Assembly Maintainability Study with Motion Planning

Hsuan Chang
Computer Graphics & Systems Prog. GE Corp. R & D
P. O. Box 8, KW–C121
Schenectady, NY 12301
changhs@crd.ge.com

Tsai–Yen Li
Robotics Lab, Computer Science Department
Stanford University
Stanford, CA 94305
li@flamingo.stanford.edu

Abstract

Maintainability is an important issue in design where the accessibility of certain parts is determined for routine maintenance. In the past its study has been largely manual and labor intensive. Either by using physical mockup or computer animation with CAD models of a design, the task relies on human to provide an access path for the part. In this paper, we present an automated approach to replace this manual process. By applying results from and developing extensions to research in motion planning and other fields, we demonstrate that an automated maintainability study system is feasible. We describe general extensions needed to adapt robotic motion planning techniques in maintainability studies. We show results from applying such a system to two classes of industrial application problems.

Key Words: Maintainability Study, Motion Planning, Design Automation, Rapid Prototyping, Visualization.

1 Introduction

Assembly maintainability studies attempt to find whether it is possible to remove a particular part (sometimes called the line replaceable unit or LRU) from an assembly, and if so, find such a path. In the past, such a task calls for a physical mockup of an assembly being designed, and a path is usually found by physically moving the mock–up part. In such an environment, when a path is found, it is very difficult to capture the sequence of motions that effected the access. More recently, as more and more assemblies are being designed using CAD systems, the trend is towards using less mock–up. With computer animation, once a path is found, the information can be systematically kept for future references. However, this trend presents a tremendous new challenge to the designers as well. For instance, attempting to move a CAD model in a graphical environment is limited, due to varying user perception of the environment displayed, the absence of physical feedback from computer controlled manipulation devices, and the lack of true 3D display in general. More importantly, with such a system, the designer needs to come up with a rough path. The computer system can only alert him of collisions encountered at discrete points selected on the rough path. When a collision situation is found, the designer will have to manually make changes to the configuration of the moving part near the collision to continue the motion. Another drawback of such a system is that collision detection are performed at discrete locales. Unless the locales are chosen to be closely spaced, there will be no guarantee that the object will be able to assume a collision–free transition from one configuration to its next. The amount of tedious manual involvement prevents, on the other hand, a user from choosing many locales. Hence, the current practice is largely still a long, tedious manual process that does not guarantee its results.

In this paper, we present an automated approach to perform maintainability study. We present an extended algorithm that employs results from research in motion planning and other fields. We demonstrate, in particular, a system tailored for solving problems in assembly maintainability studies. We show necessary extensions developed to adapt general robotic motion planning techniques to perform maintainability studies. In the next section, we analyze related motion planning algorithms as they pertain to maintainability problems. In section 3, we outline our approach. Section 4 includes results from applying the approach to different industrial problems. We conclude with remarks on the effectiveness of our approach, and speculate on how research in motion planning may benefit other disciplines further.

2 Research in Motion Planning and Their Applicability

2.1 Motion Planning

The problem faced in maintainability studies resembles those that are addressed in motion planning for robots, which has been studied extensively over the last decade. In that context, a path found for a robot in an environment is a sequence of configurations of the robot linking its initial configuration with a designated final configuration. There are many literature surveys on general motion planning (e.g., [Latombe91] and [Hwang92]). What we are interested in maintainability study maps in to what is called the ‘‘Piano Mover Problem.’’ This problem can be simply described as moving a solid in

3–space with 6 degrees of freedom (dof) amongst stationary obstacles.

In general, a motion planning technique can be exact, or heuristic. An exact technique will either find a path or declare with proof that it is impossible to move the object to the destination. This type of techniques is typically costly to apply to real world problems because of the vast search space it has to cover before making a conclusion. An example of this type of algorithm is that of Schwartz and Sharir [Schwartz83b]. Their complete polynomial algorithm takes, unfortunately, $O(n^{\wedge}(2^{d+6}))$ time, where n is the complexity of the obstacles (i.e., the number of edges) and d is the number of degrees of freedom. For the Piano Mover's Problem, where d=6, the algorithm has a time complexity of $O(n^{4096})$. So for all practical purposes, it only serves as an existence proof of a polynomial algorithm that is complete (and resolution independent).

A heuristic solution, on the other hand, strives to balance the time it needs to find a path, and the completeness of its conclusion. A compromise is often reached by attempting to search only part of the overall configuration space, for instance, at a preset resolution. The overall configuration space is called the C–space, which consists of a collection of hyper surfaces delineating obstacle space from collision–free, available configurations for the moving object (or free space). For lower dof cases such as an object moving in 2–space, its C–space can be efficiently computed to provide for a complete search for a feasible path (an example of such solution is seen in [Lozano79] and [Lozano83]). For higher dof cases, it is much more difficult to solve the problem this way simply due to the limitations with current computer technology in available memory and processing speed. For all practical purposes, explicit computation of C–space when the dof is greater than 4 is impractical. Resolution dependent solutions try to sample at a discrete resolution the C–space when n is large (see [Donald87]). Other compromises include on–demand computation of C–space (e.g., [Lozano91]).

There are many other heuristic approaches proposed in the literature (see [Hwang92] for a detailed survey). One notable technique has been demonstrated in the so–called Randomized Path Planner (RPP) [Barraquand90]. It takes a probabilistic approach to take advantage of available redundant degrees of freedom that may result in more available "paths" and the fact that the environment may not be as cluttered to allow only a smaller number of paths to exist.

A similarity exists between what assembly planning offers and what maintainability expects. Research in assembly planning has produced useful results (e.g., [Wilson92a], [Wilson92b], and [Lozano93]). A close examination reveals, however, significant differences: In assembly planning, focuses are to capture the (de)assembly sequencing information

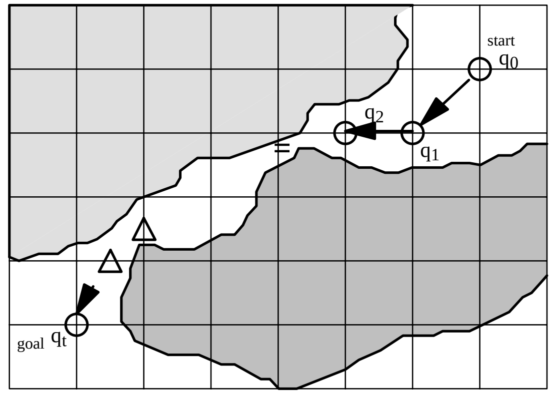

Figure 1. Search resolution in C–space. The circled configurations are attainable at the grid resolution. The triangle ones are attainable at half that resolution. The = signed one is available at one third of the initial resolution. Here the moving object is a point.

that is implied by existing surficial contact information. But for assemblies involved in maintainability studies, such contact information may be absent. Or, when exist, such information may be totally irrelevant to an intended motion that effects an access to the part.

2.2 Maintainability Study Requirements

Specifically, maintainability study calls for special attention to the following aspects of motion planning: resolution dependency, biased path, and cost of collision detection. First, modern designs strive to be compact to achieve space efficiency. An immediate result is that the environment for maintainability study tends to be more densely packed (not necessarily neatly packed so they contact one another, though). This density translates directly into a crowded C–space within which access paths lie for LRUs. In such situation, planners that employ a fixed search resolution will be less efficient, because of the reduced probability in finding those critical configurations that will lead an LRU through a small C–space opening. For instance, a fixed resolution search technique will fail given the example shown in Figure 1. In this example, the grid indicates a discrete search resolution. At this resolution, a fixed–resolution planner can not continue to make progress beyond configuration q_2. However, with a finer step size such as 1/3 (show with the = sign in the figure), one can immediately see that there exists a path for the situation. However, it is not always desirable to use a finer resolution since it takes longer to reach a fixed distance. We can envision cases where various resolutions will be needed at various stages of a search to optimize the performance of a planner. Since the C–space representation is not explicitly available for higher dof applications, some heuristics may be needed to model the search resolution adjustment process. Existing techniques in robotics motion planning do not meet this particular need.

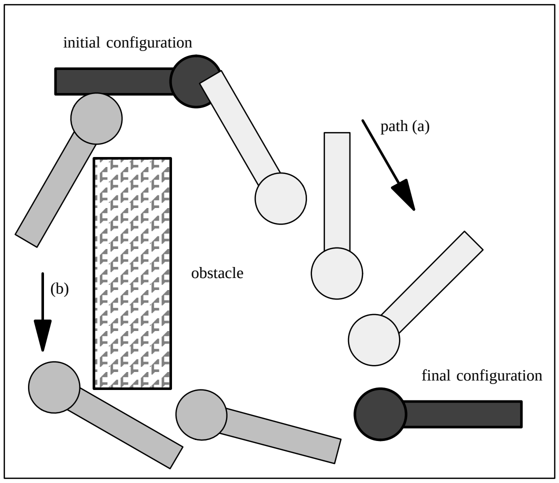

Figure 2. Typical potential guided planner will find (a) before (b), because (a) is shorter.

Secondly, sometimes there are preferred neighborhoods through which an access path should be found. This is particularly true with maintainability studies, where access is often limited by realistic concerns of an assembly such as the size of a moving object, support regions for tools, heat sources to avoid, or simply access convenience. For instance, Figure 2 illustrates a case where, using potential field guided planners such as RPP, path (a) would be found because it is a shorter path and the moving object has more room to maneuver. Instead, path (b) as shown may be preferred from maintainability point of view because of access support and ease of reach. An obvious solution to input such bias is to place additional obstacles in the free space to judiciously block certain passages. But such solution requires the user to anticipate undesirable paths before hand. For complicated 3D assemblies, it may be hard to complete such anticipation.

The third aspect is the complexity of objects and the related cost in detecting collision. To effect compliant motion (where the moving object is in contact with the obstacles as it moves), exact collision check is required. In robotics, such polygonal–based collision detection can often be avoided by using simplified models. In design automation, a sophisticated part usually replaces a subassembly of smaller parts in the old design. One consequence of such increased complexity in shape is that more polygons are needed to describe its shape, resulting in more time consuming polygonal collision checks. Another difference lies in the accuracy of collision detection. In ordinary robotic applications, it is acceptable to use approximate (and conservative) collision detection because of control uncertainty and safety concerns. In maintainability study, the need to confine to tolerances and accuracy in the models capturing the environment demands accurate collision detection.

3 An Automated Maintainability Study System

By incorporating solutions to the above concern, we developed an automated maintainability system. Its planner is based on the RPP. For collision detection, we implemented the algorithm reported in [Quinlan94] to reduce the number of calls to an exact collision checking routine that models after [Gilbert85]. We review relevant details of the RPP first (Details can be found in [Barraquand90]). Then, we elaborate on our extension to the planner to handle some of the requirements described in the previous section.

The RPP uses heuristic potential fields as a guide to search for a path and uses Brownian motions (random walks) to escape local minima of the potential fields. The heuristic potential fields are goal oriented fields created in the workspace for points selected from the robot. Each of these potential fields has unique global minimum at the goal configuration of the robot. A potential function P is defined over these potential fields such that $P(q) = 0$ iff $q = q_t$, where q is any configuration, and q_t is the goal configuration. Let C(q) be a collision checking function that returns ok when q is collision–free. The robot follows the gradient of P from q to its neighbor configuration q' if and only if $P(q') < P(q)$, C(q') = ok, and $q \neq q_t$. The distance between q and q' is the search resolution. When the search reaches a local minimum in terms of function P, a preset number of random walks, each of which is followed by a gradient motion, are performed to escape the local minimum. When all these attempts fail, a backtrack step is performed to retract part of the path found so far.

3.1 Dynamic, Adaptive Refinement of Search Resolution

A heuristic approach was developed to change the search resolution adaptively depending on how cluttered the environment is. Although we do not explicitly compute the C–space and do not know how the C–obstacle surfaces are shaped and distributed, we use the number of collisions encountered at configuration q as an indicator to how tight the C–obstacle surfaces are in the vicinity of q. Specifically, assuming x is the number of collisions encountered so far at configuration q. The heuristic model used for this adaptive multi–resolution search is given by

$$s_{i+1} = s_i * f$$

where s_{i+1} is the current step size, s_i is the successful resolution used in the last move, and .

$$f = \begin{cases} f_{max} - \sqrt{\dfrac{x}{x_s}} & 0 < x < x_s \\ f_{min} + \dfrac{1}{2}\sqrt{\dfrac{x_m - x}{x_m - x_s}} & x_s < x \end{cases}$$

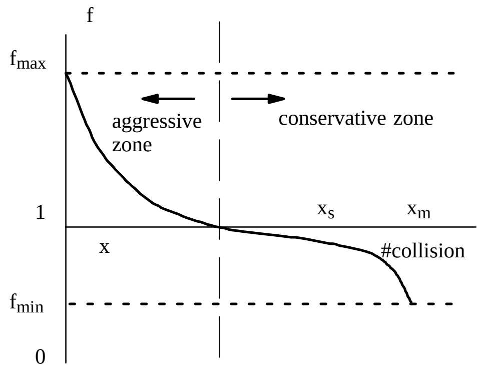

Figure 3. Search resolution refinement model: collision ratio dependent step size. Note x_m = 728 for 6 dof.

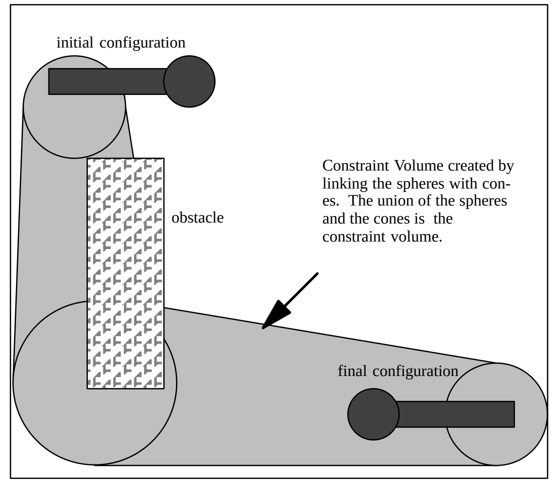

Figure 4. Constraint Volume specified by placing a set of circles (spheres for 3d) of various sizes.

which is shown in Figure 3. This model grows the step size initially if, at a given configuration q, there were only a few collisions before a feasible q' is found. We call this region of f the *aggressive zone*. Once there are more than a fixed number of collisions x_s detected at q, f becomes smaller than 1, which effectively makes s_{i+1} smaller than s_i. An important parameter is x_s. It is essentially the number of collisions one assumes under normal circumstances the planner expects before making a successful move. If the number of collisions encountered before making a successful move is near this value, the search resolution will stay very close to the previously used resolution. However, if significantly more collisions are encountered, the resolution value drops quickly, down to the floor value set by the user, in the *conservative zone*. If relatively few collisions are encountered before making a successful move, the next search resolution may grow by a factor determined by the curve between x=0 and $x=x_s$. Since we do not know the distribution of the C–space shapes and clutteredness, x_s can only be determined through empirical means. By moving this cutoff threshold between the two zones, one can change the behavior of the refinement model so as to effect a more aggressive or more conservative policy. Notice that the maximum step size factor f_{max} and the minimum step size factor f_{min} are the other parameters in the model. With these parameters, we set also a ceiling and a floor for s_{i+1} so that the user may enforce control over the search resolution. For instance, the maximum step size should not exceed the minimum obstacle dimension so that the moving object would not jump through an obstacle from one configuration to the next. These parameters are case dependent. In our experiments, we have used $f_{min} = 1 / f_{max}$ and $f_{max} = 2$.

3.2 Biased Paths

There are various ways to achieve bias in searching for a path. An obvious way is to block all undesirable passages by artificial obstacles. However, in 3D environments, it may not be intuitive for the user to specify such obstacles. The user needs to know the locations and dimensions of these passages in order to place artificial obstacles with reasonable sizes. This process may be tedious when the number of these passages are large. In addition, it is difficult to verify that these artificial obstacles will not prevent the planner from finding an otherwise feasible path, e.g., a path that requires the use of parts of an undesirable passage to adjust the orientation of the object.

In our system, we developed a different approach in which the user specifies a *constraint volume* by placing a set of spheres of various diameters (user specified) at critical branching points in the workspace where the user wants to prefer one direction to the others. These spheres are then linked together with cones smoothly to form a volume. The exterior of the volume is treated differently than obstacles. While the original obstacles are still there, confining the motion of an LRU, this volume places an additional constraint on the motion of the LRU. More specifically, a certain set of points (called *constraint points*) is constrained to stay inside the volume, while the rest of the concerned LRU may leave the volume to, for instance, make orientational adjustments. These constraint points may be critical surface landmarks of an LRU, its centroid, or some other references. Note that a given constraint through a set of user–specified spheres does not constitute a single initial path. It merely signifies that the selected constraint points will trace out a path, during the search, that lies inside the corresponding constraint volume. In fact in cases where fewer than three (non–colinear) constraint points are used, indeed at some configurations the LRU may be in collision with obstacles. Furthermore, such a constraint volume is allowed to overlap with obstacles, thus making its selection much less demanding of the user. Figure 4 illustrates such a constraint volume for the case shown in Figure 2 to bias the search to find path (b). The algorithm used in the RPP to build heuristic potential fields is modified to account for this constraint volume. The potential values inside the constraint volume are lower than those outside such that during the search process configurations that are inside or closer to the volume are preferred to those that are outside or farther. This effect is achieved by generating potential field in two phases: inside the constraint volume followed by outside the volume.

This constraining mechanism is empirical but it provides an easy and flexible way for the user to specify a bias in the path searching process. It can also be used when the user observes or predicts that the heuristic potential fields used in the RPP can not provide an effective guidance for the search.

4 Results and Discussion

We have applied the system to several maintainability study problems. Figure 5 presents a case where the accessibility of an irregularly shaped LRU (Figure 5(a)) inside an assembly. Figures 5(b) through 5(f) show a series of snapshots taken as the LRU moves along the computed path. The difficulty of this case lies in the fact that the LRU is so irregularly shaped. Any approximation using bounding box or spheres would not be efficient. Also, given the shape, the computation of a C–space representation from the environment is impractical. The path took several hours of computation time on an SGI RealityEngine computer (see the last section for more details on software as well as hardware used).

Figure 6 includes a retrospective study where a set of previously studied design were made available to us. In the design, the LRU is surrounded by several pipes as shown in Figure 6(a). The manual approach the designers used previously had determined that some of the pipes were in the way. Those pipes were consequently redesigned. To illustrate the difference, we show both sets of pipes in the figure (see pipes that split and then join – one of the branches belongs to the old design; where they fuse together is the part that did not get changed in the new design). We applied our automated system on both the old design, and the new one. First, we confirmed that a path found using the new design has collision configurations with the old design. Hence, those changes were necessary. For instance, Figures 6(c) and 6(d) show two collision situations where the LRU was in collision with the pipes from the old design, while clearing the new pipes. However, we have also found, that some of the redesign were indeed unnecessary. For example, the section of pipe near the top left hand corner of Figure 6(d) (right below the letter d) did not pose any collision threats to the LRU along the path found. Through this particular study, we show that not only is the automated system faster relative to the manual approach, it is a more accurate tool as well. Previously, when the designer is in doubt, he would tend to be more conservative in his design in order to leave room for error. Now, with the automated system, he will be provided with accurate information as to where exactly changes should be made.

To study the effectiveness of the adaptive multi–resolution search mechanism, we applied the system to some customized manufacturing cases, where specially designed parts are tested for manufacturability. In Figure 7, we show results from one such case, where a set of closely fitted matting parts is designed to be installed together. Specifically, an insert part is designed to be inserted into a twisted cavity without deformation (see Figure 7(a)). The design allowed only 10 mill clearance between them. Since both the insert and the cavity are twisted in two dimensions, it is not intuitive to know, given the curvature, that the insert can be either installed inside the cavity or taken out. With our automated system, we started with a 5 mill search resolution, with which the system was able to move the insert out 1/4 of the length after a few hours of computation. Then, as more and more collisions are detected for each move, the search resolution was reduced automatically according to the model presented in the last section. Eventually, at 1 mill search resolution, the insert came out 1/3 of the length and was determined to be totally stuck (see Figure 7(b)). The design was thus turned down and a rework was ordered. Note the perspective effect of the display in Figure 7. The cross section of the parts at both ends are of the same dimension. So the reason for it to get stuck is the effect of the twisting in the middle part.

We used several cases to study the constraint volume approach. Figure 8 shows a case where the LRU is located in a cluttered environment under a set of pipes (see Figures 8(a) and 8(e)). There is an opening between the pipes that is wide enough for the object to get out when its orientation is aligned correctly. The LRU in its initial configuration is orthogonal to this opening (as shown in Figure 8(e)), and the study is basically to show whether there is enough room for the object to make such an orientation change to get out through such a designed opening. Our study shows that the initial configuration favors a path (Figures 8(h) through 8(j)) that does not use the opening. Figures 8(k) and 8(l) show close up views of two configurations along the path. In order to force the LRU to come out through the designed opening, we specified two spheres: one at the initial configuration and one right on the other side of the obstacle pipes, forming a constraint volume that cuts off the undesirable path. Figures 8(b) through 8(d) show the path found after applying this constraint volume. Figures 8(f) and 8(g) provide a close up view of two configurations along that path.

Overall, the cases presented here are difficult ones for traditional, CAD–based move–and–detect–collision type of manual maintainability systems. We have realized, by using the automated system presented here, a tremendous productivity gain. In the past with the manual approach a study would take several days of tedious work. Now, typically, it takes a few minutes to several hours to generate a path if it can find one. With the automated system available, designers can also perform several studies simultaneously. Depending on how complex the study is, the productivity gain ranges from several factors to an order of magnitude.

But this system is not without problems. In general, we realize that most of the time is still spent on collision checking. In certain cases (that are not included in this paper) we experienced pronounced effect from the probabilistic nature of the planner. Some of our studies are non–conclusive after several days (one particular assembly took more than a week (on the SGI workstation mentioned above) and was eventually terminated manually). To address this problem, we are planning on exploring other techniques such as the randomized roadmap algorithm in [Kavraki94]. In that direction, we will try to emphasize the deterministic behavior of a planner.

5 Conclusions

We developed a first–known practical automated assembly maintainability study system by incorporating results from research in robotics motion planning. In this paper, we present a dynamic, adaptive multi–resolution model in the system. This search resolution adaptation model overcomes the difficulties fixed resolution motion planners experience. In addition, a bias mechanism was developed to facilitate specification of access path preference that is required in typical maintainability and manufacturability studies. A number of industrial application studies are presented in the paper that use the system to show that the automated system not only solves otherwise–difficult problems but also lends significant productivity improvement over existing computer–assisted manual approach. We also show that such a system is an effective tool for custom manufacturing where manufacturability can be tested before parts are made. With the results, we are confident that this automated maintainability system points to a new direction of applications of research in robotic motion planning. We hope that it will serve to provide additional driving force for research in motion planning in general.

Acknowledgements

The automated system is encapsulated in *ProductVision*, a commercial software developed at GE Corporate R&D to produce the results shown in this paper. Briefly, ProductVision provides a visual environment on UNIX workstations to import geometrical models (from major CAD packages) to generate access paths to test assembly feasibility using the motion planner described in this paper. It also creates a swept surface to outline the physical space required of a given path for the given LRU. Results can be presented using computer animation in the software as shown in the accompanying Video Proceedings. The reported applications have been performed on SGI's RealityEngine, with two R4400 processors at 150 MHZ and 128 MBytes of main memory.

References

[Barraquand90] Barraquand, J. and Latombe, J.–C. ''A Monte–Carlo Algorithm for Path Planning with Many Degrees of Freedom,'' Proc. IEEE Int'l Conf. on Robotics and Automation, 1712–1717 (1990).

[Donald87] Donald, B. ''A Search Algorithm for Motion Planning with Six Degrees of Freedom,'' Artificial Intelligence, 31. 295–353 (1987).

[Gilbert85] Gilbert E.G. and Johnson D. W. ''Distance Functions and Their Application to Robot Path Planning in the Presence of Obstacles,'' IEEE Trans. on Robotics and Automation, RA–1. 21–30 (1985).

[Hwang92] Hwang, Y. K. Ahuja, N. ''Gross Motion Planning – A Survey,'' ACM Computing Surveys, 24. 220–291 (1992).

[Kavraki94] Kavraki, L. Latombe, J.–C. ''Randomized Preprocessing of Configuration Space for Fast Path Planning,'' Proc. IEEE Int'l Conf. on Robotics and Automation, preprint. (1994).

[Latombe91] Latombe, J.–C. Robot Motion Planning, Kluwer Academic Publishers, Boston, 1991.

[Lozano79] Lozano–Perez, T. and Wesley, M., ''An Algorithm for Planning Collision–free Paths Among Polyhedral Obstacles,'' Comm. of ACM, 22. 560–570 (1979).

[Lozano83] Lozano–Perez, T. ''Spatial Planning: A Configuration Space Approach,'' IEEE Trans. on Computers, C–32. 108–120 (1983).

[Lozano91] Lozano–Perez, T and O'Donnel, P. ''Parallel Robot Motion Planning,'' Proc. IEEE Int'l Conf. on Robotics and Automation, 1000–1007, 1991.

[Lozano93] Lozano–Perez, T and Wilson, R. ''Assembly Sequencing for Arbitrary Motions,'' Proc. IEEE Int'l Conf. on Robotics and Automation, 1993.

[Quinlan94] Quinlan, S. ''Efficient Distance Computation between Non–convex Ojbects,'' Proc. IEEE Int'l Conf. on Robotics and Automation, 1994.

[Schwartz83b] Schwartz, J. T. and Sharir, M., ''On the Piano Movers' Problem: II. Advances in Applied Mathematics, 4. 298–351 (1983).

[Wilson92a] Wilson, R. and Latombe, J.–C. ''On the Qualitative Structure of a Mechanical Assembly,'' Proc. IEEE Int'l Conf. on Robotics and Automation, 1992.

[Wilson92b] Wilson, R. and Matsui, T. ''Partitioning an Assembly for Infinitesimal Motions in Translation and Rotation,'' Proc. IEEE Int'l Conf. on Intelligent Robotics and Systems, 1992.