Robot Reachability Problem: a Nonlinear Optimization Approach

ZHIYUAN YING and S. SITHARAMA IYENGAR

Department of Computer Science, Louisiana State University, Baton Rouge, LA 70803, U.S.A.

(Received: 4 February 1993; in final form: 3 May 1993)

Abstract. The reachability of a robot manipulator to a target is defined as its ability to move its joints and links in free space in order for its hand to reach the given target. This paper presents a way of testing the reachability of a robot to given target. The target could be a three dimensional object represented by a cuboid, a line or merely a point.

The reachability test problem is transformed into a nonlinear optimization problem, which is solved by using the Tunneling Algorithm [1-3].

The paradigm of the Tunneling Algorithm is described in detail. Several examples of testing the reachability of two robots to given targets are presented and the results are compared with that of the existing RGRG algorithm [5]. The results of comparisons show that the Tunneling Algorithm is better than the RGRG algorithm. It can always obtain the correct answers of testing, and it is effective and suitable to solve the reachability test problem.

Key words. Tunneling algorithm, reachability test, robot manipulator.

1. Introduction

Today, the robotics and automation community are being swept by broad, pervasive technological demands. With the increasing use of robots in remote unstructured environments, it has become necessary to identify various computation problems that need to be solved quickly in real time. One such problem, which has been of considerable interest, is the robot reachability in a given bounded workspace.

In this paper, an approach is considered to study a class of problems in robot reachability for three dimensional workspace domains.

The organization of the paper is as follows: Section 2 gives a definition of the reachability test problem, an example, and an analysis. Section 3 reviews the related research and the contribution of this paper. Section 4 presents the mathematical formulation of the reachability test problem. Section 5 briefly reviews the existing optimization methods. Section 6 reveals the details of the tunneling algorithm. The results of the tunneling algorithm to the robot reachability test problem, the comparison of the tunneling algorithm and the existing method are given in Section 7. In Section 8, a conclusion is given.

^{*} This project is partially supported by a grant from Martin Marietta (ORNL) 19x-55902V. Also this project is partially funded by ONR grant N00014-94-1-0343.

2. What is the Reachability Test Problem?

2.1. DEFINITION

The reachability of a robot manipulator to a target is defined as its ability to move its joints and links in free space in order for its hand to reach the given target. Thus, the Reachability Test is to test whether the robot has the ability to move its joints for its hand to reach the given target.

When the target is a point, the definition is quite clear. When the target is a spatial object then the points on the object are considered. If there is at least one point on the object which can be reached by the robot hand, the object is said to be reachable by the robot.

2.2. A SIMPLE ILLUSTRATION

Consider the following example (see Figure 1), place the hand as close to the reference point (x_0, y_0) as possible. The robot has two links, a and b. Obviously, when the two joint angles change, the end-effector (hand) of the robot can move around and reach many spatial points. The collection of all the points that the robot can reach is called reachable workspace of the robot. With this definition, the problem of the reachability test of the robot to a given point (x_0, y_0) is actually a problem of testing whether or not the given point is within the robot reachable workspace.

3. Previous Work and the Scope of this Paper

There is a wealth of literature on robot workspace analysis and synthesis, many authors have investigated reachable workspace related problems. Tsai and Soni [6], investigated the accessible region of planner two-three link manipulators with pinjoints. Lee and Yang [7, 8] presented an algorithm for detecting the boundary of the



Fig. 1. The 2R planar robot and its workspace with $0^{\circ} \leq q_1 \leq 90^{\circ}$ and $0^{\circ} \leq q_2 \leq 90^{\circ}$.

workspace and gave the criteria for detecting the holes and voids in the workspace. Sugimoto and Duffy [9] developed an algorithm for calculating the extreme distances reached by a robot hand. Rastegar and Deravi [10] presented a general method to determine the workspace and its subspace with different number of configurations. Hsu and Kohli [11] provided a method for conducting the workspace analysis to determine the boundaries of the workspace and to determine the voids and holes inside the workspace.

Many of their methods can be applied to the reachability test problem, but most of the methods deal with a special class of robots (i.e. three link, simple robot) or a particular robot. In this paper, the reachability test is formulated as a nonlinear optimization problem so that the method can be applied to any type of robot manipulators provided that the robot's forward homogenous transformation and the first and second order derivatives of the function exist. The Tunneling Algorithm, which is easy to apply and guaranteed to converge to the global minimum, is used to solve the nonlinear programming problem.

4. The Mathematical Formulation of the Reachability Test Problem

In order to formulate the problem, let us consider a general case. Assume there is a robot manipulator, which has n links and n joint angles and moves in a three dimensional workspace. The relationship between the manipulator joint coordinates and end-effector's Cartesian coordinates is given by:

$$P_x = f_x(q_1, q_2, \dots, q_n) \tag{4.1}$$

$$P_y = f_y(q_1, q_2, \dots, q_n) \tag{4.2}$$

$$P_z = f_z(q_1, q_2, \dots, q_n)$$
(4.3)

where (P_x, P_y, P_z) is the Cartesian coordinate of the robot end-effector, q_1, q_2, \ldots, q_n are its *n* joint variables, that is, joint angles. When the joint angles change, P_x, P_y, P_z will have different values, a set of (q_1, q_2, \ldots, q_n) corresponds to a unique set of (P_x, P_y, P_z) . f_x, f_y, f_z are continuous single value function mapping (q_1, q_2, \ldots, q_n) to (P_x, P_y, P_z) .

The problem whether a given point (X, Y, Z) is within or out of the manipulator reachable workspace may be solved by formulating the following problem, which is a set of nonlinear equations:

PROBLEM 4.1. Solving equations:

$$X = f_x(q_1, q_2, \dots, q_n)$$
(4.4)

 $Y = f_y(q_1, q_2, \dots, q_n)$ (4.5)

$$Z = f_z(q_1, q_2, \dots, q_n) \tag{4.6}$$

subject to :
$$q_{i\min} \leqslant q_i \leqslant q_{i\max}$$
 (4.7)

where f_x, f_y, f_z are identical to the functions given by equations (4.1)–(4.2); $q_{i\min}$ and $q_{i\max}$, i = 1, ..., n are the lower and upper bounds of the manipulator joint variables.

The left-hand side of the equations is the given point, while the right-hand side of the equations is the point that the robot can reach. Problem 4.1 is to find a set of (q_1, q_2, \ldots, q_n) which satisfies equations (4.4)–(4.6) and the constraint conditions (4.7). If there is at least one solution for the above problem, then the given point (X, Y, Z) is within the reachable workspace of the manipulator; if no solution exists, (X, Y, Z) is out of the reachable workspace of the manipulator. This is similar to a reverse kinematic problem, but without orientation information. To look at it from a different perspective, let us consider the following facts:

The relation between the workspace of a robot and an object is equivalent to that between two spatial geometric bodies. There are three cases: inclusion, intersection and separation, corresponding to fully overlapping, partly overlapping and not overlapping respectively. To test whether an object is reachable by the robot is just to test whether the object and the workspace of the robot overlap each other.

When two objects A and B fully or partly overlap, we can find at least two points P_1 and P_2 , where

$$P_1 \in A, \qquad P_2 \in B \tag{4.8}$$

such that

$$\|P_1 - P_2\| = 0. (4.9)$$

When two object are separate, for arbitrary $P_1 \in A$, and $P_2 \in B$,

$$\min \|P_1 - P_2\| > 0. \tag{4.10}$$

Suppose the object C is described by the cuboid:

$$C: \begin{cases} x_{\min} \leqslant x \leqslant x_{\max} \\ y_{\min} \leqslant y \leqslant y_{\max} \\ z_{\min} \leqslant z \leqslant z_{\max} \end{cases}$$
(4.11)

and the reachable workspace of the robot is given by:

$$W: \begin{cases} P_x = f_x(q_1, q_2, \dots, q_n) \\ P_y = f_y(q_1, q_2, \dots, q_n) \\ P_z = f_z(q_1, q_2, \dots, q_n) \end{cases}$$
(4.12)

where (P_x, P_y, P_z) , (q_1, q_2, \ldots, q_n) , f_x, f_y, f_z have the same definitions as in equations (4.1)–(4.3).

According to (4.8) and (4.9), with $P_1 = (x, y, z) \in C$, $P_2 = (P_x, P_y, P_z) \in W$, the following index function could be defined:

$$J(q_1, q_2, \dots, q_n) = ||P_1 - P_2||^2 = (x - P_x)^2 + (y - P_y)^2 + (z - P_z)^2$$
(4.13)

and the Reachability Testing problem can be transformed into the following nonlinear optimization problem:

PROBLEM 4.2.

minimize
$$J(q_1, q_2, \dots, q_n)$$
 (4.14)

s.t.
$$\begin{cases} q_{i\min} \leqslant q_{i} \leqslant q_{i\max}, & i = 1, \dots, n, \\ x_{\min} \leqslant x \leqslant x_{\max} & \\ y_{\min} \leqslant y \leqslant y_{\max} & \\ z_{\min} \leqslant z \leqslant z_{\max}. \end{cases}$$
(4.15)

If a set of q_i and (x, y, z) could be found such that J = 0, it can be concluded that the cuboid overlaps with the robot reachable workspace because there is at least one point of the cuboid in the workspace of the robot.

The cuboid defined by (4.11) is actually a general representation. When the lower and upper bounds of one, two or three inequalities in (4.11) are set to the same value, a rectangle, a line segment or a point can be represented.

It should be mentioned that although the solution of the Problem 4.2 is not unique due to the multi-solution property of trigonometric functions, it is enough to solve the Reachability Testing problem. There may be more than one joint configuration for which J = 0. If any of them is found, we are done. If no joint configuration that makes J = 0 can be found, we conclude that the object is not reachable. In other word, J = 0 means that there exists at least one point of the cuboid within the workspace of the robot while $J \neq 0$ means there is not. So the problem of whether two objects overlap can be definitely solved by the Problem 4.2. It is particularly valuable in the collision free path planning problem where the primary concern is to keep the robot from colliding with the enviornment. Now the question becomes what is the appropriate algorithm to solve this problem. If the global minimum of the optimization of Problem 4.2 can be found, the original reachability test problem is solved. A brief literature review will be given in the next section.

5. A Short Overview of the Optimization Methods

Almost all non-linear optimization algorithms have certain conditions for convergence. To ensure the convergence, most of them require that the Hessian Matrix be positive definite. But in Problem 4.2, because of the nature of f_x , f_y , f_z , the index function J's Hessian Matrix is not always positive definite. Also, many algorithms deals with a certain type of constraints. There are few algorithms which deal with the same constraints as in the reachability test problem. So, most of the classical optimization methods are not suitable for this problem.

The Tunneling Algorithm, in the case one dimension function can handle any kind of function with the constraints of the form $A \leq X \leq B$ (A and B are the low and up bounds of vector X), it does not have any requirements on the Hessian Matrix, that is one of its points of superiority; the other point of superiority is its guaranteed convergence to a global extrema in finite steps. It is also simple and easy to apply. The only requirement that the tunneling algorithm needs about the function is the existence of the first and second order derivative. All these make the Tunneling Algorithms suitable for solving the reachability test problem.

6. Tunneling Algorithm Paradigm

The paradigm could viewed at two different levels: the abstract level and the concrete level. At the abstract level, the paradigm for designing the tunneling algorithm [1-3] can be split into two phases: (a) a local minimization phase; and (b) the tunneling phase.

6.1. ABSTRACT LEVEL DESIGN

The minimization phase is designed to decrease the value of the function until a local minima is found. For a given feasible start point, any minimization algorithm with a descent property on f(X) could be used to fulfill the task.

The tunneling phase is designed to obtain a good starting point for the next minimization phase. It begins with the point X^* obtained in the previous minimization phase, any try to find a zero of the tunneling function.

A tunneling function could be defined as:

$$T(X, J(X^*)) = (J(X) - J(X^*)).$$
(6.1)

Once a new point X_0 is found such that $T(X_0, J(X^*)) \leq 0$ and $X_0 \neq X^*$ the tunneling phase is terminated, this new point X_0 is taken as the starting point for the next minimization phase.

Notice that the local minima X^* of the minimization phase will also make the tunneling function $T(X, J(X^*))$ equal to zero. How could the local minima obtained in the minimization phase be excluded? The answer is to construct a tunneling function in the following form:

$$T(X,L) = \frac{(J(X) - J(X^*))}{\left((X - X^*)^T (X - X^*)\right)^l}$$
(6.2)

where $L = X^*$, $J(X^*)$, l), l is called pole strength of X^* , l is used to cancel the zero of $T(X^*, L)$, as shown in Figure 2.

By using this definition, the desired pole X_0 (i.e. X_2^*) is kept and the X^* (i.e. X_1^*) is eliminated (see Figure 2(b)).

It is possible that the tunneling phase tunnels the solution to another local minimum at the same level. To avoid this situation, we need to keep the information X_i^* , l_i of all intermediate points and their $J(X_i^*)$ values and generate the function T(X, L) as:

$$T(X,L) = \frac{J(X) - J(X^*)}{\prod\limits_{i=1}^{p} \left\{ (X - X_i^*)^T (X - X_i^*) \right\}^{l_i}}.$$
(6.3)

When a better value of J^* is found, the stored information is removed, setting p = 1.

6.1.1. A Geometric Interpretation of Tunneling Algorithm

Let us look at simple one dimensional example. Suppose there is a continuous function f(x), with the x - f(x) relation plotted on Figure 2. Starting from any point



Fig. 2. A simple illustration of Tunneling Algorithm (a) f(X) - X relation; (b) $T(X, X^*) - X$ relation.

 x_1 ($x_{\min} \leq x_1 \leq x_{\max}$), a local minima is found as x_1^* ,

$$f(x_1^*) \leqslant f(x_1) \tag{6.4}$$

with the value of x_1^* and $f(x_1^*)$, a tunneling phase is started ending at x_2 , the zero of the tunneling function. Obviously $f(x_2) \leq f(x_1^*)$, this becomes a new start point for the next minimization phase, which results in x_2^* such that

$$f(x_2^*) \le f(x_2).$$
 (6.5)

6.1.2. The Elegant Property of the Tunneling Algorithm

Repeating these two phases in sequence, a series of local minima of J(X) can be found at X_i^* , i = 1, 2, ..., m, such that

$$J(X_i^*) \leq J(X_{i+1}^*) \quad A \leq X_i^* \leq B, \quad i = 1, 2, \dots, m-1.$$

In this way the global minima can be approached in an orderly fashion. If a zero of (6.1) can not be found after a sufficient computing time (this time could be chosen as the worst case time needed in finding the previous optima times a constant. See also 6.2.3) then it can be assumed that the desired minima is found.

Notice that the tunneling phase tunnels the irrelevant local minima, regardless of how many there are and where they are. Each time the tunneling phase is looking for a new X with its function value equal or less than the local minima obtained in the minimization phase, that means it always tries to tunnel X to a new yet lower position.

It is very interesting to note that throughout, the two phases themselves converge only to local minimum. The sequence of their connection ensures the algorithm to converge to the global extrema in finite steps. (Proof can be found in [3].)

6.2. THE ALGORITHMS

In this section the details of the two phases in the tunneling algorithm will be given and some practical considerations on applications will also be presented.

6.2.1. The Minimization Phase Algorithm

In the minimization phase the conventional gradient method is used and described as follows:

```
input: (\alpha, X_0)
output: (X_0)
begin
while (d J(X_0)^T d J(X_0) > 10^{-9}) and (\alpha > 1/2^{20})
X_1 = X_0 - \alpha^* J(X_0)
if (J(X_1) < J(X_0)) then X_0 = X_1
else \alpha = \alpha/2
end while
end
```

where $dJ(X_0) = [\partial J(X)/\partial x_1, \partial J(X)/\partial x_2, \dots, \partial J(X)/\partial x_n | X = X_0]^T$ is the gradient vector of J(X) at X_0 , and $0 < \alpha \leq 1$ is a scalar to control the step length. Initially, α is chosen as 1.

6.2.2. The Tunneling Phase Algorithm

The tunneling phase is actually a zero finding phase. It tries to find out a point X^* which is different from the output of minimization phase (X_0) but has the same or lower function value.

The basic structure of the algorithm can be written as (6.6)–(6.7):

$$T_x(X_0, L)^* dX = -T(X_0, L).$$
(6.6)

By solving this equation, the direction vector d X can be obtained. The next position vector can be computed as:

$$X^* = X_0 + \beta^* d X \tag{6.7}$$

where $0 < \beta < 1$ is a stepsize controller used to enforce a descent property on the norm of the error

$$P(X_0) = T(X_0, L)^T T(X_0, L).$$
(6.8)

If $P(X^*) < P(X_0)$ is satisfied, accept X^* as the new position vector to start the next solution of (6.6).

If the stepsize controller becomes too small, (<1/32[2]) then it is said to have been attracted to the neighbor of a singular point X_s . A singular point is a point that $T'_x(X_0, L) = 0$; once it is known, an equivalent system as follows can be established:

$$S(X) = \frac{T(X)}{\left((X - X_m)^T (X - X_m)\right)^p}$$
(6.9)

where X_m denotes the position of a "movable pole" [1] and p denotes its strength. The strength of the movable pole p can be easily calculated as follows:

$$S_x(X) = \frac{1}{\left((X - X_m)^T (X - X_m)\right)^p} \left[f_x(X) - \frac{2^* p^* (X - X_m)^* f(X)}{(X - X_m)^T (X - X_m)} \right].$$
 (6.10)

Compute $S_x(X)$ at point $X = X_m + e$, where e is a small random number used just to avoid a divizion by zero. A tentative displacement dX is computed from the equivalent system:

$$S_x(X)^* dX = -S(X)$$
 (6.11)

$$X_{\rm new} = X + d X/32. \tag{6.12}$$

If the performance index $Q(X) = S(X)^T S(X)$ does not decrease, the pole strength p is increased to p + dp. Repeat the above process. If the condition $Q(X_{new}) < Q(X)$ is satisfied, then keep the present pole strength and use X_{new} as the new position vector for the next iteration on S(X). Typically dp = 0.1. By using the equivalent system, the singularity is very effectively canceled [1].

6.2.3. The Terminator Conditions

The stopping condition for the minimization phase and tunneling phase is quite straight forward. The stopping condition for global optimality is to some extent difficult, because it is impossible to verify T(X, L) > 0 for every X in the search space. According to the authors of [1], a reasonable limit of iterations could be set as the overall stopping condition. For example, if the longest previous tunneling phase has N iterations then the overall stopping condition can be taken as 10N. This condition does take into account the complexity of each problem and the previous information gained by the algorithm as it proceeds to the global minimum. In practice an additional parameter e is introduced, that is the algorithm will also terminate when J (global minimum) $\leq J$ (local minimum) $\leq e$, where e denotes the safe distance from the robot to the object. These two stopping condition work together can reduce the computation time.

6.2.4. The Convergence of the Tunneling Algorithm

The global convergence to global optima of this tunneling algorithm has been theoretically proven for any given one dimensional scalar function of n variables (represented by a vector $X = [x_1, x_2, ..., x_n]$) with $A \leq X \leq B$. [2] Interested reader can find the detailed proof in [3].

7. The Application of the Tunneling Algorithm to the Reachability Test Problem

In this section, some results of applications of the tunneling algorithm to the Reachability Test Problem are given and they are compared with the method RGRG, which stands for the Revised Generalized Reduced Gradient, presented in [5].

First let us look at the advantage of using the Tunneling Algorithms (see Table I).

Table I clearly shows the advantages that the Tunneling Algorithm has the RGRG algorithm. By using the tunneling algorithm, the start point could be arbitrary while in most of the other classical algorithms and the RGRG algorithm, the results depend on the start point. Although one use a lot of start points, the computation cost will be high, and there is still no guarantee of obtaining the correct result.

To look at the effectiveness of the tunneling algorithm, the example of a two link planar robot (shown in Figure 1) is used. Suppose the range of the two joint variables are:

$$-90^0 \leqslant q_1 \leqslant 180^0,\tag{7.1}$$

(7.2)

$$-90^0 \leqslant q_2 \leqslant 180^0$$

Table I. A simple comparison between the Tunneling Algorithm and the RGRG Algorithm

Method	Start Point	Convergence
Tunneling Algorithm	Arbitrary	Guaranteed
RGRG (proposed in [5])	Dependent	No guarantee

Given Points (x_0, y_0)	Start Point (q_1, q_2) (deg)	RGRG test result	Tunneling Algorithm test result
a: (-14,0)	$(0^{\circ}, 0^{\circ})$	N	R
b: (-10, -4)	(45°, 45°)	R	R
c: (-4, -10)	(180°, 180°)	R	R
d: (−7, 0)	$(0^{\circ}, 0^{\circ})$	Ν	R
e: (-10, -4)	(0°,0°)	Ν	R
f: (0,13)	(-90°, 0.1°)	R	R
g: (15,0)	(180°, 180°)	Ν	N
h: (1,0)	(-90°, -90°)	Ν	Ν

Table II. The comparison of the Tunneling Algorithm and RGRG Algorithm (R: reachable, N: non-reachable)

and the two link length is a = 10, b = 4.

By observing Table II and look at Figure 3, we find that the Tunneling Algorithm can always find the right answer while the RGRG sometimes does not. To further verify the effectiveness of the Tunneling Algorithm, a three dimensional robot with:

$$P_x = \cos\left(q_1\right) \cdot PP,\tag{7.3}$$

$$P_y = \sin\left(q_1\right) \cdot PP,\tag{7.4}$$

$$P_z = 9\sin(q_2) + 9\sin(q_3) + 4\sin(q_4) + 10.5$$
(7.5)

$$PP = 9\cos(q_2) + 9\cos(q_3) + 4\cos(q_4) \tag{7.6}$$

is also used to do the following reachability test. P_x, P_y, P_z are the X, Y, Z coordinates of the robot's hand, q_1, q_2, q_3, q_4 are the four joint angles of the robot. Here not constraints are imposed on the joint angles.

Notice that in Table III, test No. 1 and No. 3 are cuboid; while test No. 2 is acturally a square, test No. 4 is a line segment and test No. 5 is a single point. This robot does not have any joint limit, its reachable workspace is a ball with origin at (0,0,10.5) and a radius of 22. The test results are correct.

The general computation time for Tunneling and RGRG algorithm are $O(a(M^*n + K))$ and $O(M^*n)$ respectively, where M is the maximum number of cycles for minimization, n is the input size, and a is the maximum number of switches between

Test No.	Given Target	Tunneling Algorithm test result
1	(-1, -1, 33)-(1, 1, 35)	N
2	(9,-22,9)-(11,-22,11)	Ν
3	(7,7,2)–(12,12,4)	R
4	(0,0,20)–(0,0,23)	R
5	(0,30,0)-(0,30,0)	Ν

Table III. The result of reachability test for 3D targets (R: reachable, N: non-reachable)



Fig. 3. The reachable workspace of 2R planar robot with $-90^\circ \leqslant q_1 \leqslant 180^\circ$ and $-90^\circ \leqslant q_2 \leqslant 180^\circ$.

a minimization phase and a tunneling phase, K is the maximum number of cycles in tunneling phase algorithm. This means, the tunneling algorithm takes longer time, but that is the cost to guarantee the global minimum. Overall it is still a better algorithm than RGRG algorithm.

8. Other Problems that could be Benefitted from the Tunneling Algorithms

One of the direct benefits we can get is applying the tunneling algorithm in the computation of the robot reachable workspace and the boundary of the dexterous workspace [13].

If the reachable workspace is bounded in a three dimensional cuboid, we can divide the cuboid into finite sub-cuboids with desired accuracy, and use the Tunneling Algorithm to test these sub-cuboids one by one.

To make it more specific, assume that the workspace is bounded in a cuboid:

$x_{\min} \leqslant P_x \leqslant x_{\max},$	(8.1)
$y_{\min} \leqslant P_{oldsymbol{y}} \leqslant y_{\max}$,	(8.2)
$z_{\min} \leqslant P_z \leqslant z_{\max},$	(8.3)

where P_x, P_y, P_z are the same as in (4.1)–(4.3). If the desired accuracy is e, then the cuboid can be divided according to three directions:

x-direction	$u = \lceil (x_{\max} - x_{\min})/e ceil,$	(8.4)
y-direction	$v = \lceil (y_{\max} - y_{\min})/e \rceil,$	(8.5)
z-direction	$w = \left[(z_{\max} - z_{\min})/e \right],$	(8.6)

There will be altogether $u^*v^*w^*$ sub-cuboids. Using the Tunneling Algorithm to test them one by one, the reachable workspace of the robot can be obtained. This is of cource very inefficient. We will have another paper to deal with this matter.

9. Conclusion

In this paper, the reachability test problem is addressed and formulated in a new way. A global minimization algorithm - Tunneling Algorithm is introduced, which is said [3] to converge in the case of one dimensional scalar functions. Two robots are taken as examples to show that the tunneling algorithms are working and have good convergence properties.

Acknowlegement

The author would like to thank all the reviewers for their comments which improved the presentation of this paper.

References

- 1. Levy, A. V. and Gomez, S.: The tunneling method applies to global optimization, Numerical Optimization (1984).
- 2. Levy, A. V., Montalvo, A., Gomez, S., and Calderon, A.: Topics in global optimization, Lecture Notes in Mathematics 909, 1982.
- 3. Levy, A. V. and Montalvo, A.: A modification to the tunneling algorithm for finding the global minima of an arbitrary one dimensional function, Communications Tecnicas, Serie Naranja, No. 240, IIMAS-UNAM, 1980.
- 4. Scales L. E., Introduction to Non-Linear Optimization, Springer-Verlag, New York, 1985.
- 5. Ying Z., Xi, Y. and Zhang Z., Test of the reachability of a robot to an object, Proc. IEEE Int. Conf. on Robotics and Automation, 1989, pp. 490-494.
- 6. Tsai Y. C. and Soni A. H. Accessible region and synthesis of robot arms, ASME J. Mech. Design 103 (1981), 803-811.
- 7. Lee T. W. and Yang D. C. H. On the evaluation of manipulator workspace, ASME J. Mech. Trans. Autom. Des. 105 (1983), 70-77.
- 8. Yang D. C. H. and Lee T. W.: On the workspace of mechanical manipulators, ASME J. Mech. Trans. Autom. Des. 105 (1983), 62-69
- 9. Sugimoto K. and Duffy J., Determination of extreme distance of a robot hand part 1: A general theory, ASME J. Mech. Design 103 (1981), 631-636.
- 10. Rastegar J. and Deravi P., Methods to determine workspace, its subspaces with different numbers of configurations and all possible configurations of a manipulator, Mech. Mach. Theory 22 (1987), 343-350.

- 11. Ming-shu Hsu and Kohli D., Boundary surfaces and accessibility regions for regional structures of manipulators, *Mech. Mach. Theory* 22(3) (1987), 227–289.
- 12. Paul R. P., Robot Manipulators: Mathematics, Programming and Control, MIT Press, 1981.
- 13. Zone-Chang Lai and CHia-Hsuoang Menq, The dexterous workspace of simple manipulators, *IEEE Journal of Robotics and Automation* 4(1) (1988).