Planning Two-Fingered Grasps for Pick-and-Place Operations on Polyhedra

Joseph L. Jones and Tomás Lozano-Pérez

MIT Artificial Intelligence Laboratory
545 Technology Square
Cambridge, MA., 02139

Abstract

In this paper, we focus on the crucial step in a pick-and-place operation – the choice of grasp. We describe a new approach to choosing grasps, implemented in the task level planning system HANDEY, which attempts to deal in a general fashion with the interaction between the choice of grasp and the choice of paths to reach the grasp. The approach is based on a systematic use of the configuration-space representation of the motion constraints on the robot. A very simple and efficient algorithm for approximating the configuration-space constraints leads to a practical method.

Introduction

The goal of the HANDEY project is to endow an industrial robot with the most basic competence one expects of a device built to manipulate objects: the ability to pick up user-specified objects and place them at user-specified positions. This type of manipulation problem is called a **pick-and-place problem**. Note that pick-and-place represents a small subset of the manipulation problems we would ultimately expect a robot to perform. But, almost every manipulation problem contains some aspect of the pick-and-place problem.

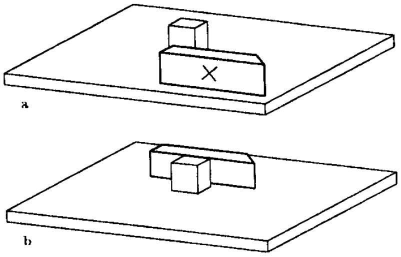


Figure 1: Interaction between stability of a grasp and collisions with obstacles.

In this paper, we focus on the crucial step in a pick-and-place operation, which is the choice of grasp. We describe a new approach to choosing grasps, implemented in HANDEY, that attempts to deal in a general fashion with the interaction between the choice of grasp and the choice of paths to reach the grasp.

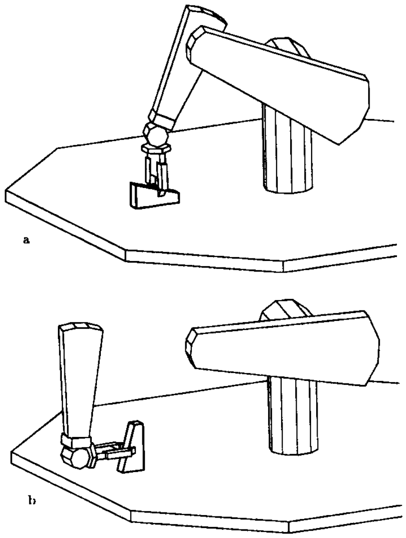


Figure 2: Interaction between grasp choice and kinematic limits.

We will start by reviewing the constraints on the choice of grasp imposed by a general pick-and-place operation. Then, we will describe our approach to dealing with these constraints.

Pick-and-place constraints

The following list summarizes the constraints that must be satisfied by a valid pick-and-place operation:

1. A collision-free, kinematically feasible path exists from the robot's starting point to the grasp at the object's pickup position (and orientation).
2. The grasp is stable; that is, the object will not twist or slip relative to the gripper.
3. This grasp is such that no part of the robot is in collision with any obstacle at either the object's pickup or putdown position.

CH2876-1/90/0000/0683$01.00 © 1990 IEEE

4. The grasp is kinematically feasible at *both* the object's pickup and putdown position.
5. A collision-free, kinematically feasible path exists from the object's pickup to its putdown position for the robot holding the object.
6. A collision-free, kinematically feasible path exists from the object's putdown position to the robot's final position.

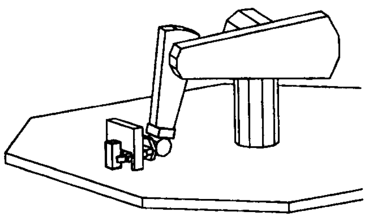


Figure 3: Interaction between grasp choice and pickup departure path.

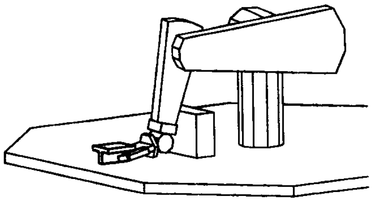


Figure 4: Interaction between grasp choice and putdown departure path collisions.

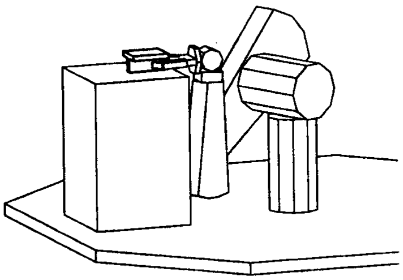


Figure 5: Interaction between grasp choice and putdown departure path kinematic constraints.

While the robot must execute the steps of a pick-and-place operation in the above order, it is not advantageous to actually plan them in this order. The problem is that valid choices made early in the sequence can eliminate all valid choices for steps later in the sequence. For example, the choice of an initial grasp may preclude finding a valid placement at the destination. These interactions are examined in more detail below.

A few examples will illustrate how a plan may fail if each of the above constraints is satisfied in isolation. Figure 1 demonstrates a simple interaction between the choice of a stable grasp and collisions with obstacles in the world. The most stable grasp for picking up the trapezoidal-shaped block in Figure 1.a would clearly be a grasp in which the fingers of the gripper overlap the center of the largest visible face (marked with an 'X'). However, if the goal is to place this block behind the cube as in Figure 1.b it can be seen that this is possible only if a less stable grasp is chosen in the preceding step.

Another common way for a plan to fail is choosing a grasp which is kinematically incompatible with the commanded putdown position. See Figure 2. The goal in this case is simply to rotate the wedge shaped object about its lower left edge – yet it is impossible for the robot to accomplish this if it makes the most obvious choice of initial grasp. People can solve both preceding problems by making the obvious grasp choice then repositioning the part in their hand on the way to its putdown position. Such an option is unavailable to two fingered robots.

Some more subtle interactions are depicted in figures 3, 4, and 5. Suppose that in the first of these we wish to rotate the block about its lower left edge. The grasp shown satisfies the stability requirement, it produces no collisions with any obstacles, there is a kinematic solution at both the pickup and putdown positions, there is a viable path from the robot's starting position to the grasp, and the robot could find a viable departure path. However, as can readily be seen, for this choice of grasp there is no collision free path that will achieve the goal.

Figure 4 shows a 'T' shaped block that the robot has just placed in its final position on the table. Even at this point the grasp choice can cause problems. In this example it is impossible for the robot to extricate itself without colliding either with the object it has just placed or the block behind the gripper.

At first glance Figure 5, which shows the placement of a 'T' shaped object on the large block, seems to have succeeded. In fact, once again the combination of grasp choice and kinematic solution at the putdown position has caused the failure of a plan. In this case the shoulder joint (which connects the large horizontal cylinder and the large wedge shaped link) and the elbow joint (which connects the two wedge shaped links) are both so close to their kinematic limits that it is not possible for the gripper to move away from the object.

Previous work

There has been a substantial amount of previous work in grasping. A very thorough review of this work can be found in [Pertin-Troccaz 89]. Most of this work has focused on grasping the part in a particular environment without considering all the constraints required to ensure the success of a pick-and-place operation.

Our previous work on grasping in HANDEY [Lozano-Pérez et al. 89] has considered some approximation to these constraints. But, because the earlier planner employed an incremental path search technique akin to the potential-field method [Khatib 86], we could not deal with a very precise form of the constraints. In particular, when approaching a grasp we attempted to avoid the union of the obstacles at the pickup and the putdown positions. If a path to a grasp point could be found that avoided this union, then one could guarantee that the grasp was feasible. However, this constraint is too strong. The approach path itself does not need to be constrained by the obstacles at the putdown; it is only the final grasp that should be so constrained. This distinction was essentially impossible to capture using our previous approach.

The approach to grasp-planning presented here deals in detail with most of the pick-and-place constraints when choosing a grasp, for example, guaranteeing safety, kinematic feasibility, and reachability at both pickup and putdown. One other constraint is not handled at all (except to detect a violation), namely, guaranteeing that the choice of a grasp does not prevent the robot from taking the grasped object from its pickup point to the goal.

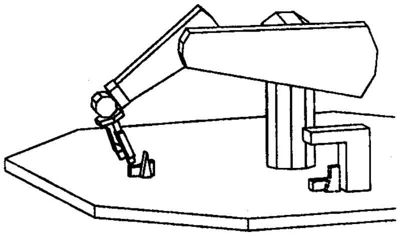


Figure 6: A sample pick-and-place problem. The highlighted object must be repositioned to the location indicated by the dashes

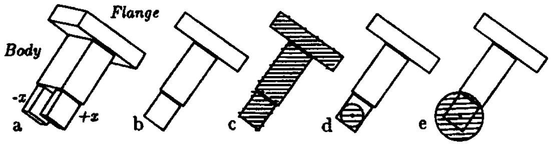


Figure 7: *(a)* The gripper is modeled as four rectangular solids: **fingers** labeled $+x$ and $-x$, a **body**, and a **flange**. This type of model is adequate for most parallel-jaw grippers. *(b)* Planar representation of the gripper. *(c)* Scan line representation. *(d)* Scan representation of orientation independent finger overlap region. *(e)* Finger non-overlap region

Some basic assumptions and decisions

Our approach to grasping is carried out under the following simplifying assumptions:

1. All the objects are modeled as polyhedra, including non-convex polyhedra. The actual objects may have smooth surfaces but the system will treat them as polyhedra. Figure 6 shows a simple pick-and-place operation of the type that we are addressing.
2. The robot grasps objects using parallel jaw grippers. Figure 7.a shows the model of a parallel-jaw gripper we use. The dimensions can be specified by the user, the key modeling assumption is the division into four independent components.
3. At least one object face is involved in a grasp thus there is a well-defined **grasp plane** parallel to this face. The approach to and departure from a grasp is done with the fingers moving parallel to the grasp plane. See Figure 8.
4. The grasp process is separated into the following phases:
    (a) Joint-interpolated gross motion from the origin to **grasp approach point.** This is planned using the gross-motion planner described in [Lozano-Pérez et al. 89].
    (b) Cartesian motion of the gripper in the grasp plane from the grasp approach point to the **grasp point.**
    (c) Joint-interpolated gross motion of the arm to the destination with the grasped object.
    (d) Cartesian motion of the gripper in the grasp plane from the grasp point to the **grasp departure point.**

    The grasp planner chooses the grasp point, the grasp approach and departure point, and the cartesian paths for the gripper to and from the grasp point.
5. The joint-interpolated gross motions in the grasp plan are planned independently and without regard for any considerations of grasping. This is the key point where the grasp planner ignores interdependencies in its decisions. As we saw earlier, the choice of grasp should take into account the existence of a gross motion between the pickup and putdown positions. We have compromised this guarantee in exchange for computational efficiency.
6. All of the planar objects that we deal with in the algorithm are approximated using a **scan-line** representation. An example of the scan-line representation of a planar gripper model is shown in Figure 7.c. This is a standard representation used in graphics to paint polygons onto a raster-scan device. The scan-line representation allows the computation of the configuration-space (C-space) constraints for arbitrarily shaped objects in a simple and efficient fashion. See section *C-Spaces for scan-converted objects.*

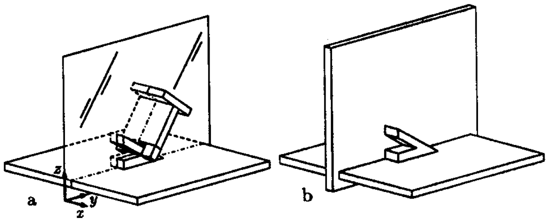


Figure 8: The motion of the gripper wrist while grasping the 'V' shaped object is confined to the plane in *a*. The motion of the $-x$ finger sweeps out the volume in *b*.

Outline of the approach

The approach to planning a grasp proceeds as follows:

1. Choose a pair of features to grasp. As pointed out earlier, at least one of the features should be an object face. We will not concern ourselves with this choice here; we assume the grasp features have been chosen and that the grasp plane has been determined.
2. Project any obstacles that could intersect the path of the gripper while moving in the grasp plane into the grasp plane. In fact, four different projections are made: one for each of the basic components of the gripper. These planar projections represent the obstacles for a planar projection of the gripper moving in the grasp plane. These projections are done for obstacles both at the pickup and putdown location. Figure 9 shows what these obstacles look like for each of the gripper components in the sample problem from Figure 6. Note that the obstacles are defined in the

coordinate system of the grasp face. The obstacles from the pickup and the putdown locations are *not* merged at this point; that would overconstrain the motion of the gripper.

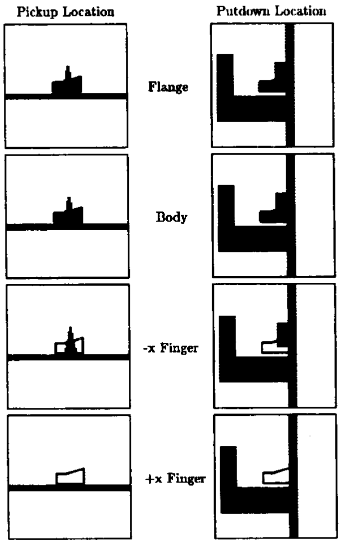


Figure 9: Obstacles for each of the gripper components at the pickup and putdown locations.

Figure 10: *(a)* The set of candidate grasp points and *(b)* the set of candidate approach or departure points for our example.

3. At this point we characterize the locations of the gripper that are candidate grasps and those locations that are candidate approach/departure points. A candidate grasp is one in which the gripper fingers overlap some part of *both* grasp features and overlaps none of the projected obstacles. A candidate approach/departure point is a position where the gripper fingers overlap *neither* of the grasp features and none of the projected obstacles. In principle, we would need to characterize gripper locations as both positions and orientations in the grasp plane. We simplify the problem by considering inscribed and circumscribed circles in the gripper fingers and requiring that the inscribed circle overlap the intersection of the grasp features (for a grasp point) and that the circumscribed circle not overlap the union of the grasp features (for approach/departure points). The center of the circles is used to characterize the position of the gripper. See Figure 7.d. The non-hatched areas in Figure 10 show (a) the set of candidate grasp points and (b) the set of candidate approach/departure points for our example.

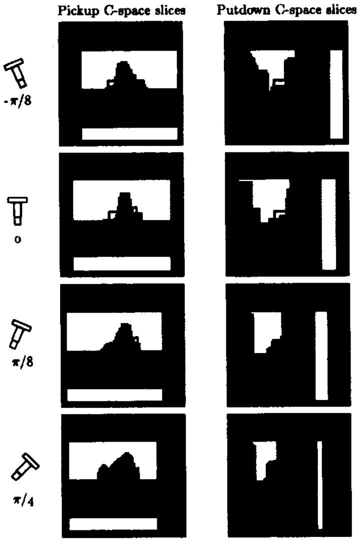


Figure 11: A typical set of x, y slices representing the gripper position constraints for different orientations of the gripper.

4. Now, we compute a set of representations of the configuration space of the gripper; one for moving near the pickup point and another for moving near the departure point. What is actually computed is an approximation to the motion constraints of the whole robot while the gripper is moving in the grasp plane. For such a cartesian motion, the C-space of the robot can be parameterized by (x, y, θ) where x, y characterizes the position of a reference point on the gripper (the center of the grasp circle used above) and the θ is the orientation of the gripper relative to the coordinate frame of the object face. These x, y, θ constraints are computed as a set of x, y **slices** each defined for some range of θ. The details of this computation can be found in the ***C-spaces for scan-converted objects*** seciton Note that the C-space constraints must first be computed for each of the gripper components and then merged into one set of slices for the whole gripper and then modified to incorporate the constraints on the rest of the robot. A typical set of slices representing the gripper constraints are shown in Figure 11. (In all C-space drawings the horizontal lines indicate unreachable points and white areas indicate reachable points. The object face is superimposed for clarity.)
5. The key step is now possible. We can search in the combined C-spaces for pickup and putdown for a combination of grasp approach point, grasp point, and grasp departure point that can all be connected in the configuration space, that is, for which collision-free space paths exist. See Figure 12. This step is described in more detail in the next section.

Searching for an approach and departure path

The choice of grasp, approach, and departure points is done by a search process that looks for a set of connected points in the

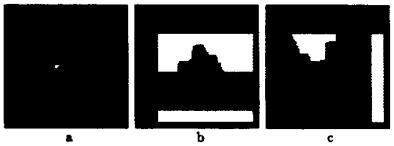


Figure 12: For gripper orientation $\theta = -\pi/8$: *(a)* The set of points for which a grasp exists at both pickup and putdown. *(b)* The set viable approach points. *(c)* The set of viable departure points.

C-spaces at pickup and putdown. The goal is to find:

- An approach point in some orientation of the pickup C-space that is a candidate approach/departure point, *i.e.*, does not overlap the grasp features.
- This point must be connected through the empty areas of the pickup C-space to a candidate grasp point, i.e., one that overlaps both faces. This point can also be at any orientation in the C-space.
- This candidate grasp point must also be connected to a legal departure point at some orientation in the putdown C-space.

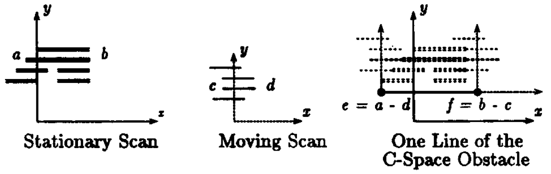
This search is carried out by starting with the candidate grasp points and marking regions in adjacent slices of the pickup and putdown C-spaces that are reachable. The search terminates in each C-space when we reach approach/departure points in the corresponding pickup and putdown C-spaces that are collision-free.

C-spaces for scan-converted objects

The computation of configuration-space constraints for a moving object is often thought of as a very time consuming operation. It is true that as the degrees of freedom of the moving object increase, the time to compute explicit C-space constraints grows exponentially with the number of degrees of freedom. This is natural since, after all, one is computing volumes in these spaces. However, for low-order C-spaces, it is possible to devise efficient algorithms for such computations. For example, the C-space obstacle corresponding to a convex translating polygon and a convex stationary obstacle can be computed in linear time [Lozano-Pérez 83]. One could use this algorithm as the basis for a slice approximation for a rotating moving object. Two practical difficulties arise in doing this: (a) non-convex objects must be partitioned into convex objects and (b) some representation of the empty space outside of the obstacles must be constructed. The latter is usually the more cumbersome task.

In fact, one simple way of avoiding both of these practical problems is to represent the input objects and the C-space as arrays. Then there are simple algorithms for computing the C-space constraints and the array provides a natural representation. This approach has been used in many implementations. Of course, such a representation does not take advantage of any of the coherence in the input or output; each point is treated separately. A boundary representation is at the other extreme; it exploits coherence, but the algorithms are more difficult.

There are two well-known techniques for exploiting coherence in arrays: one is the quad-tree representation and the other is the scan-line (sometimes known as run-length encoding) representation. A number of researchers have suggested using quad-trees for computing C-space constraints [Faverjon 84], [Noborio, Naniwa, Arimoto 89]. Typically, they assume that the objects are represented by their boundary description and the quad-tree is used to represent the C-space obstacles. We have adopted the scan-line representation because we believe it leads to simpler algorithms. Furthermore, we have adopted a scan-line representation for the input objects as well as for the C-space constraints. This leads to extremely simple and efficient algorithms.



Stationary Scan

Moving Scan

One Line of the C-Space Obstacle

Figure 13: The range of positions of the moving object that are forbidden due to the possible collision of one scan line of the moving object with one scan line of the stationary obstacles. Line *ab* from the stationary object and line *cd* of the moving object generate C-space obstacle line *ef*.

The basic operation of the algorithm is the computation of the range of positions of the moving object that are forbidden due to the possible collision of one scan line of the moving object with one scan line of the stationary obstacles (Figure 13). The resulting slices are simply merged into a scan-line representation of the output. An important speedup can be obtained by never computing the detailed constraints for a C-space slice that is already full.

The x, y, θ C-space is approximated simply by repeating this process for different ranges of orientation of the gripper. The current implementation computes slices only on demand as they are needed to find a path. This is easy to do, as the search proceeds sequentially through the slices until an answer is found.

The algorithm we have been describing handles the constraints due to collisions of the gripper with nearby obstacles. In fact, we also need to consider two other sorts of constraints: kinematic feasibility and collisions with other parts of the robot.

Kinematic constraints

The limitation of the motion of the gripper to the grasp plane simplifies the grasp planning tremendously since it reduces a six degree-of-freedom motion-planning problem to a three degree-of-freedom problem. The disadvantage of any such cartesian planning problem is that the kinematic feasibility becomes a major problem. In the joint-space of the robot, kinematic limits can be represented very easily; not so in the cartesian space. Our approach to this problem is to use a very simple pre-computed approximation to the reachable workspace of the robot. The representation is chosen to mesh well with the demands of the grasp planning process.

To understand our representation imagine the robot gripper at a fixed location in the workspace with the gripper aligned to some specified plane. Then the only degree of freedom is the range of orientations of the gripper about the normal to the plane. There will typically be a single range for each of the distinct configurations of the robot, *e.g.*, elbow up or elbow down. What we do is compute these ranges for a large sample of plane orientations at a particular point. Then we consider a series of nearby points, repeat the process and intersect the resulting ranges for each plane. We discard any small ranges remaining after the intersection. This process needs to be done only once for a particular robot and gripper.

The result of this process is a conservative characterization of the kinematics in some region of the workspace. The resulting ranges represent legal orientations of the gripper within some region when the gripper is in a particular grasp plane. When building the C-space constraints for a particular grasp, we identify the nearest region and the nearest kinematically characterized grasp plane within the region. Then, we only need to consider ranges of θ within the legal ranges stored there.

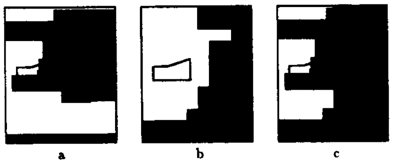


Figure 14: Merging gripper and arm constraints. *(a)* C-space constraints due to gripper and obstacles. *(b)* C-space constraints due to arm kinematic limits and collision of arm with obstacles mapped into grasp plane. *(c)* Union of all constraints.

Considering the rest of the robot

The constraints we have discussed so far are limited to potential collisions with the gripper. In general, we need to consider potential collisions with the rest of the robot. We do this as follows.

The joint-space gross-motion planning algorithm used in HANDEY computes a representation of the valid regions in the joint space of the first three joints of the robot. These regions are represented as a collection of legal ranges of the third joint defined for all reachable combinations of the first two joints. The range of motion of the third joint for fixed values of the first two joints trace out circles in the work space where the robot wrist (at the end of the third joint) can be positioned.

Every position and orientation of the gripper also defines a unique wrist position. When the gripper is moving in the grasp plane, all the wrist position will also be in a plane, call it the **wrist plane**, parallel to the grasp plane. For a position and orientation of the gripper to be valid, its corresponding wrist position must also be one of the wrist positions generated by the valid range of joint computed by the gross-motion planner. This is a necessary but not sufficient condition; one must also satisfy the full kinematic constraints that capture the limited range of motion of the wrist angles, but we have already done that as in the previous section.

The intersection of the wrist circles defined by the valid joint angles (from the gross-motion planner) with the wrist planes at pickup and putdown characterize all the wrist positions reachable without an arm collision. These positions can be transformed directly into constraints on the legal position and orientation of the gripper and used to limit the C-spaces. See Figure 14.

Conclusion

The method outlined here represents a tradeoff between generality and efficiency in planning grasps. The current implementation is somewhat slower than our earlier more conservative implementation based on potential fields but it tends to find grasps more reliably than the earlier one, thus reducing the need to plan expensive regrasping operations. Finding a solution for the sample problem from Figure 6 requires building a single θ slice of the C-space and requires approximately 20 seconds to plan (in Lisp on a Sun Sparcstation 1). More than half that time is devoted to the initial obstacle projection into the grasp plane; this needs to be done only once (for a given choice of grasp plane). Each additional θ slice can be computed in a few seconds (between five and ten). Further elaboration of the algorithm should produce an implementation that is even faster than the less powerful incremental method.

The main limitation of the method as described above is that it does not take into account the effect of the choice of grasp point on the existence of a path from pickup to putdown. This is a difficult problem that deserves further work.

Acknowledgements

This work was funded primarily by the Office of Naval Research under the University Research Initiative, through contract N00014-86-K-0685. Additional support was provided by an NSF Presidential Young Investigator Award (Lozano-Pérez).

References

[Faverjon 84] B. Faverjon, "Obstacle Avoidance Using an Octree in the Configuration Space of a Manipulator" *IEEE Intl. Conference on Robotics*, 1984, pp. 504-512.

[Khatib 86] O. Khatib, "Real-Time Obstacle Avoidance for Robot Manipulator and Mobile Robots," *Intl. Journal of Robotics Research*, Vol. 5, No. 1, Spring 1986, pp. 90-98.

[Lozano-Pérez 83] T. Lozano-Pérez, "Spatial Planning: A Configuration Space Approach" *IEEE Transactions on Computers* Vol. C-32, No. 2, Feb. 1983, pp. 109-120.

[Lozano-Pérez et al. 89] T. Lozano-Pérez, J. L. Jones, E. Mazer, P. A. O'Donnell, "Task-Level Planning of Pick-and-Place Robot Motions" *Computer* (publication of the IEEE Computer Society), Vol. 22, No. 3, Mar. 1989, pp. 21-29.

[Noborio, Naniwa, Arimoto 89] H. Noborio, T. Naniwa, S. Arimoto, "A Feasible Motion-Planning Algorithm for a Mobile Robot on a Quadtree Representation", *IEEE Intl. Conference on Robotics and Automation*, Vol. 1, 1989, pp. 327-332.

[Pertin-Troccaz 89] J. Pertin-Troccaz, "Grasping: A State of the Art," in *The Robotics Review 1*, Khatib, Craig, and Lozano-Pérez (eds), MIT Press, 1989.