Path planning using potential field representation

Yong Koo Hwang Narendra Ahuja

The University of Illinois, Coordinated Science Laboratory
1101 w. Springfield Ave. Urbana, Illinois 61801

ABSTRACT

Finding a safe, smooth, and efficient path to move an object through obstacles is necessary for object manipulation in robotics and automation. This paper presents an approach to two-dimensional as well as three-dimensional findpath problems that divides the problem into two steps. First, rough paths are found based only on topological information. This is accomplished by assigning to each obstacle an artificial potential similar to the electrostatic potential to prevent the moving object from colliding with the obstacles, and then locating minimum potential valleys. Second, the paths defined by the minimum potential valleys are modified to obtain an optimal collision-free path and orientations of the moving object along the path. Three algorithms are given to accomplish this second step. The first algorithm simply minimizes a weighted sum of the path length and the total potential experienced by the moving object along the path. This algorithm solves only "easy" problems where the free space between the obstacles is wide. The other two algorithms are developed to handle the problems in which intelligent maneuvering of the moving object among tightly packed obstacles is necessary. These three algorithms based on potential fields are nearly complete in scope, and solve a large variety of problems.

1. INTRODUCTION

This paper presents a solution to the findpath problem defined as follows. Given a space littered with obstacles and a moving object (MO), find a continuous path and orientation connecting the starting position/orientation and the goal position/orientation of MO. Findpath algorithms are classified as either being complete or approximate. Complete algorithms are aimed at guaranteeing a solution if there is one or proving that there is no solution. These algorithms are useful for very hard findpath problems, e.g., when the space is cluttered densely with obstacles and MO is bulky, and intelligent maneuvering is necessary to move an object to the destination. The complexity of the findpath problem has been found to be polynomial in the number of obstacles. At present, there appear to be two complete algorithms. The only known complete polynomial algorithm for the classical mover's problem is that of Schwartz and Sharir [7]. Unfortunately, this algorithm takes $O(n^{(2^{2^{d}})})$ time where n is the number of edges of obstacles and d is the number of degrees of freedom. For the classical mover's problem in three dimensions, where d=6 corresponding to the 3 translational and 3 rotational degrees of freedom, this becomes $O(n^{4096})$. Configuration space [2] approach is the other complete algorithm, and the only complete algorithm that appears to have been implemented [3].

Approximate algorithms make simplifications at the representation level in order to generate fast algorithms. They are attractive for the problems where the free space between obstacles is wide, and tight maneuvering is not necessary. Brooks [1] represents the free space in two dimensions as a union of generalized cylinders, and MO is moved along the spines of the cylinders. Maddila [6] reports an algorithm that moves a line segment among iso-oriented rectangles. Singh and Wagh [8] have studied the problem of moving a point object among iso-oriented rectangles by representing the free space with maximal convex regions. Herman [4] developed a fast algorithm for three-dimensional robot motion planning using octree representation of the free space.

Khatib [5] uses an artificial potential for repulsion between objects to avoid imminent collisions among them. This algorithm is aimed at the local, short term avoidance of obstacles in real time for moving robot arms rather than planning good global paths. This paper describes algorithms that use an artificial potential to solve the findpath problem.

1.1. Overview of potential field approach

This paper presents findpath algorithms in two and three dimensions (2D and 3D). Obstacles are assumed to be polygons or polyhedra. We describe the topological structure of the problem space by a scalar potential field. Imagine that all the obstacles are composed of positively charged matter. If MO is also positively charged, the obstacle avoidance problem is resolved by the repulsive force. This force can be calculated as the negative gradient of a potential field.

Our algorithm divides the problem into two stages. First, topologically distinct paths are found from the valleys of potential minima. The best candidate path which is mostly likely to yield a collision-free path of the minimum length is selected from the minimum potential valleys (MPV). Second, three findpath algorithms modify the best candidate path to

obtain a collision-free path and smoothly changing orientations along the path. These algorithms solve problems of different levels of difficulty. First, the parallel optimization algorithm (POA) employs a numerical method to minimize a weighted sum of the path length and the total potential experienced by MO along the path, and solves the simplest of the findpath problems. The second is the serial optimization algorithm (SOA), intended for problems where MO must maneuver through tight spaces between obstacles. SOA identifies narrow regions along the candidate path, finds collision-free configurations of MO in these regions, and tries to connect the start and goal configuration through a sequence of collision-free configurations, one from each narrow region. The sidetracking algorithm (STA) is for the problems where in addition to tight maneuvering, MO must also exploit nonlocal geometry of the free space. This algorithm is useful in problems where the free space is so narrow that sidetracking, or brief excursions away from the candidate path, are necessary to change the orientation of MO. These three algorithms are tested on many examples, and they seem to solve a variety of findpath problems.

Section 2 introduces potential field representation. Sections 3-4 describe three findpath algorithms and their performances on various findpath problems.

2. DESCRIPTION OF FREE SPACE WITH POTENTIAL FIELD

The potential field serves two purposes in the development of findpath algorithms. First, it is used to compute repulsive force which is used to avoid collisions. Second, and more important, the potential field brings out the topological structure of the free space between the obstacles in form of the minimum valleys of the potential field. It serves as an analog representation of object shapes, and what object should be paid immediate attention to while moving through a certain part of the free space. Such an analog representation circumvents the combinatorial complexity characteristic of representations of the free space in terms of shape primitives, wherein collisions need to be often checked against individual primitives separately.

2.1. Selection of the potential function

Since an expression of the electrostatic potential is not available for polytopes, a new function is developed. If a polytope is represented as $\cap(x \mid g_i(x) \leq 0)$ where $g_i(x) = 0$ are the boundary equations, then $p(x) = [\sum g_i(x) + \mid g_i(x) \mid]^{-1}$ has similar properties to those of the electrostatic potential.

2.2. Generation of MPV

It is necessary to classify all possible paths into a finite number of topologically distinct paths, and examine only these for the possibility of yielding a collision-free path. MPV lie as far away from the obstacles as possible, and are thus good candidates for topological paths. MPV are generated in the following way. Beginning from the start location, the largest circle (sphere) contained in the free space is drawn. The points on the circle (sphere) with locally minimum potentials are labeled as the neighboring points of the start location. Neighboring points of the newly found points are found in a similar manner. This process is continued until all MPV are covered with points of locally minimum potentials.

2.3. Selection of the best candidate path

After MPV are generated, a path from the start to the goal position has to be selected. The selected path has to be optimal in the sense that the path should be of minimum length and least likely to cause collisions between the MO and the obstacles. A cost function is used for each branch in MPV to denote the length and the difficulty of each branch to be traversed by MO. The cost function is proportional to the length of the branch and to the potential values along the branch. Dynamic programming is used to compute the minimum cost path from the start to goal location.

3. FINDPATH ALGORITHMS

3.1. Parallel optimization algorithm

POA tries to find a path that minimizes a weighted sum of the path length and the total potential experienced by MO along the path. Minimizing the potential on MO favors object motion away from obstacles to avoid collisions, whereas minimization of the path length prevents MO from wandering in wide parts of the free space. The objective functional to be minimized is

$$J = \int_{x_0,\theta_0}^{x_f,\theta_f} [(1+bP(x,\theta))(dx)^2 + a(d\theta)^2]. \qquad (1)$$

where $P(x,\theta)$ is the total potential experienced by MO at position x and orientation θ. A numerical method must be

employed to minimize J. The first-order gradient method is used since it is robust to initial guesses and converges rapidly. The best candidate path selected from MPV serves as initial guesses to the numerical method.

3.2. Serial optimization algorithm

When an intelligent maneuvering of MO is required to find a solution to a findpath problem, an algorithm which utilizes the shape of MO and the local geometry of the free space is needed. POA minimizes the weighted sum of the path length and the potential experienced by MO over the entire path. POA may or may not converge to a collision-free path. In contrast, SOA first examines the initial path and identifies the parts of the path that may cause collisions. Then minimization of the potential on MO is carried out locally in these likely collision regions. Since the minimization is done locally, SOA can take into consideration the shape of MO and the local geometry of the free space much more effectively.

SOA divides the task into four steps. First, MO is moved along the candidate path, and the regions where collisions occur are identified. Then collision-free configurations (CFC's) of MO in the collision regions are found. The next task is to connect a CFC of a collision region with a CFC of an adjacent collision region. If the start configuration and the goal configuration can be connected through a sequence of CFC's of the collision regions, then a solution of the findpath problem follows. If no such sequence is found, then the candidate path is assumed not to lead to a solution and the next best path from MPV are used. The steps of SOA are illustrated in Fig. 1.

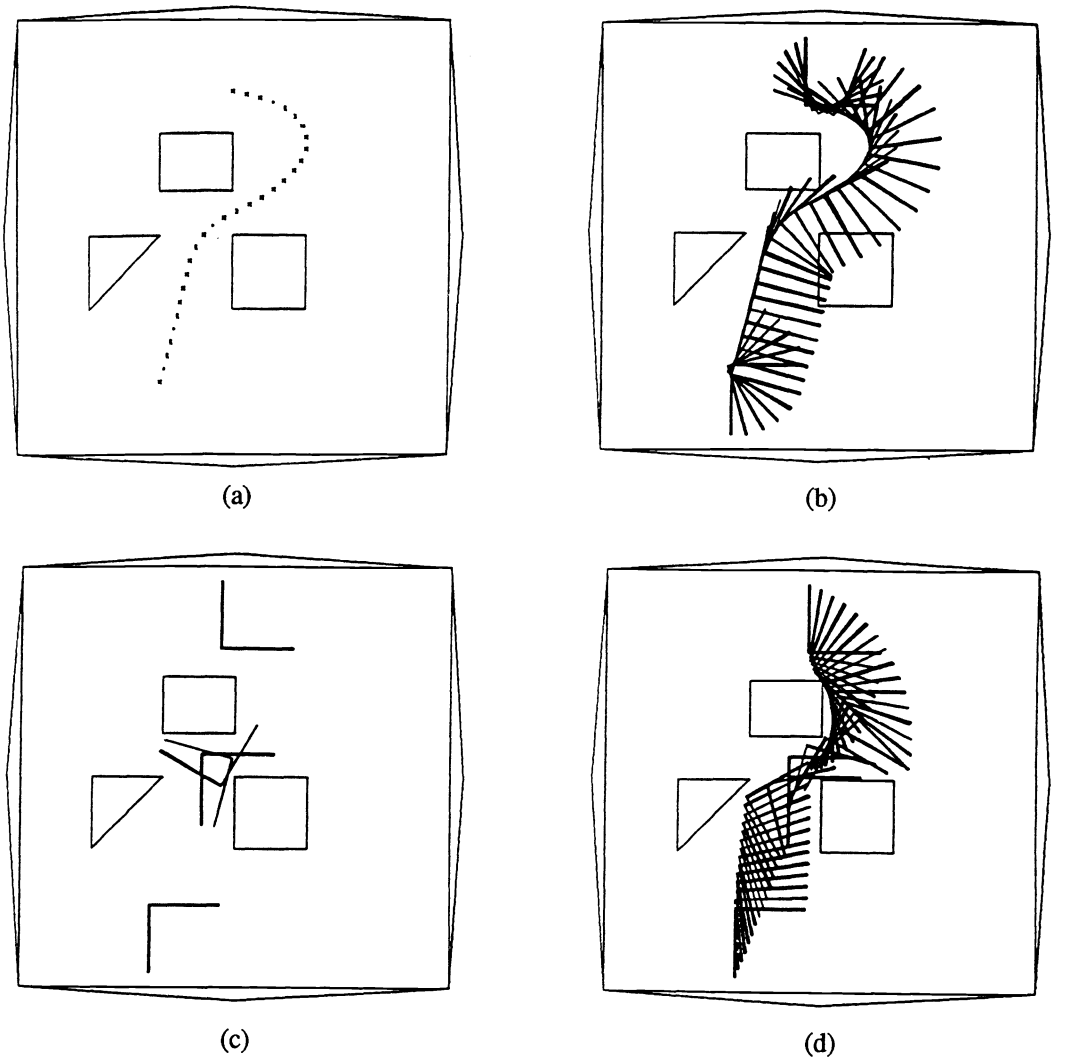


Fig. 1. (a) The initial candidate path.
(b) The initial movement of MO.
(c) Collision-free orientations of MO in a collision region.
(d) The result of SOA.

3.2.1. Feasible configurations in collision regions There are usually an infinite number of CFC's in a collision region, and they have to be grouped into a finite number of topologically distinct configurations. This is achieved by selecting only those configurations which yield locally minimum potentials on MO. In 2D, MO is rotated at discrete angular intervals, and the orientations of locally minimum potentials are stored. In 3D, the orientation space is partitioned into a finite number of cells, and the orientations with locally minimum potentials within each cell are stored. Then for each stored orientation, MO is allowed to translate to further lower the potential on MO. The results are topologically distinct configurations in which MO has locally minimum potential in the collision region. Some of these configurations may still make MO collide with the obstacles, and only those without collisions are considered as CFC's.

3.2.2. Connecting feasible configurations SOA tries to connect two adjacent configurations of MO by making MO move from one configuration to the other. In doing so, the initially chosen path from MPV must provided MO with the general direction to move. The MO follows the initial path while adjusting the position and the orientation a little at a time to avoid the obstacles. After MO moves one step, the connecting algorithm checks whether the two configurations can be connected by moving MO in a straight line and changing the orientation at a uniform rate. If this succeeds, then the two feasible configurations are connected before the algorithm steps through the whole path between the two configurations.

This process of advancing one step and relaxing the potential makes it easier for MO to move out of a high potential region into open space, rather than to converge to a configuration in a high potential region from a low potential region. It may happen that MO collides with an obstacle while moving from one of the two configurations being connected. Then the connecting algorithm moves MO only from the other configuration until the two configurations are connected. The connecting algorithm signals failure when MO collides with obstacles from both directions, or when there are no nodes between the two configurations being connected.

3.3. Sidetracking algorithm

This section deals with the hardest class of findpath problems that we have considered. These problems require intelligent nonlocal maneuvering of MO. That is, MO may have to take into consideration the geometry of the free space far away from the current location of MO in order to solve the local findpath problem at the current location. Fig. 11 well illustrates such a situation. Even humans may exhibit noticeable delay in solving such findpath problems, possibly because sequential search of the space may be hard to avoid.

STA is best explained with the example in Fig. 2. Fig. 2(a) shows MPV, the start and goal configurations, and the collision-free configurations (CFC1, CFC2) in the collision region, which is at the sharp corner. The graphical representation of the problem is shown in Fig. 2(b). The solid edges between two nodes mean that the two nodes (or configurations) are connected by SOA, and the dotted edges mean that the nodes cannot be connected. SOA tries to connect the nodes in Fig. 2(b) in the order of the numbers labeling the edges. SOA signals failure after trying to connect the start node and CFC2. It can be easily seen that the solution to this findpath problem is to connect CFC1 and CFC2, and then connect CFC2 and the goal configuration. Connecting CFC1 and CFC2 requires to move MO from these configurations toward the

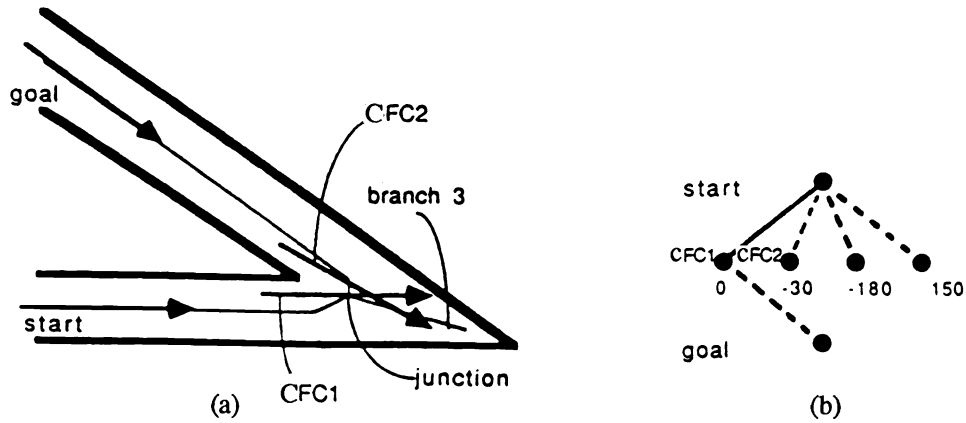


Fig. 2. A findpath problem SOA cannot solve.
(a) The start and goal configurations of MO (the arrow) is shown along with the collision-free configurations in the collision region. The thin lines denote MPV.
(b) The graphical representation of SOA.

dead end corner, following branch 3 of MPV in Fig. 2(a). Checking the connectivity of two CFC's in a collision region always involves sidetracking from the initial candidate path. If SOA is equipped with sidetracking capability to establish the connectivities among the CFC's in the same collision region, the hardest class of the findpath problems we have considered could be solved by the potential field algorithm.

The sidetracking algorithm (STA) works as follows. SOA is applied forward from the start node. When the forward SOA fails to find a solution, SOA is applied backwards from the goal node. Whenever the backward SOA connects the goal node to a CFC in a collision region containing a CFC connected to the start node, STA tries to connect the two CFC's in the collision region. STA moves MO from these two CFC's in the directions away from the collision region. If STA succeeds in connecting them, a solution is found. Otherwise, backward SOA is continued until the goal node is connected to another CFC in a collision region containing a CFC connected to the start node.

4. EXAMPLES

The three algorithms are tested on a variety of examples. Fig. 3 shows a bar moving object turning an L-shaped hallway. Although the initial path and orientation involve collisions, POA successfully found a solution. Fig. 4 demonstrates the limitation of POA. This problem will be solved by STA. A 3D example is shown in Fig. 5. POA results in a shortest path with smoothly changing orientations of MO.

The next three examples are solved by SOA. Fig. 6 shows a rigid foot of a robot coming out of a shoe. Fig. 7 shows how to make the word "CTA" into "CAT" by moving the letter "T" only. For the example in Fig. 8, SOA examines three different paths before finding a solution. Fig. 9 shows how to move a grand piano into the living room.

STA solves the problem of moving a bar around a sharp corner, as shown in Fig. 10. Fig. 11 shows an arc making two sidetrackings at the T-shaped junction. In Fig. 12, the chair is to be moved from one side of the desk to the other. The result of STA shows the chair sidetracking from one side of the desk to get the proper orientation.

Fig. 13 shows a case for which all three algorithms fail to generate a solution. MPV between three circular obstacles do not give a good clue as to how to move the τ-shape to an open space. It is crucial in our potential based algorithms that MPV give a good estimate of a solution path.

5. CONCLUSIONS

Although the findpath algorithms presented in this paper have not been proven to constitute a complete algorithm, they have been demonstrated to solve a large variety of findpath problems. The strongest point of this near-complete algorithms is the short computation time, especially in 2D. For all the 2D examples presented in this paper, it takes less than 5 minutes to generate the solutions on a SUN 260 computer. The 3D algorithms run in the order of tens of minutes, most of the time being spent on the generation of MPV. Once a candidate path is given, however, the 3D algorithms run in less than 10 minutes for most of the 3D examples.

Two aspects of the potential field based algorithm need to be improved. First, a more efficient way to compute the best candidate path is needed, especially for 3D algorithm. A faster way to compute the potential function will also speed up the algorithms since the potential field is used extensively in all phases of the algorithms. With these improvements, the potential field approach should become a powerful tool to solve findpath problems.

6. REFERENCES

[1] Rodney A. Brooks, "Solving the Findpath Problem by Good Representation of Free Space," *IEEE Trans. on Syst., Man, and Cybern.*, vol. SMC-13, pp. 190-197, March/April 1983.

[2] Rodney A. Brooks and Tomás Lozano-Pérez, "A Subdivision Algorithm in Configuration Space for Findpath with Rotation," *Int. Joint Conf. on Artificial Intelligence*, Karlsruhe, Germany, 1983.

[3] Bruce Donald, "Motion Planning with Six Degrees of Freedom," Massachusetts Institute of Technology Artificial Intelligence Laboratory, AI-TR-791, 1984.

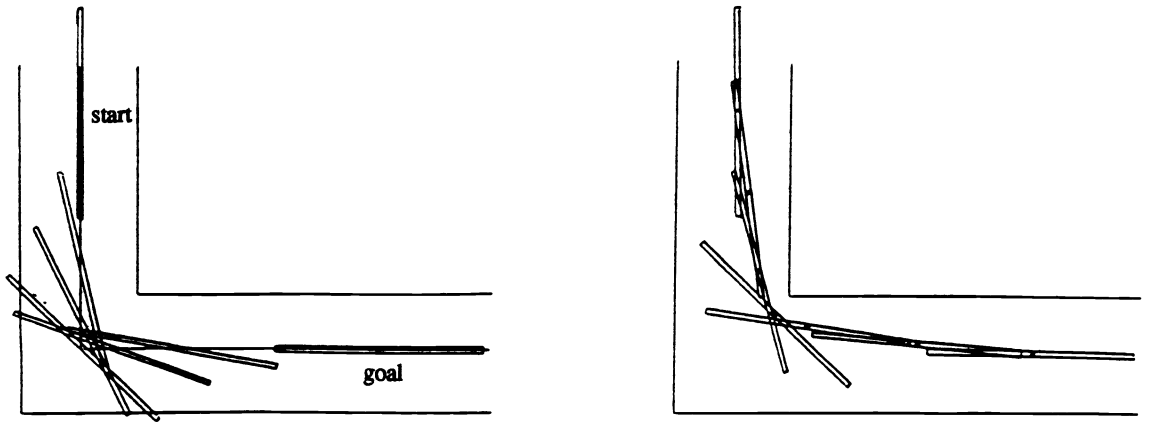
[4] Martin Herman, "Fast, Three-Dimensional, Collision-Free Motion Planning," *Proc. of IEEE Int. Conf. on Robotics and Automation*, San Francisco, California, April 1986.

[5] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," *Proc. of IEEE Int. Conf. on Robotics and Automation*, St. Louis, Missouri, March 1985.

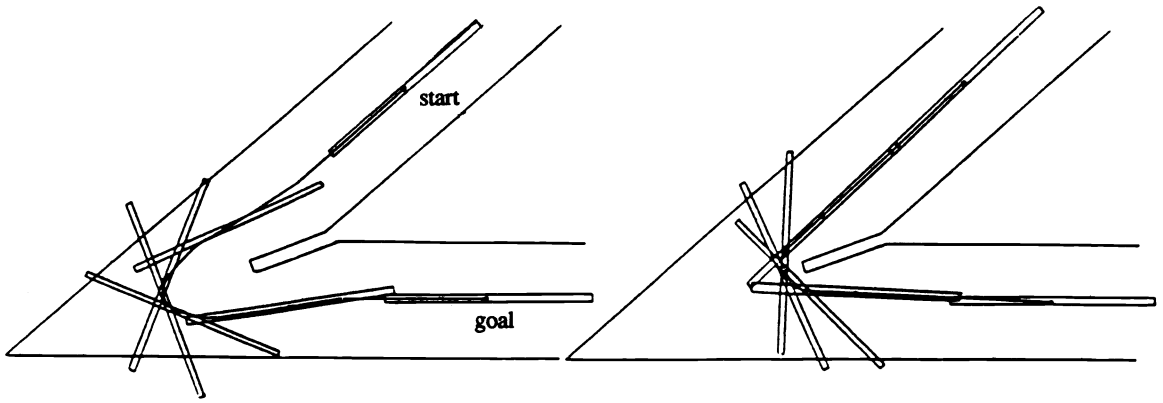
[6] Sanjeev R. Maddila, "Decomposition Algorithm for Moving a Ladder Among Rectangular Obstacles," *Proc. of IEEE Int. Conf. on Robotics and Automation*, San Francisco, California, April 1986.

[7] Jacob T. Schwartz and Micha Sharir, "On the Piano Movers' Problem: I. The Case of a Two-Dimensional Rigid Polygonal Body Moving Amidst Polygonal Barriers," *Communications on Pure and Applied Mathematics*, vol. 34, pp. 345-398, 1983.

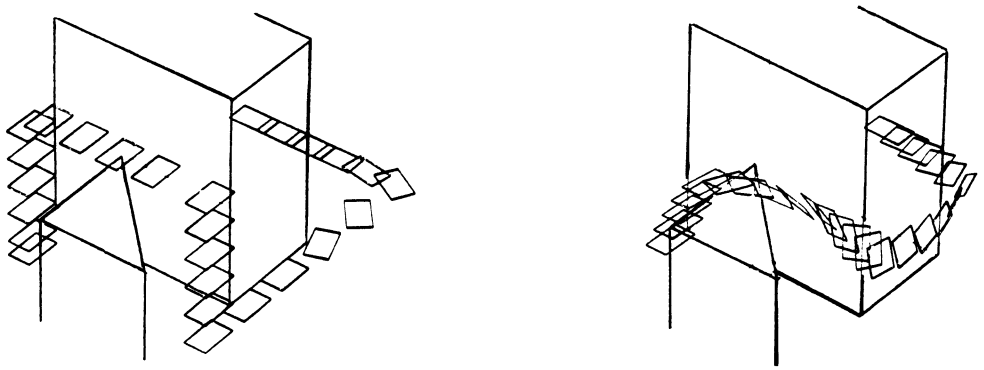
[8] Sanjiv Singh and Meghanad D. Wagh, "Robot Path Planning Using Intersecting Convex Shapes," *Proc. of IEEE Int. Conf. on Robotics and Automation*, San Francisco, California, April 1986.



(a) Initial guesses of the path/orientation involve collisions. (b) A collision-free path/orientation found by POA.
Fig. 3 Problem of moving a bar around a corner of an L-shaped hallway.



(a) Initial guesses of the path and orientation. (b) The path found by POA. POA fails to find a solution.
Fig. 4 A problem requiring intelligent maneuvering of the moving object at the corner.



(a) Initial guess of a solution. (b) The result of POA.
Fig. 5 Problem of moving a board from behind the tall box to the front, and to the left of a triangular table.

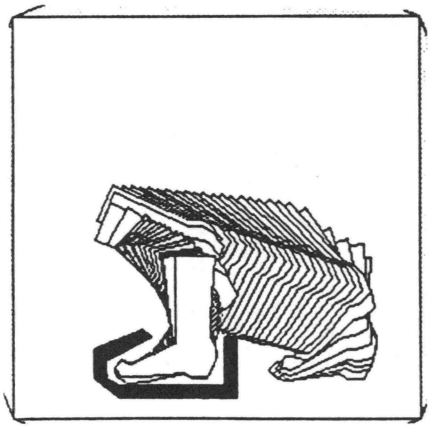


Fig. 6 Problem of taking off a shoe.

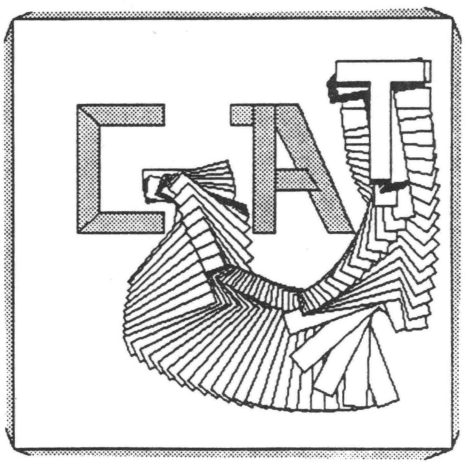
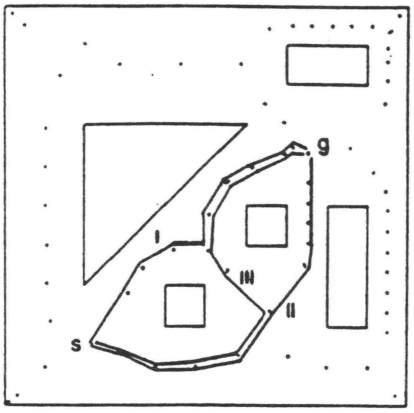
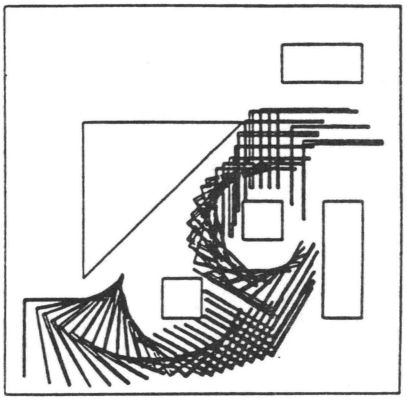


Fig. 7 Moving a "T" without touching other letters.





(a) Three topologically distinct paths examined by SOA. (b) The solution found by SOA.
Fig. 8 Problem of moving an L through polygons.

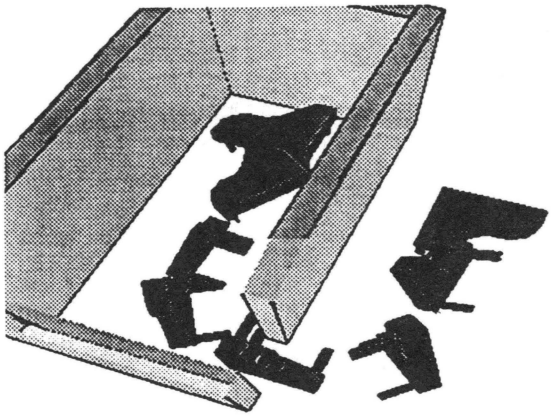


Fig. 9 Moving a grand piano into a living room.

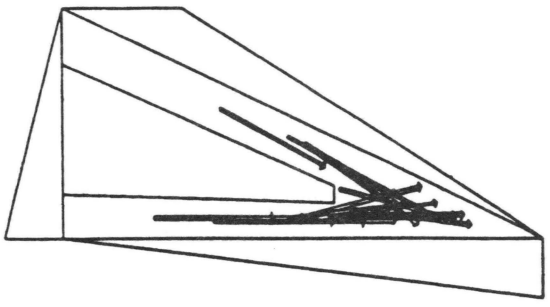


Fig. 10 A bar turning a sharp corner

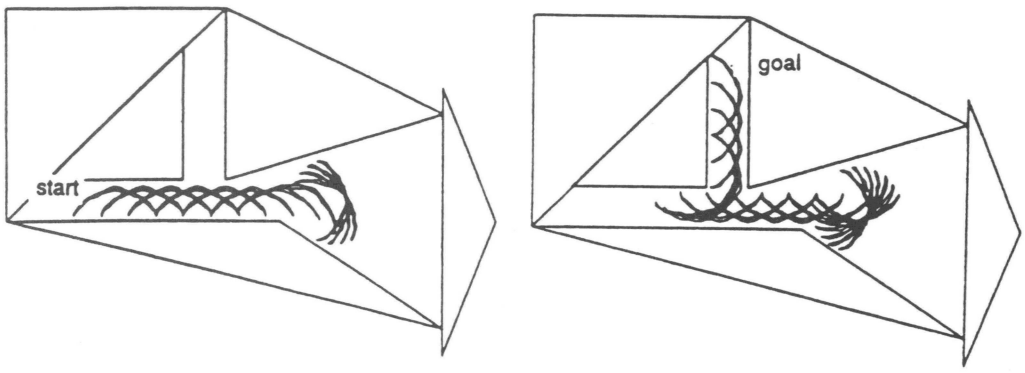


Fig. 11 One of the hardest findpath problems. This example requires two sidetrackings to find a collision-free path and orientation.

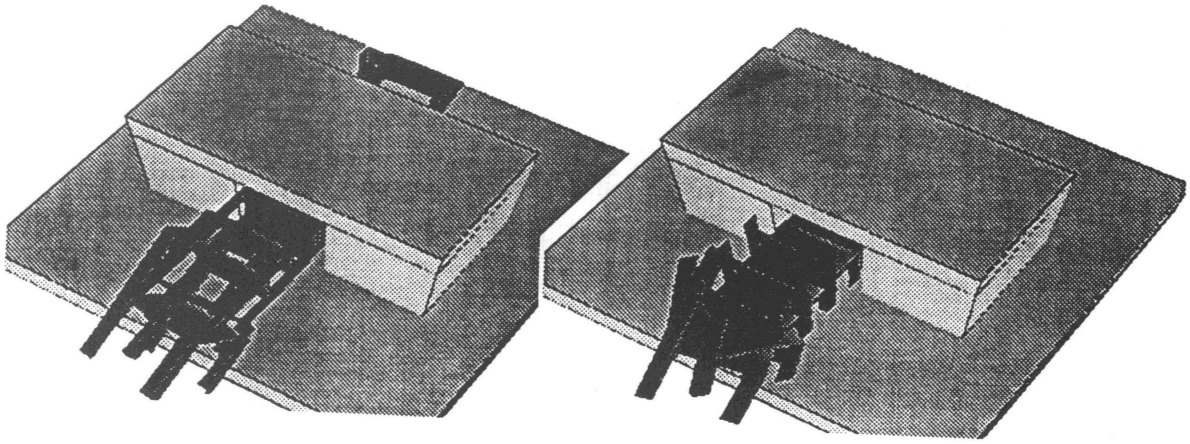


Fig. 12 Problem of moving a chair from one side of a desk to the other side.

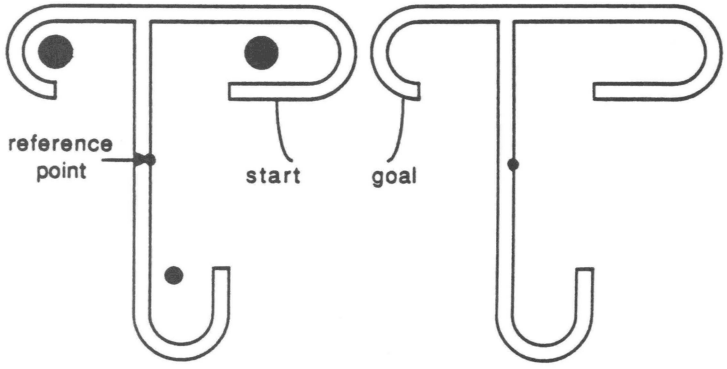


Fig. 13 The potential field algorithms fail to move the τ-shaped MO among the three circular obstacles out to an open space at the right.