical region 10 (Figure 6). This region gives rise to three ACS cells numbered $10A$, $10B$ and $10C$ in Figure 8. $10C$ is the node corresponding to the label mentionned above.

The fundamental property of the graph is : *there exists a coordinated motion between two configurations in ACS if and only if they belong to nodes of a same connected component of the coordinated motion graph.* The proof is exactly the same as in [10].

3.2 $GRASP$ cell decomposition and contact motion

Now we show how to use the previous structure in order to provide a cell decomposition of $GRASP$.

Let us consider the above $ACSR(co)$ component labeled by $(3, 4, [4, 5]b, b[8, 9], 9, [9, 10]b, b[1, 2], 2)$ (Figure 7). There are two edges labeled by b in it. This means that there are two connected sets of configurations where the robot is in contact with the object. It is not possible to go from one set to the other one without leaving the contact. These connected sets are easily extractable from the symbolic description of the $ACSR(co)$ components. In our example, $([4, 5]b, b[8, 9])$ and $([9, 10]b, b[1, 2])$ are the symbolical descriptions of these two contact classes. By definition, they always contain two terms.

Now, let us consider a non-critical region R. $\{\{co\} \times COL(co) \cap ACSR(co) \mid co \in R\}$ constitutes cells of $GRASP$.

We follow the same method as for the coordinated motion problem. We structure the $GRASP$ cells into a *contact graph*. Two cells are adjacent in this graph if and only if :

- the associated non-critical regions are adjacent in $ACSO$, and
- their symbolic descriptions just differ by one term.

Figure 9 shows the contact graph of our example. Region 10 gives rise to three ACS cells. The frontier of two of them ($10A$ and $10B$) contain one connected component in $GRASP$. They then give rise to two $GRASP$ cells (which keep the same name in the contact graph). The frontier of $10C$ contains two connected components in $GRASP$; they give rise to $GRASP$ cells $10CA$ and $10CB$.

This graph verifies the following property : *there exists a motion keeping the contact between the robot and the object, between two configurations in GRASP, if and only if both configurations belong to nodes of a same connected component of the contact graph.* The proof is exactly of the same kind as for the coordinated motion graph.

3.3 The manipulation graph

Let us come back to our manipulation problem. At this time we have captured the connectivity of $GRASP$. We know that two configurations in the same connected component of $GRASP$ may be linked by a manipulation path (Reduction Property).

Now we have to study the existence of transit paths between $GRASP$ components. This study is very easy from the above labeling. Indeed let us consider $ACSR(co)$ in Figure 7. There are two grasp classes in the bottom component. Nevertheless, it is possible for the robot to move *alone* in this component; that means that the robot can go from a position in the first grasp class to any other one in the second grasp class. Then, these two classes are linked by transit paths.

The existence of such transit paths is very easy to compute from the labeling of $ACSR(co)$. Indeed, two $GRASP$ cells are connected by a transit path if and only if they belong to the frontier of the same ACS cell. In our current example, the solely cells $10CA$ and $10CB$ (which come from the same ACS cell $10C$) are linkable by a transit path.

Compute the connectivity of $GRASP$ components by transit path is just equivalent to add to the contact graph edges between nodes defined from a same ACS cell. These additonal edges are refered as "transit edges" in Figure 10. With our notations, two nodes in the contact graph whose "names" contain the same number and the same first letter are linked by a transit edge. The consecutive graph is called the *manipulation graph.*

Putting together all the previous results we may conclude that :

Property 2 *Two configurations in GRASP are connected by a manipulation path if and only if they belong to cells which appear in a same connected component in the manipulation graph.*

3.4 Manipulation path finding

Now, we show how to use the manipulation graph in order to find a manipulation path. Let us consider an initial configuration c_i and a final one c_f, defining the initial and final positions of the robot and the object in the environment. According to the property of the manipulation graph, we use a three-steps procedure :

1. First, we compute the ACS cells C_i and C_f containing c_i and c_f. Then, we compute the set $\mathcal{G}_i$ (resp. $\mathcal{G}_f$) of $GRASP$ cells reachable from c_i (resp. c_f). This computation is a 2-dimensional problem since the transit paths have to lie in $ACSR(co_i)$ (resp. $ACSR(co_f)$.

2. The second step consists in searching a path in the manipulation graph between a cell in $\mathcal{G}_i$ and a cell in $\mathcal{G}_f$. If no such path exists, the procedure stops. There is no solution. Otherwise, we obtain a sequence $\mathcal{P}ath$ of $GRASP$ cells.

3. Finally the complete path is built from elementary manipulation paths lying in the $GRASP$ cells of $\mathcal{P}ath$ and from transit paths associated to the transit edges contained in $\mathcal{P}ath$.

Comments on Step 1 : The computation of C_i (resp. C_f) is a 2-dimensional location problem performed in $ACSR(co_i)$ (resp. $ACSR(co_f)$) : we have to determine the connected component of $ACSR(co_i)$ (resp. $ACSR(co_f)$ containing cr_i (resp. cr_f). This fully characterizes C_i (resp. C_f). Then the computation of $\mathcal{G}_i$ (resp. $\mathcal{G}_f$) is very easy, since these $GRASP$ cells belong to the frontier of C_i (resp. C_f) : with our graph node notations, a $GRASP$ cell belonging to the frontier of some ACS cell appear in the contact graph with the same numerical label and the same first letter as the ACS cell appears in the coordinated motion graph.

Comments on Step 2 : The second step is performed by a A^* algorithm. Several cost criteria can be introduced : the length of the complete path, the number of grasping changes...

Comments on Step 3 : Step 3 consists in computing the complete manipulation path. $\mathcal{P}ath$ is a sequence of $GRASP$ cells. Two consecutive cells in this sequence are linked either by an edge already appearing in the contact graph, or by a transit edge. Moving inside a $GRASP$ cell needs a specific procedure : we have implemented the method presented in the proof of the reduction property (that kind of elementary manipulation path give rise to numerous –but finite– grasping changes). Finally, there remains to compute the transit path associated to a transit edge : this is a 2-dimensional problem solved in some $ACSR(co)$ slice by a visibility graph method for instance (this is the method we have implemented).

3.5 Complexity

The complexity of the manipulation graph computation is clearly dominated by the construction of the non-critical cells. [4] shows that the graph is built in $O(n^3 \log n)$. In addition, the complexity of the manipulation path depends on the size of ACSR (see 2.3).

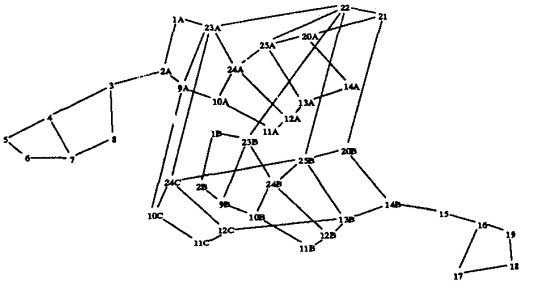

Figure 8: The coordinated motions graph.

4 Experimental results

Figure 1C shows the solution given by the planner to our illustration example. The initial and final objects positions are respectively in the non-critical regions 6 and 19 (Figure 6). The initial and final configurations of the manipulation problem are respectively in the ACS cells 5 and 18 (Figure 8).

Let us consider now the solution path (5_B, 4_B, 3_B, $2AB$, $9AB$, $23AB$, 22_B, 22_A, $25BA$, $13BA$, $14BA$, 15_A, 16_A, 19_A, 18_A). The first figure (top, left) in Figure 1C shows the transit path along which

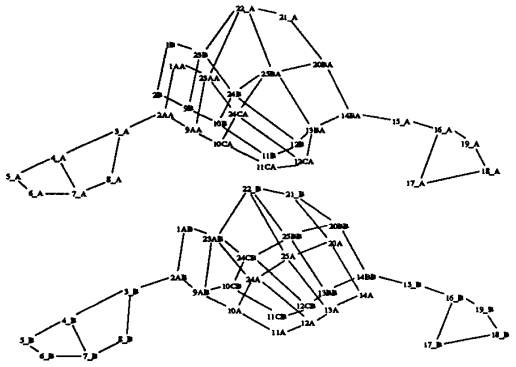

Figure 9: The contact graph.

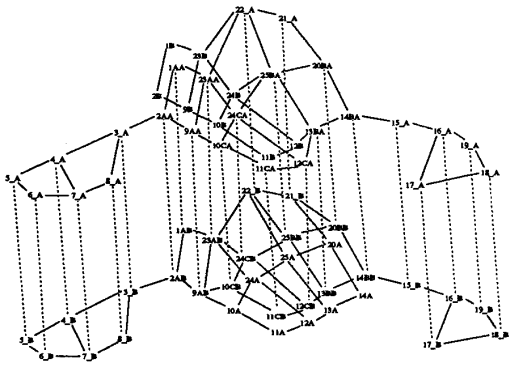

Figure 10: The manipulation graph, where dotted lines describe transit edges.

the robot first reaches the object. In the second one (top, middle), the object is moved with a sequence of transit and transfer paths along the same connected component of $GRASP$. If we refer to the manipulation graph (Figure 10), the manipulation path is built, at this time, from the sequence (5_B, 4_B, 3_B, $2AB$, $9AB$, $23AB$, 22_B) of $GRASP$ cells. A transit edge appears between vertices 22_B and 22_A; the associated transit path is shown in the third figure (top, right). The fourth one (bottom, left) describes the subsequence (22_A, $25BA$, $13BA$, $14BA$, 15_A, 16_A, 19_A, 18_A). A last transit path allows then to reach the robot goal configuration.

Many improvements of the path planner could still be done. Path planning inside each cell, which is performed with visibility graph for the moment, may be performed using other methods, like Voronoï diagrams for example, as an exact description of the free space is available at each step of the motion.

The algortihm has been implemented in C on a Sun SPARC Station 1. The computation of the manipulation graph of Figure 10 (including the computation of the non-critical regions) takes 9 seconds. Search for a path takes 0.45 seconds for the example in Figure 1.

References

[1] R. Alami, T. Siméon, J.P. Laumond, "A geometrical Approach to planning Manipulation Tasks. The case of discrete placements and grasps." International Symposium on Robotics Research, Tokyo, August 1989.

[2] P.C. Chen, Y.K. Hwang, "Practical path planning among movable obstacles." in *IEEE* Conf. on Robotics and Automation, 1991.

[3] B. Dacre Wright, "Une nouvelle approche de la planification des taches de manipulation: le cas d'un robot et d'un obstacle mobile polygonaux, en translation dans un environnement polygonal." Technical Report, ENSTelecom, Toulouse, Juillet 1989.

[4] B. Dacre Wright, J.P. Laumond, R. Alami, "Motion planning for a robot and a movable object amidst polygonal obstacles." Technical Report LAAS/CNRS 91421, Toulouse, September 1991.

[5] S. Fortune, G. Wilfong, C. Yap, "Coordinated motion of two robot arms." in *IEEE* Conf. on Robotics and Automation, 1986.

[6] J.-C. Latombe, *Robot Motion Planning*, Kluwer Press, 1991.

[7] J.P. Laumond, R. Alami, "A new geometrical approach to manipulation task planning: the case of a circular robot amidst polygonal obstacles and a movable circular object." Technical Report LAAS/CNRS 88314, Toulouse, Octobre 1988.

[8] J.P. Laumond, R. Alami, "A geometrical approach to planning manipulation tasks in Robotics.", Technical Report LAAS/CNRS 89261, Toulouse, August 1989.

[9] D. Parsons, J. Canny, "A motion planner for multiple mobile robots." in *IEEE* Conf. on Robotics and Automation, 1990.

[10] J.T. Schwartz, M. Sharir, "On the piano mover problem III : Coordinating the motion of several independant bodies : the special case of circular bodies amidst polygonal barriers." Int. J. of Robotics Research, 2 (3), 1983.

[11] M. Sharir, S. Sifrony, "Coordinated motion planning for two independent robots." ACM Symposium on Computational Geometry, 1988.

[12] G. Wilfong, "Motion planning in the presence of movable obstacles." in *ACM* Symp. on Computational Geometry, 1988.