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

Experiments in Dual-Arm Manipulation Planning

Yoshihito Koga and Jean-Claude Latombe
Robotics Laboratory
Department of Computer Science, Stanford University
Stanford, CA 94305, USA

Abstract

The goal of dual-arm manipulation planning is to plan the path of two cooperating robot arms in order to carry a movable object from a given initial configuration to a given goal configuration amidst obstacles. This problem typically occurs in tasks requiring the manipulation of long and/or heavy objects. It differs from the classical path planning problem in that it involves grasp and ungrasp operations that dynamically change both the kinematic structure of the total robotic system and the constraints imposed on the arms' motion by the obstacles. We describe three implemented planners that tackle increasingly difficult versions of this problem.

Acknowledgments: This research was funded by DARPA contract DAAA21-89-C0002 (Army) and a grant of the Stanford Center for Integrated Facility Engineering. Y. Koga is supported in part by a Canadian NSERC fellowship.

1 Introduction

We consider a new motion planning problem, which we call *dual-arm manipulation planning.* The goal is to plan the motion of two cooperating robot arms to carry a movable object from a given initial configuration to a given goal configuration amidst fixed obstacles. Each arm can move separately, but *both* arms have to grasp the object in order to move it. Avoiding collision between the two arms, or between one arm and an obstacle, or between the movable object and either an arm or an obstacle may require the arms to change their grasp of the object. This problem occurred to us as we were working on a space robotics project aimed at assembling truss structures [7]. It also arose in a construction robotics project involving the manipulation of long and heavy objects (e.g., pipes, beams).

To illustrate our problem, consider Fig. 1. It shows snapshots of a manipulation path produced by one of the planners described below. The movable object is a long bar AB (shown as a bold line) that can be moved in the plane by two identical arms, each with 3 revolute joints. The arms can grasp the object by positioning their endpoints at the extremities of the bar (then the grasp contacts behave as passive revolute joints). There is a single, rectangular obstacle. The initial and goal configurations of the bar are shown in Fig. 1 (a) and (l), respectively, with the corresponding configurations of the arms. The problem is to plan a path for the two arms to transport the bar from its initial configuration to its goal. Fig. 1 displays such a path. The two arms first grasp the bar as shown in (b). A collision with the obstacle (c) yields the right arm to ungrasp the bar (d), move around the obstacle (e), and then regrasp the bar at a new configuration (f). The motion of the bar is resumed (g) until a collision between the two arms (h) requires them to swap their grasp positions (i). The motion of the bar is resumed again (j) and the goal configuration of the bar is achieved (k). The two arms then ungrasp the bar and move to their final configurations (l).

Dual-arm manipulation planning differs from classical path planning in that it involves grasp and ungrasp operations that dynamically change (1) the kinematic structure of the total robotic system, and (2) the constraints imposed on the arms' motion by the obstacles. Consider the composite configuration space of the two arms and the movable object. In this space, a manipulation path is a sequence of subpaths separated by ungrasp/regrasp operations. Every subpath lies in a different, lower-dimensional submanifold of the composite configuration space and the transfer between two subpaths occurs at a configuration of the total system contained in the intersection of the corresponding submanifolds. For every subpath, the problem reduces to path planning for some fixed kinematic structure and obstacle constraints. For example, in Fig. 1, between snapshots (c) and (f), the planned path is that of a single 3-revolute-joint arm (the right arm) in the presence of multiple obstacles (the environment obstacle, the left

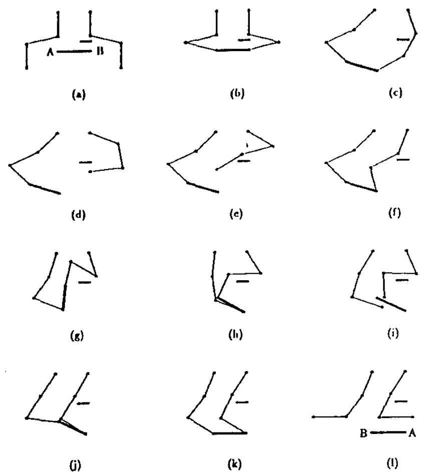

Figure 1: A Dual-Arm Manipulation Path

arm, and the bar). Between (f) and (h), the path is that of a 5-degree-of-freedom closed-loop kinematic chain in the presence of a single obstacle.

In Section 2 we survey previous work in manipulation planning and we relate it to our own work. In Section 3 we give a formal configuration space presentation of the problem addressed in this paper. In Sections 4 through 6 we describe three implemented planners, Planners 1, 2, and 3. These three planners operate in a two-dimensional workspace and tackle increasingly difficult versions of the dual-arm manipulation problem. They are implemented in the C language. Execution times given in the paper were obtained by running them on a DEC 5000 workstation (28-mips processor).

2 Related Work

Research in motion planning has been very active over the past decade [10, 11, 3]. Most of it has addressed the problem of planning a collision-free path for a mechanical system with a fixed kinematic structure. The interest in manipulation planning is relatively recent [12, 4, 1, 5].

The problem of planning the path of a convex polygonal robot translating in a two-dimensional polygonal workspace in the presence of multiple convex polygonal movable objects is addressed in [12]. The robot can "grasp" a movable object by making one of its edges coincide with an edge of the object; then the robot and the object move together as a single rigid object until the robot ungrasps the movable object. In the particular case where there is a single movable object, a complete planning algorithm is given to find a path between an initial configuration of the robot and the object and a goal configuration.

A similar problem is considered in [4] where the robot and the movable objects are discs in a polygonal environment. The robot can grasp the object by making contact with it. While the robot is grasping the object, they both move together as a single rigid object. A complete algorithm based on an exact cell decomposition method previously proposed in [9] is described to plan the coordinated paths of two independent circular robots.

An implemented algorithm for one robot and several moving objects is described in [1]. Both the number of grasps of each object (positions of the robot relative to the object) and the number of placements of each object in the workspace are finite. In [5] the problem is generalized to the case where the set of placements of the movable objects and the sets of grasp positions on these objects are described algebraically. Using an adaptation of the general planning algorithm given in [8], it is shown that the problem of finding a manipulation path is decidable, but the time complexity of the algorithm is very high.

Our work differs from this research in several ways. Previous work has considered the case of a single robot grasping one object at a time. We explicitly address cooperative manipulation with two arms, a problem which naturally yields changes in the kinematic structure of the robotic system. In addition, dealing with multiple arms also increases the number of degrees of freedom that can move at a single time. In one of the implemented planners, this led us to use randomized planning techniques introduced in [2]. Finally, our research has been mainly experimental in nature. Rather than just investigating the problem theoretically, and show that it is provably complex (we know that!), we have implemented successive planners and experimented with them in order to acquire the empirical understanding of the problem that will ultimately be needed to produce a practical planner.

3 Problem Statement

We now give a more formal statement of the dual-arm manipulation problem. For simplicity, we restrict this statement to cover only those problems that are pertinent to our planners. However, it is not difficult to extend this formulation to include more complicated manipulation problems.

We consider a two-dimensional workspace $\mathcal{W}$ with two arms $\mathcal{A}_1$ and $\mathcal{A}_2$, a movable object $\mathcal{M}$ and fixed obstacles $\mathcal{B}_i$, $i = 1, \cdots, q$.

Let $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_{obj}$ be the configuration spaces of the two arms $\mathcal{A}_1$ and $\mathcal{A}_2$ and the object $\mathcal{M}$, respectively [6, 3]. The *composite configuration space* of the whole system is $\mathcal{C} = \mathcal{C}_1 \times \mathcal{C}_2 \times \mathcal{C}_{obj}$. Hence, a configuration $\mathbf{q} \in \mathcal{C}$ is of the form $(\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_{obj})$, with $\mathbf{q}_1 \in \mathcal{C}_1$, $\mathbf{q}_2 \in \mathcal{C}_2$, and $\mathbf{q}_{obj} \in \mathcal{C}_{obj}$. Let m_1, m_2, and m_{obj} be the respective dimensions of the manifolds $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_{obj}$ (in a two-dimensional workspace, as is the case here, we have $m_{obj} = 3$). The dimension of $\mathcal{C}$ is $m = m_1 + m_2 + m_{obj}$. In the example of Fig. 1, $m_1 = m_2 = 3$.

We define the *C-obstacle region* $\mathcal{CB} \subset \mathcal{C}$ as the set of all configurations $\mathbf{q} \in \mathcal{C}$ where two or more bodies in $\{\mathcal{A}_1, \mathcal{A}_2, \mathcal{M}, \mathcal{B}_1, \cdots, \mathcal{B}_q\}$ intersect.[1] We assume that all bodies are described as closed subsets of $\mathcal{W}$ so that $\mathcal{CB}$ is a closed subset of $\mathcal{C}$. Let us denote the open subset $\mathcal{C} \setminus \mathcal{CB}$ by $\mathcal{C}_{free}$ and the closure of this subset by $cl(\mathcal{C}_{free})$.

Given the geometry of all the objects in $\mathcal{W}$ and the location of the obstacles, a manipulation problem is specified by an initial configuration $\mathbf{q}^i = (\mathbf{q}_1^i, \mathbf{q}_2^i, \mathbf{q}_{obj}^i) \in \mathcal{C}_{free}$ and a goal configuration $\mathbf{q}^g = (\mathbf{q}_1^g, \mathbf{q}_2^g, \mathbf{q}_{obj}^g) \in \mathcal{C}_{free}$. A solution to the problem is a path $\tau : [0,1] \rightarrow cl(\mathcal{C}_{free})$ connecting these two configurations. We call τ a *dual-arm manipulation path.*

However, not all paths τ in $cl(\mathcal{C}_{free})$ are valid solutions of the manipulation planning problem. Valid paths are further constrained by the fact that the movable object cannot move by itself; the two arms must simultaneously grasp the object in order to move it. We define an *effector point* E_i on each arm $\mathcal{A}_i$, $i = 1, 2$. In all our examples, E_i will be the endpoint of $\mathcal{A}_i$. We also define two *grasp points* G_1 and G_2 on $\mathcal{M}$. The two arms can grasp $\mathcal{M}$ by positioning E_1 onto G_1 and E_2 onto G_2, or E_1 onto G_2 and E_2 onto G_1. (The case where the two effector points are located at the same grasp point is not allowed.) We call *grasp contact* any pair of the form (E_i, G_j), $i, j \in \{1, 2\}$, and *grasp pairing* each of the two pairings $\{(E_1, G_1), (E_2, G_2)\}$ and $\{(E_1, G_2), (E_2, G_1)\}$. We assume that the connection (E_i, G_j) behaves as a passive revolute joint. Hence, when the two arms hold the movable object, they form with this object a closed loop chain with $m_1 + m_2 + 2$ joints.

A dual-arm manipulation path τ is the concatenation of several subpaths τ_k $(k = 1, 2, \cdots)$. Each subpath is:

(1) Either a *one-arm transit path.* One arm $\mathcal{A}_i$ moves as an open-loop chain from one grasp contact to the same grasp contact, while the other arm $\mathcal{A}_j$ and the object $\mathcal{M}$ are stationary with $\mathcal{A}_j$ grasping $\mathcal{M}$. This path lies in a cross-section of $\mathcal{C}$ defined by the current fixed configurations of $\mathcal{A}_j$ and $\mathcal{M}$. (This cross-section is a copy of $\mathcal{C}_i$ in which the obstacle region depends on the current configurations of $\mathcal{A}_j$ and $\mathcal{M}$.)

(2) Or a *two-arm transit path.* Neither of the two arms hold the movable object, and both move as open-loop kinematic chains. Hence, the movable object is stationary. This path lies in a cross-section of $\mathcal{C}$ defined by the current fixed configuration of $\mathcal{M}$. (This cross-section is a copy of $\mathcal{C}_1 \times \mathcal{C}_2$ in which the obstacle region depends on the current configuration of $\mathcal{M}$.)

(3) Or a *transfer path.* The two arms hold the movable object and moves as a closed-loop kinematic chain. This path lies in a subset of $\mathcal{C}$ of dimension $m_1 + m_2 - 1$.

Collision-avoidance constraints are defined as follows:

- During a one-arm transit path, the moving arm should touch no obstacle, nor should it touch the stationary arm or the movable object, except at the two ends of the path where the arm's effector point touches the movable object at a grasp point.
- During a two-arm transit path, the two arms should touch no obstacle, nor should they touch each other or the movable object, except at the two ends of the path where the arms' effector points touch the movable object at the grasp points.
- During a transfer path, the two arms and the moving object should touch no obstacle, nor they should touch each other, except for the contact at the grasp points.

A one-arm transit path moves an arm from a grasp contact to the same grasp contact. Its purpose is to avoid collision of the closed-loop chain with an obstacle (or possibly the joint limit of one of the two arms) by changing the homotopic position of the closed-loop chain relative to one or several obstacles. A two-arm transit path may have the same purpose, but more often it is to swap grasps (i.e., change the grasp pairing) in order to avoid collision between the two arms, or between an arm and the movable object.

The dual-arm manipulation path τ is an alternation of transit and transfer subpaths. Hence, $\tau = \tau_1 \bullet \tau_2 \bullet \cdots \bullet \tau_{2p+1}$, with $p \geq 0$ and "$\bullet$" denoting the concatenation of two subpaths. Each subpath τ_k, for an odd value of k, is a one- or two-arm transit path (for $k = 1$ and $k = 2p+1$, it is a two-arm transit path). Each subpath τ_k, for an even value of k, is a transfer path.

We denote by $\mathcal{C}_{grasp} \subset cl(\mathcal{C}_{free})$ the set of collision-free configurations achieving one of the two grasp pairings, i.e. the space in which all transfer paths lie. It is a submanifold of $\mathcal{C}$ of dimension $m_1 + m_2 - 1$. We call *grasp configuration* to be any configuration in $\mathcal{C}_{grasp}$.

4 Planner 1

We now describe Planner 1, which solves a simplified version of the dual-arm manipulation problem. Fig. 2

[1] We regard joint limits in $\mathcal{A}_1$ and $\mathcal{A}_2$ as obstacles that only constrain the arms' motion.

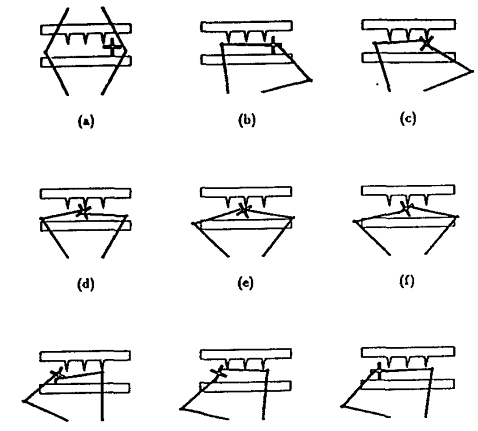

Figure 2: Path Generated by Planner 1

shows a path generated by this planner.

The first simplifying assumption is that the two arms can only collide with each other. This is physically achieved by having the two arms move in a horizontal plane above the plane containing the movable object and the obstacles. Grasping consists of inserting pins located at the arms' effector points into holes located at the grasp points of the movable object. Hence, the only possible collisions are either between the two arms or between the movable object and the obstacles.

The second assumption is that any two collision-free configurations of the arms are connected by a collision-free path in their configuration space $\mathcal{C}_1 \times \mathcal{C}_2$. In our implementation, this assumption is enforced by having two identical arms, each with two revolute joints not limited by any mechanical stops At any configuration of the movable object, there are up to eight configurations of the two arms where they can grasp the object. Indeed, there are two possible grasp pairings and for each one, each arm may take two configurations.

These assumptions have three major consequences:

(1) The only useful transit paths are to avoid inter-arm collision or to deal with arm singularities.

(2) At any configuration of the movable object, there is a collision-free path of the two arms connecting any two grasp configurations.

(3) The grasp configurations can be symbolically characterized by the grasp pairing and the posture ("elbow in" or "elbow out") of each arm. We call this characterization the ***type*** of the grasp configuration.

Planner 1 works as follows. It computes a collision-free path of the movable object $\mathcal{M}$ among the obstacles, assuming that the movable object can translate and rotate freely by itself. The planner uses the best-first search potential field method described in [2, 3]. It discretizes the three-dimensional configuration space $\mathcal{C}_{obj}$ into a fine regular grid. A potential field with a global minimum at the goal configuration $\mathbf{q}^g_{obj}$ of $\mathcal{M}$ is defined over this grid (see [2, 3] for a definition of this potential). The path of $\mathcal{M}$ is constructed by exploring the configuration space grid in a best-first fashion using the potential as the cost function and starting at the initial configuration $\mathbf{q}^i_{obj}$ of $\mathcal{M}$. The best-first search escapes local minima, if any, by filling them up, which is reasonable in a three-dimensional grid. The outcome of the search, if successful, is a path τ_{obj} of $\mathcal{M}$ described as a sequence of grid configurations.

The search algorithm takes the arms into account as follows. Using the simple inverse kinematics of the arms, it first computes the grasp configurations for the initial configuration $\mathbf{q}^i_{obj}$ and associates them with $\mathbf{q}^i_{obj}$. Then, at every reached configuration $\mathbf{q}_{obj}$ (initially, $\mathbf{q}^i_{obj}$), the algorithm considers the neighbors of $\mathbf{q}_{obj}$ that have not been explored yet. It selects the neighbor $\mathbf{q}'_{obj}$ with the smallest potential, computes the grasp configurations for $\mathbf{q}'_{obj}$, and verifies that at least one of them has the same type as one grasp configuration previously associated with $\mathbf{q}_{obj}$. If this is the case, the algorithm stores $\mathbf{q}'_{obj}$ at the end of the current path, associates the computed grasp configurations (and the type) to it, and if $\mathbf{q}'_{obj} \neq \mathbf{q}^g_{obj}$, continues the search from there. If this is not the case, the algorithm considers the neighbor of $\mathbf{q}_{obj}$ with the next smallest potential, and so on. If the neighborhood of $\mathbf{q}_{obj}$ is fully explored without success, the algorithm backtracks.

If the algorithm succeeds and returns a path τ_{obj}, the planner next generates the path of the arms, including the ungrasp/regrasp operations. It considers the first configuration $\mathbf{q}^i_{obj}$ in τ_{obj}. One or several grasp configurations are associated with it. For each one, the planner computes the maximal subpath of τ_{obj} starting at $\mathbf{q}^i_{obj}$ such that a grasp configuration of the same type is associated with every configuration in this subpath. The planner selects the grasp configuration for $\mathbf{q}^i_{obj}$ that results in the longest subpath (call this subpath τ^1_{obj} and let T be the type of this configuration). The path of the two arms along τ^1_{obj} is described by the sequence of grasp configurations of type T associated with the configurations in τ_{obj}. Now, let $\mathbf{q}^c_{obj}$ be the last configuration of τ^1_{obj}. At this configuration of $\mathcal{M}$ (assume that it is not yet the end of τ_{obj}), the arms must change the type of their grasp configuration. By construction of τ_{obj}, it is guaranteed here that one of the grasp configurations associated with $\mathbf{q}^c_{obj}$ has the same type as a grasp configuration associated with the successor of $\mathbf{q}^c_{obj}$ in τ_{obj}. The planner selects the longest subpath τ^2_{obj} of τ_{obj} beginning at $\mathbf{q}^c_{obj}$, whose grasp configuration type remains valid throughout. It proceeds in the same way until the last configuration ($\mathbf{q}^g_{obj}$) of τ_{obj} is attained. This algorithm minimizes the number of ungrasp/regrasp operations along τ_{obj}.

At this point, the planner has computed the configurations of $\mathcal{M}$ where the arms ungrasp and regrasp the movable object to change the type of their grasp configuration. It remains for the planner to construct the path of the arms to connect the two successive grasp configurations at every ungrasp/regrasp operation, as well as the initial configuration of the arms to the first grasp configuration, and the last grasp configuration to the goal configuration of the arms. Planner 1 constructs every such path by applying the same best-first search potential field method as above, in a discretized grid placed over the four-dimensional configuration space of the two arms. Given the simplicity of the two arms, a more elegant solution is probably feasible.

We said above that, along a subpath τ^i_{obj}, the sequence of computed grasp configurations of the selected type determines the path followed by the two arms. This is actually a simplification. It corresponds to assuming that every two consecutive grasp configurations are close enough to each other to regard the transition between them as collision-free. But this assumption is violated when the two links of one arm are almost superposed, i.e. the arm is near a singular configuration. In this case a small motion of the arm's effector point may induce big changes in the arm's joint angles. Though the two consecutive grasp configurations are collision-free, there may be no collision-free path of the closed-loop chain between them. We deal with this problem as follows. We say that two grasp configurations of the same type are *close to each other* if and only if the Euclidean distance travelled by the second revolute joint of each of the two arms is of the same order of magnitude (or less) as the increment along the translational axis of the grid placed over $\mathcal{C}_{obj}$. During the search for the path τ_{obj}, when a configuration $\mathbf{q}'_{obj}$ is considered as a potential successor of $\mathbf{q}_{obj}$ (see above), it is verified that at least one grasp configuration for $\mathbf{q}'_{obj}$ has the same type as a grasp configuration associated with $\mathbf{q}_{obj}$ *and* that the two grasp configurations are close to each other. If any two successive configurations of the movable object in τ_{obj} admit grasp configurations of the same type that are not close to each other, the planner stores these configurations with different internal type names. If later two consecutive subpaths τ^i_{obj} and τ^{i+1}_{obj} have the same type of grasp configuration, with two different internal names, the transition from the last configuration of τ^i_{obj} to the first configuration of τ^{i+1}_{obj} is achieved by a transit path computed as if the grasp configuration was changing type.

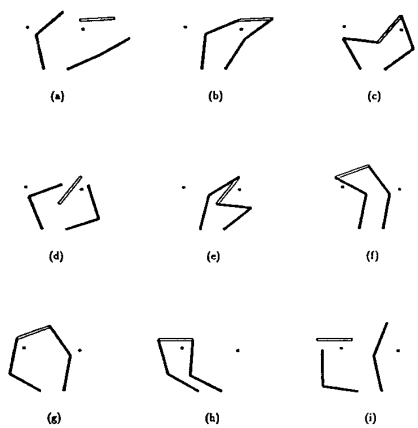

Figure 3: Path Generated by Planner 2

The path of Fig. 2 includes two regrasp operations. Planner 1 took 8 seconds to generate it. The grid placed over $\mathcal{C}_{obj}$ was $128 \times 128 \times 120$.

The best-first potential field method is resolution-complete [2], consequently Planner 1 is resolution-complete. This means that if a manipulation path exists, the planner is guaranteed to find it, provided that the resolution of the configuration space grids have been set fine enough.

5 Planner 2

Planner 2 addresses the same restricted problem as Planner 1, but with the important difference that the arms now move in the same plane as the movable object and the obstacles, and thus may collide with them. Fig. 3 illustrates this new problem with a path produced by Planner 2. The main consequence of the extension considered here is that a change of grasp configuration is no longer guaranteed to be always feasible. Planner 2 deals with this difficulty by constructing a *manipulation graph* similar to the one introduced in [1]. Each node of the manipulation graph represents a (maximal) connected component of the set $\mathcal{C}_{grasp}$ of grasp configurations. All paths lying in this subset are transfer paths. Two nodes of the manipulation graph are joined by a (non-directed) link if and only if there exists a transit path between a configuration in the first connected component and a configuration in the second. Two nodes corresponding to the initial configuration $\mathbf{q}^i$ and the goal configuration $\mathbf{q}^g$ of the arms and the movable object are added to the graph. The node representing $\mathbf{q}^i$ (resp. $\mathbf{q}^g$) is connected to a node representing a connected component if and only if there exists a transit path connecting $\mathbf{q}^i$ (resp. $\mathbf{q}^g$) to a configuration in this component.

The planner first generates the nodes of the manipulation graph as follows. $\mathcal{C}_{grasp}$ is partitioned into two subspaces, each corresponding to a specific grasp pairing. Each subspace has dimension 3. The planner discretizes each of the two subspaces into a grid and checks each configuration in the grid for collision. A sweep algorithm groups the discretized collision-free configurations into connected components. Since $\mathcal{C}_{grasp}$ is the configuration space of a closed-loop chain, we construct each grid by discretizing the values of three angles, the two joint angles of the arm $\mathcal{A}_1$ and the joint angle at the contact point between $\mathcal{A}_1$ and the movable object. The two joint angles of the other arm are derived using its inverse kinematic equations. Since these equations have two solutions, we obtain two subgrids that we glue together at configurations where the two links of the second arm are almost aligned (i.e., their angle is less than one increment in the grid). Two configurations in a grid are adjacent if they are neighbors *and* the corresponding configurations of the second arm are close to each other (i.e., their joint angles differ by approximately one increment at most). The connected components are the equivalence classes for the transitive closure of this adjacency relation.

The planner then builds the links of the graph as it searches it for a path connecting $\mathbf{q}^i$ to $\mathbf{q}^g$. Assume that the search algorithm has reached some node N (initially, $\mathbf{q}^i$) and that it now wishes to go from N to N'. In order to proceed, it must first check that there is a link between the two nodes. Let Q and Q' denote the sets of configurations associated with N and N', respectively. Since a transit path does not change the moving object's configuration, a link between N and N' can only exist if there exists at least one pair of configurations in $Q \times Q'$ where the movable object has the same configuration. We call a *jump point* $\mathcal{J}$ between N and N' any pair $\mathbf{q}_\mathcal{J} \in Q$ and $\mathbf{q}'_\mathcal{J} \in Q'$ such that the movable object has the same configuration in $\mathbf{q}_\mathcal{J}$ and $\mathbf{q}'_\mathcal{J}$. The planner computes all jump points between N and N'. If there are none, N and N' are not connected, and another potential successor of N must be considered. Otherwise, the planner considers each jump point in turn. For each one (call it $\mathcal{J}$), it searches a four-dimensional grid[2] placed over the two-arm configuration space $\mathcal{C}_1 \times \mathcal{C}_2$ for a path connecting $\mathbf{q}_\mathcal{J}$ to $\mathbf{q}'_\mathcal{J}$. The planner performs a best-first search using a simple distance function. As soon as a transit path is found the search algorithm stores N' in the search graph, and discards the remaining jump points. If none of the jump points result in a transit path, N' cannot be a successor of N. (In practice, there are often too many jump points differing among them by small variations of the movable object's configuration. In our implementation, we set a maximal number of jump points that the planner may consider to generate a transit path. Each attempted jump point is selected randomly among the possible ones.)

The current version of the planner searches the manipulation graph in a breadth-first fashion, starting at the initial node. It terminates successfully when the goal node is attained. It returns failure when it cannot reach any new node.

The outcome of the search of the manipulation graph, if successful, is a sequence of transit paths connecting connected components of $\mathcal{C}_{grasp}$. In general, two successive transit paths end and begin, respectively, at different configurations. However, by construction, these configurations are contained in the same connected component and can be joined by a transfer path. The planner generates this path by performing a best-first search of the connected component using a simple distance function.

Planner 2 is resolution complete when all the jump points are considered. For the path of Fig. 3 it took 7.5 minutes to generate the nodes of the manipulation graph and 5 minutes to search for the path (and construct the required links). The maximal number of jump points between two components of $\mathcal{C}_{grasp}$ was set to 30. The joint angles were discretized into 3-degree increments to construct the grids.

6 Planner 3

Because of the lack of redundancy of the dual-arm system used in Planner 2, many manipulation problems have no solution. In order to increase the versatility of the dual-arm system, we now consider the case where the two arms have redundant joints. For example, in Fig. 1, each arm has three revolute joints.

The increase in the number of joints yields higher-dimensional configuration spaces and makes the exhaustive techniques used in Planner 2 impractical for all non-trivial examples. This led us to develop a third planner that uses randomized potential field planning techniques. These techniques[3] have already been shown

[2]By doing so, we treat all transit paths as if they were to be two-arm transit paths; the search may, however, produce a path that does not move one of the two arms, hence a one-arm transit path.

[3]Although Section 6 is self-contained at some level of abstraction, it addresses several issues whose in-depth understanding requires prior knowledge of these techniques.

to be effective to plan paths for robots with many degrees of freedom (see [2]). The development of Planner 3 is still at an early stage, and the current implementation embeds several ad hoc restricting assumptions discussed below, which may prevent the planner from finding a manipulation path. The path of Fig. 1 was generated by Planner 3.

In the course of planning a manipulation path, Planner 3 generates one-arm and two-arm transit subpaths and transfer subpaths. Let us start with the transfer subpaths.

Because of the large dimension of $\mathcal{C}_{grasp}$, Planner 3 does not attempt to precompute the connected components of $\mathcal{C}_{grasp}$. Therefore, when it plans a transfer path, it does know if the goal configuration of the movable object $\mathcal{M}$ is directly attainable or not. For that reason, it uses an adaptation of the randomized search potential field method described in [2, 3] that allows the insertion of transit paths. The planner traces adjacent[4] configurations along the negated gradient (down motion) of a potential field defined over the configuration space grid of the closed-loop chain, until it reaches a minimum of the potential. If this minimum is the goal (global minimum), it returns the path, otherwise it makes a bounded number of attempts to escape it. Each attempt consists of either executing a random walk keeping the search within the same connected component of $\mathcal{C}_{grasp}$, or performing a transit path to move the search to another connected component. The planner chooses randomly between these two sorts of paths, which are executed as explained in the next paragraph. In the current implementation, we use the following empirical probability distribution: 0.8 for a random walk and 0.2 for a transit path.

A random walk to escape a local minimum is executed exactly as explained in [2] and is immediately followed by a down motion reaching a (hopefully) new minimum. A transit path is either a two-arm or a one-arm transit path. The type of the transit path is deterministically chosen as follows. The planner collects collision statistics at the configurations explored during the down motion before attaining the local minimum. If inter-arm collisions predominated over collisions with obstacles, it performs a two-arm transit path; otherwise, it perfoms a one-arm transit path of the arm that hit obstacles the most often. A transit path is generated as explained below and is immediately followed by a down motion in the configuration space grid placed over $\mathcal{C}_{grasp}$, until it reaches a minimum of the potential.

An attempt to escape the local minimum succeeds if the newly attained local minimum has a smaller potential; then the search proceeds from this new minimum in a depth-first manner. If all attempts fail, the planner randomly backtracks at a configuration of the current path (see [2]).

The current version of Planner 3 assumes that a two-arm transit path is always aimed at swapping grasp. It computes such a path by using the same randomized planning method as in [2], the initial configuration of this path being the local minimum that the planner tries to escape, and the goal configuration region being the set of all grasp configurations achieving the new grasp pairing.

In the case of a one-arm transit path, the planner uses the same randomized planning technique. However, we now wish the moving arm to change its position relative to an obstacle with its effector point coming back to the same grasp point G_i of the movable object. To do so, we select an obstacle randomly among those hit by the arm during the generation of the down motion, we connect G_i to the nearest point (call it P) in this obstacle, and we treat the segment G_iP as an artificial barrier not to be crossed by the moving arm. Using a wavefront propagation algorithm similar to those described in [2], we compute a potential field that attracts the moving arm's effector point toward G_i around the obstacle.

For both sorts of transit paths, the goal configuration is not uniquely defined, which poses no particular difficulty to the randomized planning technique as long as there is at least one. If there is none, this technique is in general unable to detect it and may run for ever. One way to deal with this difficulty is to put a time limit on the search for a one- or two-arm transit path and assume failure when this limit is attained. Instead, the implemented planner verifies the existence of a goal configuration. For example, in the two-arm transit path case, it attaches the two effector points to the grasp points that they should attain, and moves the arms systematically through a grid placed over the cross-section of $\mathcal{C}_{grasp}$ defined by the current configuration of the movable object, until a collision-free configuration is found or the cross-section is fully explored. This technique is practical as long as the arms have not too many joints. In our experiments we used 3-revolute-joint arms (as shown in Fig. 1). Thus a cross-section of $\mathcal{C}$ at a fixed configuration of $\mathcal{M}$ has only dimension 2 (1 for each arm). The case of a one-arm transit path is treated in a similar fashion, and requires the planner to explore an even smaller grid (dimension 1).

All subpaths including random walks are smoothed by Planner 3 in a postprocessing step [2].

Planner 3 generated the path shown in Fig. 1 in approximately 5 minutes, with the joint angles discretized in

[4] The adjacency of two configurations is defined as in Planner 2.

3-degree increments.However, the planner also failed to generate paths for similar problems that had a solution (actually, the planner did not return failure, but we stopped it after a long computation time). Such failures became even more frequent as we increased the number of obstacles in the workspace.

Three design assumptions made above appear to be the main causes for the failures of Planner 3:

(1) Only one-arm transit paths are attempted to change the homotopic position of the closed-loop chain relative to the obstacles (two-arm transit paths are only aimed at swapping grasp). But there are cases where a one-arm transit path cannot be generated because of obstruction by the current configuration of the other robot.

(2) The construction of barriers to create potential fields forcing an arm to move around one or several obstacles is too simplistic. It works well when there are few obstacles, but it becomes inadequate when the number of obstacles increases, explaining the failures of the planner when we increased the number of obstacles.

(3) Transit paths are only executed upon reaching local minima of the potential. This may not be sufficient. Perhaps transit paths should also be inserted during down motions of the potential. The issue here is more subtle, and we have no obvious experimental examples so far where the planner failed because of this restriction.

There are several ways to eliminate or alleviate these assumptions. However, the difficulty is that each modification of the current planner tends to have undesirable side-effects, such as increasing the running time for simple problems. We think that improving the planner requires a significantly better understanding of the phenomena sketched above than we now have. We currently try to develop this understanding by analyzing theoretical models of the connectivity of the composite configuration space, and by conducting additional experiments with the planner.

7 Conclusion

We have described a new motion planning problem, which we call dual-arm manipulation planning. This problem occurs in various tasks aimed at moving long and/or heavy objects. For example, in the construction domain, using multiple arms to move pipes could result in more cost-effective manipulation and yield major improvements in productivity and safety. In space, the construction of truss structures will require manipulating long beams with light-weight arms.

We have described three implemented planners that solve increasingly complicated versions of the dual-arm manipulation problem. The first two planners use systematic techniques to search through configuration space grids. They are resolution-complete, but they can only deal with arms with few degrees of freedom. The third planner uses randomized search techniques that allow it to deal with more degrees of freedom. However, in its current version, it embeds ad hoc assumptions that significantly affect its reliability.

References

[1] R. Alami, T. Siméon and J.P. Laumond, "A Geometrical Approach to Planning Manipulation Tasks: The Case of Discrete Placements and Grasps," in *Robotics Research 5*, edited by H. Miura and S. Arimoto, The MIT Press, Cambridge, 1990, pp. 453-459.

[2] J. Barraquand and J.C. Latombe, "Robot Motion Planning: A Distributed Representation Approach," *The Int. J. of Robotics Research*, 10(6), December 1991, pp. 628-649.

[3] J.C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Boston, MA, 1991.

[4] J.P. Laumond and R. Alami, *A Geometrical Approach to Planning Manipulation Tasks: The Case of a Circular Robot and a Movable Circular Object Amidst Polygonal Obstacles*, Tech. Rep. No. 88314, LAAS, Toulouse, 1988.

[5] J.P. Laumond and R. Alami, *A Geometrical Approach to Planning Manipulation Tasks in Robotics*, Tech. Rep. No. 89261, LAAS, Toulouse, 1989.

[6] T. Lozano-Pérez, "Spatial Planning: A Configuration Space Approach," *IEEE Tr. on Computers*, 32(2), 1983, pp. 108-120.

[7] S.A. Schneider, *Experiments in the Strategic and Dynamic Control of Cooperating Manipulators*, Ph.D. Dissertation, Dept. of Electrical Engineering, Stanford University, 1989.

[8] J.T. Schwartz and M. Sharir, "On the 'Piano Movers' Problem: II. General Techniques for Computing Topological Properties of Real Algebraic Manifolds," *Advances in Applied Mathematics*, 4, 1983, pp. 298-351.

[9] J.T. Schwartz and M. Sharir, "On the 'Piano Movers' Problem: III. Coordinating the Motion of Several Independent Bodies. The Special Case of Circular Bodies Moving Amidst Polygonal Barriers," *The Int. J. of Robotics Research*, 2(3), 1983, pp. 46-75.

[10] J.T. Schwartz, M. Sharir and J. Hopcroft (editors), *Planning, Geometry, and Complexity of Robot Motion*, Ablex, Norwood, NJ, 1987.

[11] J.T. Schwartz and C.K. Yap (editors), *Algorithmic and Geometric Aspects of Robotics*, Lawrence Erlbaum Associates, Hillsdale, NJ, 1987.

[12] G. Wilfong, "Motion Planning in the Presence of Movable Obstacles," *Proc. of the 4th ACM Symp. of Computational Geometry*, 1988, pp. 279-288.