Planning Single-Arm Manipulations with N-Arm Robots*

Benjamin Cohen
bcohen@seas.upenn.edu
University of Pennsylvania

Mike Phillips
mlphilli@andrew.cmu.edu
Carnegie Mellon University

Maxim Likhachev
maxim@cs.cmu.edu
Carnegie Mellon University

Robotic arms often share a common workspace. Examples range from dual-arm manipulation platforms to factory floors equipped with dozens of industrial manipulators. Yet, despite sharing heavily overlapping workspaces, these arms are frequently treated as independent entities and collaboration between the arms is minimal. On the other hand, effective cooperation between them can dramatically increase their productivity and overall cost-effectiveness. For instance, they can share common tools by passing them to each other when necessary and they can pick up assembly parts from remote locations and pass them to the arm(s) requiring them. The key requirement among these tasks is the ability to plan arm motions that relocate an object from one location to another in the workspace spanned by the N-arm robotic system.

This planning problem is challenging for several reasons. First, it requires reasoning over which arms and in which order should manipulate the object. In the simplest scenarios a single arm may be able to relocate the object. In other scenarios, all N arms may potentially be involved if the object is relocated from one end of the workspace to the opposite end. Furthermore, sometimes, the same arm may have to re-grasp the object several times with the help of another arm in order to grasp the object in a way that allows the arm to place the object at the goal with the desired orientation. Second, the planning problem also involves computing valid locations for each of the handoffs between two consecutive arms. Finally, the planner also needs to consider different possible grasps and plan these grasps in a way that allows for the successful sequence of handoffs. In this paper, we show how to exploit the characteristics of this problem in order to construct a planner that addresses these challenges within a single search.

Overview

The problem is to move an object from a start pose to a goal pose in SE(3) without colliding with obstacles in the environment. The object can only be moved by an arm that is currently grasping it. We are given a set of manipulators in the environment that can be used to move the object. We are also given the list of valid grasps for the object. The planner has to find a joint space plan for all the arms that results in moving the object from its start to goal pose where neither the object nor arms collide. This plan can include "handoffs" which allows the object to be transferred from one arm to another.

The full dimensionality of this problem is immense as it includes all the degrees of freedom for each arm as well as the 6 DoF pose of the object. This simply doesn't scale when planning for a large number of arms. One of the key insights to our approach is that for the vast majority of scenarios only one arm is relevant at any given point in the movement of the object, except during a handoff where the configurations of two arms matter. Our novel representation plans for the 6D pose of the object floating through space while ensuring there is a "support arm" at each pose.

Our problem has both discrete and continuous components. The discrete component is finding the sequence of arms and grasps needed to support the object throughout its motion. The continuous problem comes from the manipulators and object moving through a continuous configuration space. Our algorithm uses a search-based planner (a variant of A*) which plans on a graph. Such planners excel at discrete problems (Hart, N. J. Nilsson, and Raphael 1968) as graphs are inherently discrete and can use informative heuristics to focus the search toward the solution. Search based planners have also been shown to perform comparably to their sampling-based counterparts in continuous planning problems such as manipulation (Cohen, Chitta, and Likhachev 2014). A search-based planner is a natural choice for this problem as it seamlessly combines the discrete and continuous parts of the problem. In our problem we will be planning for the object with a heuristic that guides the path of the object through the appropriate sequence of handoffs.

Representation. We represent the planning problem with a graph. Since we are assuming that all arms not currently supporting the object are in their "safe configurations", a state only needs to contain information about the pose of the object and the arm supporting it. There are 6 dimensions for the position and orientation of the object. There is also one variable representing the arm that is supporting the object and one for the grasp it is using. These two dimensions are just indices since we are given a finite lists of arms and grasps. Finally, while this is enough information to know where the gripper of

*This is a short version of a paper that appeared at RSS'14

the supporting manipulator is, for many manipulators this doesn't map to a single set of joint angles. There are often many solutions for a particular end-effector pose. These manipulators can have this redundancy captured by a set of "free angles", denoted below as $\phi_0, \ldots, \phi_m$. For example, in our experiments we use PR2 arms which have 7 degrees of freedom. An end-effector pose can be mapped to a set of joint angles given one free angle (in this case the upper arm roll joint). Formally a state vector is defined as: $\{x_{obj}, y_{obj}, z_{obj}, roll_{obj}, pitch_{obj}, yaw_{obj}, arm_{id}, grasp_{id}, \phi_0, \ldots, \phi_m\}$.

Motion primitives are actions that can be applied at any state in the graph to transition to neighboring states. There are motions for modifying the object position and orientation state variables by small amounts. There are also motion primitives for changing the support arm free angles. All of these motions are computed by running inverse kinematics with the new object pose or free angle.

The last kind of motion primitive is to switch supports. To do this a newly selected arm must move from its safe configuration to the newly selected grasp and then the previous support arm has to move back to its safe configuration. These motions can be quite complicated so we call an arm planner for a short period of time to check for feasibility. The planner we chose to use is also a search-based motion planner (Cohen, Chitta, and Likhachev 2014). Evaluating this edge can be expensive.

Search. When planning in high-dimensional problems the graph is too large to compute up front and is instead generated as the search progresses. One of the characteristics of our problem is that some edges in our graph are far more expensive to evaluate than others. In particular, determining if the object can switch support arms requires two calls to a single-arm path planner. Additionally, most states have the option to transition to more than one arm for several different grasps, which is a computationally expensive edge to evaluate. We use a lazy version of Weighted A* (Pohl 1970) search, which fully evaluates edges only when the planner intends to use them, drastically reducing a potentially large waste of computation time.

Heuristic. Our heuristic is the sum of two components: guiding the object toward the goal and encouraging the most promising handoffs.

In order to guide the search toward the goal, we run a reverse Dijkstra search for the 3D location of the object backward from the goal to all (x,y,z) cells in the environment while accounting for obstacles. We call this a "Point Search". This approximates the distance from all positions in the world to the goal location. In order to capture some of the orientation information of the goal pose, we actually run several of these Point Searches for different points on the object.

The second component of the heuristic is a handoff penalty. We approximate the number of remaining handoffs needed to get the object at the goal pose. This approximation is formulated as a dynamic programming problem. It computes the minimum number of handoffs required to get to the goal assuming that the object is being held by arm with $grasp$. This heuristic discourages using unneeded handoffs, encourages needed handoffs and helps the planner choose the right grasps for the handoff. Details of the heuristic computation and how we ensured it is admissable are discussed in (Cohen, Phillips, and Likhachev 2014).

	success rate	mean time(s)	max time(s)	mean expands	mean handoffs
tabletop (2-arm, tray)	100	10.25	36.52	226	1.33
bar (3-arm, tray)	50	16.93	39.93	857	2
checkout (3-arm, tray)	100	9.67	16.22	220	1.8
factory (4-arm, rod)	100	36.09	78.41	2239	1.8

Table 1: Performance of the planner in four different scenes.

Experimental Results

To measure the performance of the planner, we generated a set of four realistic scenes in which we strategically placed between two and four arms such that almost all areas of interest can be reached by at least one. The manipulators used are the 7 DoF arms of the PR2 robot. The position of only a single redundant joint has to be recorded in the state representation, resulting in a total of nine dimensions. The four scenarios include a PR2(with two arms) infront of a cluttered tabletop, three arms working behind a bar, three arms at a self-checkout counter and four arms configured around a car chassis on an assembly line in a factory. The planning problems in each scenario were created by hand to assure that each one requires at least one handoff and is challenging either because of obstacles, kinematic constraints or restrictive start and goal poses. The planner is given a maximum of 100 seconds to compute a solution.

Results from the 25 trials can be seen in Table 1. Overall, the average planning time is 18.2 seconds and the planner was successful in planning in 20 of 25 total trials, for a total success rate of 80%. In particular, we found that in most of the trials the minimum number of handoffs required were performed but not in all of them. Additionally, we were pleased to see that the planner successfully determined that because of the initial object pose, a series of back and forth handoffs were needed between the arm used to put the object down and the arm that transferred the object to it, so that the final grasp was capable of achieving the goal pose.

References

Cohen, B. J.; Chitta, S.; and Likhachev, M. 2014. Single- and dual-arm motion planning with heuristic search. *The International Journal of Robotics Research* 33(2):305–320.

Cohen, B.; Phillips, M.; and Likhachev, M. 2014. Planning single-arm manipulations with n-arm robots. In *Robotics Science and Systems (RSS)*.

Hart, P. E.; N. J. Nilsson; and Raphael, B. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems, Science, and Cybernetics* SSC-4(2):100–107.

Pohl, I. 1970. First results on the effect of error in heuristic search. *Machine Intelligence* 5:219–236.