Robotica (1997) volume 15, pp 435–447. Printed in the United Kingdom © 1997 Cambridge University Press

Manoeuvring highly redundant manipulators

E. Sahin Conkur and Rob Buckingham

AMARC, Faculty of Engineering, University of Bristol, 26–32 Park Row, Bristol, BS1 5LY (UK)

(Received in Final Form: August 2, 1996)

SUMMARY

A task based approach to the issue of redundant robots starts from the premise that there are obstacles that cannot be removed from the working area and which therefore must be avoided. This statement produces the requirement for a device with a certain degree of mobility, and stresses the need to ensure that the aim is twofold: reach the goal and avoid obstacles. But avoiding obstacles is not the same objective as keeping as far away from an obstacle as possible; the primary goal is still to reach the target. In fact humans use soft contact to reach targets that are at the periphery of their reach. This soft distributed contact has the effect of smoothing the surface of the object and hence there is an element of only being interested in obstacle detail at the appropriate scale to achieve the task.

This paper describes a new approach to collision avoidance based on using a global path finding algorithm, in this case using Laplacian potential fields, in conjunction with a simple local geometrically based algorithm for avoiding obstacles and maximising the use of manoeuvring space in a manner which is not limited by digital computation resolution issues. This extra technique is in some ways analogous to the human soft contact approach.

Three examples are presented to illustrate the robustness of the algorithm. In order to be able to compare results with other techniques, an environment measurement scheme is defined which gives an indication of the difficulty of the trajectory being followed.

KEYWORDS: Redundant robots; Path planning; Obstacle avoidance.

1. INTRODUCTION

A redundant manipulator is defined as having an infinite number of solutions to the joint variables of the manipulator for a given task.[1] One way in which this is achieved is when there are more joint variables than required for a specific task. Although doing so includes great difficulties, redundant manipulators have been of interest for some time[2] since they have the very considerable potential of offering great flexibility to use the workspace efficiently and reduce the need for well arranged environments.[3] When multiple degrees of freedom are employed, such a manipulator can also be used for reaching the end of a long curved path in large machines for the purposes such as maintenance, repair or inspection.[4,5]

There are two distinct aspects of controlling a highly redundant robot such that it reaches its goal whilst avoiding all obstacles. The first aspect is one of finding a path.[6] The second is following that path.[7] Within both tasks there is a requirement to ensure that the finite dimensions of the robot are taken into account. This is the technique of motion planning that humans use, in the sense that, even when there is a incomplete knowledge about whether a path leads to a goal, a path is chosen in a 'cerebral' manner taking into account available kinematic and environment knowledge, before the motion is conducted at the 'cerebellum' level using proprioception and senses interacting with the environment itself.

1.1 Jacobian based techniques

When the path for the end-effector in Cartesian space is determined, a sequence of joint variables satisfying constraints while end-effector follows this path is referred to as redundancy resolution.[6] For non-redundant manipulators, *resolved motion rate control* has been mostly used, which makes use of inverse kinematics at velocity level.[8] When the relationship between task space variables and joint space variables is defined as

$$x = \mathbf{F}(\theta) \tag{1}$$

where $\mathbf{F}(\theta)$: $R'' \rightarrow R'''$ is a continuous function, by differentiating this equation with respect to time, the equation below is obtained.

$$\dot{x} = \mathbf{J}(\theta)\dot{\theta} \tag{2}$$

where $\mathbf{J}(\theta) \in R'''^{\times''}$ is the Jacobian matrix for $\mathbf{F}(\theta)$.

When a desired end-effector velocity is given, joint velocities are computed by solving equation (2). This equation will be underdetermined in the case of redundant manipulators since $n > m$. A solution to equation (2) can be determined by the following equation which is widely used for obstacle avoidance;[9]

$$\dot{\theta} = \mathbf{J}^{\#}\dot{\mathbf{r}} + k(\mathbf{I} - \mathbf{J}^{\#}\mathbf{J})\mathbf{z} \tag{3}$$

where $\mathbf{J}^{\#}$ is a generalised inverse, $\mathbf{I}$ is unit matrix and $\mathbf{z}$ is an arbitrary vector.

The first term on the right hand side in the above equation causes the end-effector to follow the trajectory while the null space component (the second term) configures the links without affecting the end-effector position. The null space component includes an arbitrary vector $\mathbf{z}$ which can be chosen as smooth scalar function and which can be used as a performance criterion such as

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

obstacle avoidance. The main problem here is to determine an appropriate **z** function which is to be optimised.[10] Rahmanian-Shahri and Troch[11] have recently proposed a way of constructing the optimisation criterion for obstacle avoidance. It is difficult to find closed-form expression of distances between links and obstacles. Hence, boundary ellipse functions are proposed as optimisation criteria to be used in the null space of joint variables since ellipse has a simple expression. This function of the link near the obstacle is maximised to avoid the obstacle. The value of the gain constant k which affects the magnitude of the null space vector is determined on an error and trial basic. Optimisation criterion for obstacle avoidance is usually chosen out of a potential function to maximise distances between the links and the obstacles, which may be very difficult to compute. Nevertheless, a simple null pace vector is presented in reference 12 maximising areas rather than distances between obstacles and links. Another approach for redundancy resolution is called task priority based redundancy control. A task is divided into two sub-tasks and one of them has priority over the other. Nakamura chooses second priority task as obstacle avoidance and implements it.[3] This technique is derivable from the gradient projection technique if one task variable is cancelled.[8] Maciejewski and Klein present a similar approach.[10] The primary command of end-effector velocity is first satisfied and then system redundancy is used for obstacle avoidance in a way that an obstacle avoidance point which is closest to the obstacle is determined and a velocity vector opposite to the obstacle surface is assigned to it. As already seen, a generalised inverse is used in the above techniques simply because it is not defined to invert a matrix which is not square. On the other hand, to compute joint velocities, *extended Jacobian technique* uses additional constraints extending the dimension of task space, which are added to Jacobian matrix so that it can be inverted. Chiacchio et al.[2] use this approach with closed-loop inverse kinematic scheme. One difficulty is to determine the constraints for a general case for obstacle avoidance as seen from the example of seven degrees of freedom manipulator successfully inserted into a torus in reference 4. Schilling et al.[7] consider planning a joint-space trajectory which results in manipulator links following a given simply curved path for the end-effector closely, therefore avoiding obstacles indirectly. Manipulator links are decoupled into proximal and distal parts, which uses resolved-rate control equations. The joint motions near the end-effector are kept tangent to the path as much as possible, whereas the motion of the other links is limited to avoid collision. Bagchi and Hatwal use sensory data, analysed using a fuzzy logic controller, to avoid stationary and moving obstacles.[13] In this paper, when any manipulator link is close enough to an obstacle, the null space of the joint variables are activated in a manner determined by the fuzzy controller. Generally speaking, there are some serious restrictions on employing the above techniques. First, in a cluttered environment, it is difficult to find functions for optimisation to be used for obstacle avoidance. Second, singularities occur both kinematically and algorithmically. Third, they are limited to special cases of few degrees of freedom. Fourth, global optimality which can be dealt with global approaches[8] at the cost of being computationally very expensive cannot be guaranteed.

1.2 Geometric techniques

Path planning as a separate field is based on geometric techniques. Using such an approach the distinction between route finding and path following is not always clear, because, if a path can be found that is free from obstacles for the whole manipulator, obstacle avoidance is automatically performed and path following becomes a trivial task.

Graph search techniques find a collision free path by determining collision free spaces by means of a graph.[14] A representative example of graph search techniques is the configuration space approach. A configuration of moving objects is defined as a set of all parameters that completely specifies every point on the object. Configuration space presentation transforms the workspace, obstacles and paths expressed in Cartesian co-ordinates into the space of joint co-ordinates. Each configuration of the manipulator is presented by a point in C-space while trajectory is presented by a line. C-space presentation is used by most techniques. There are several ways of finding a collison free path in the C-space; A path can be found by decomposing k-dimensional free space into a finite number of connected cells. The adjacency between cells are set up, which results in a discrete path-searching in a graph.[15] The road-map approach does not decompose free space into simple cells, instead a one dimensional construction of a structure of curves in free space is established. Denker and Atherton[16] improve the efficiency of roadmap approach by reducing the necessity for computing time and memory requirements. Searching through the map is exhaustive since road map graph structure consists of a large number of nodes and links. The search is simplified with the aid of some means such as simplification of obstacle shapes and employment of tangent graphs. The algorithm that Schweikard[17] presents determines free space of motion assuming that an algorithm to compute the intersection of a line with an obstacle in configuration space is given. The computed motion segments consist of a sequence of line segments in C-space. If the initial line segment is not collison free, then it is redefined using via points connecting initial point and goal point. There is, as always expressed, a great difficulty in building and searching C-space. This still prevents real-time applications, especially for manipulators with many links as well as not guaranteeing avoidance of forbidden regions. There are some alternative methods such as sequential framework.[18] In this technique, motion planning in an environment cluttered with obstacles is achieved by decomposing n dimensional problem into m-dimensional sub-problems. Each sub-group's motion is individually

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

planned and in the case of no solution for any of these sub-groups, backtracking is used. Gupta and Zhu[19] improve this method using potential fields determined over bitmap presentation on the sub-spaces in solving each sub-problem. The method Li and Trabia[20] present divides a given path into points and minimises the distance between the end-effector location and a given path point while penalties included in each objective function prevent links from colliding with obstacles. In the algorithm presented by Reznik and Lumelsky[21,22] the manipulator links are decoupled so that the motion of every link can be planned individually in the presence of obstacles. To avoid obstacles, each link is moved using the curve called the tractrix, which causes no larger motion at the links for a given motion at the end-effector. Trajectory for the end-effector itself can be computed through any searching algorithm used for a point robot. It is assumed that every link has a sensing envelope which allows the link to have full information about environment.

There are also hybrid methods such as the one Mayorga et al.[12] propose. It uses a potential function to guide the end-effector to the goal point as obstacle avoidance of the joints is carried out by the null space vector which includes a potential function maximising some areas between the links and the obstacles. Kinematic singularity and local minima avoidance are also dealt with in the algorithm.

1.3 Potiential fields

The potential field method has been of interest to resolve path planning and obstacle avoidance for mobile robots and manipulators since it was first introduced by Khatib.[23] The method proposed in that paper uses a working area which is under influence of an artificial potential field. In this field the goal is presented by an attractive pole while the obstacles are presented by repulsive surfaces. This force that is used to control robot motion is the negative gradient of the artificial potential function. This force can be calculated analytically at any point in the working envelope, which results in the robot reaching the goal point. The methods based on potential fields can also be categorised as global and local.[24] The global methods using C-space obstacles are able to create a free path from the initial position of the end-effector to the goal point, whereas the local methods using the local information generate repulsive forces taking the robot to the goal point. One major criticism the potential field technique largely receives is that they have the local minima that are defined as the points in which the robot cannot move and is trapped.[25] In addition, the existing methods are in general presented for point robots other than manipulators with many links. There have been two approaches to cope with local minima.[24] The first one is to detect local minima by means of effective searching algorithms. Altering field to eliminate local minima is the solution. Nevertheless, it causes extra computation. The second one is to create a potential field that does not include local minima, which is straightforward.

A potential field technique presented by Graham[4] and others overcomes some of the difficulties mentioned above. This technique is used for a robot manipulator rather than point robots. It is a global method in real or near real time. The working envelope and obstacles are uniquely defined by boundary conditions. There are no local minima or singularities, although second order minima (saddle points) can cause difficulties. The end-effector is guided towards the goal point as the links avoid obstacles by means of the potential field gradient values which directed to the goal point away from obstacles. However, the potential field technique may fail to avoid obstacles when tight manoeuvring of the links is required.

The method presented in the work by Graham uses a scalar potential field governed by Laplace equation:

$$\nabla^2\Phi = 0 \tag{4}$$

under Dirichlet boundary conditions. Equation (4) is on a domain Ω and the boundary of Ω, that is $\partial\Omega$, consists of the boundaries of all obstacles and the goal. The workspace Ω is represented as a grid of certain dimensions that is determined with respect to the precision of the task. The partial equation which presents equation (4) and is used for iteration process on the grid is given:

$$\Phi_{(i,j)} = \tfrac{1}{4}(\Phi_{(i+1,j)} + \Phi_{(i-1,j)} + \Phi_{(i,j+1)} + \Phi_{(i,j-1)}) \tag{5}$$

where i is position on the grid in the x direction and j is position on the grid in the y direction.

The maximum gradient at any point is analogous to the direction of conventional current flow in the conducting medium and is the gradient which is used to generate the control force at that point. The goal is set to a value of -2^{126} and boundary points are set to zero. The iteration procedure produces field values at all the points on the grid, from which linear interpolation is used to get a field value at any point in the region. Any field line uniquely defines a path from any point within the field to the goal position and guarantees obstacle avoidance. The solution to the find path problem is actually solved on line as the robot moves.

1.4 Collision avoidance using a potential field technique

Route finding and path following can both be achieved with Laplacian fields, and in the process can avoid any call on inverse kinematics. The technique described by Graham, decouples each link and uses the field values around various control points along the length of the link to derive a virtual torque which controls the link motion. The virtual torque, τ, is computed using equation:

$$\tau = \frac{\sum \sin(\beta(i)) \cdot i}{n} \tag{6}$$

where n is number of the control point on the link, Figure 1.

The configuration of the redundant manipulator is

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

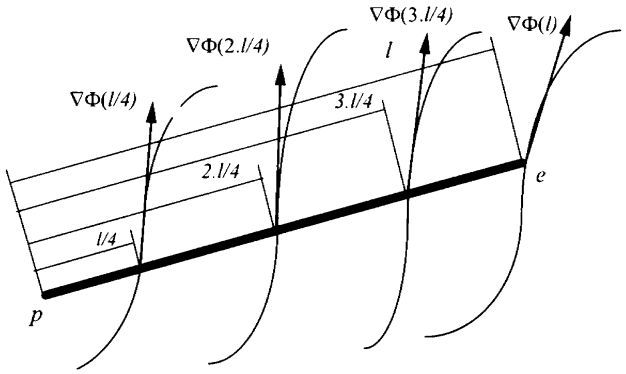

Fig. 1. Robot-field interaction.

controlled using a process called *settling*. Each link is moved to position of lower potential difference and followed a movement of each distal link until the whole device is settled. The motion paradigm is minimise the potential difference all the links at each time increment. However, optimum performance would be achieved with an infinite number of control points on the surface of a link and a grid size that would capture all possible objects. This makes such a technique computationally intensive and also reduces its attractiveness as a method that allows a dynamically changing environment.

2. ENHANCED COLLISION AVOIDANCE

The algorithm described and illustrated in the remainder of the text, implements the settling algorithm described by Graham and adds a geometrical collision avoidance layer which is only activated when the settling algorithm produces a collision. The extra layer has been designed to specifically overcome the loop holes of the potential field scheme as described in the previous section.

Within this extra layer, each obstacle is described as a single or collection of ellipses. Each ellipse is grown by a safety margin so that there is a elliptical safety zone around each ellipse. Additionally each manipulator link is modelled as a line. Since these geometric shapes are completely defined by simple equations, any intersection can be found. This avoids the problem of not being able to implement an infinite number of potential field control points on each link.

If an intersection is found between a link and an ellipse, the link must then be repulsed from the ellipse. The key to the robustness of such an algorithm is the ability to ensure appropriate movement of all links distal to the one being repulsed.

The exact movement of each link is therefore decided just prior to motion, based on the potential field algorithm and, if there is an intersection detected, the ellipse repulsion algorithm.

The use of ellipses, or ellipsoids in three dimensions, is a simplification for this demonstration. An ellipse has the useful property of being able to describe long thin objects as well as circles, whilst being a simple shape to describe geometrically. Obstacles could be described as pixels or voxels, in which case the basic geometric intersection calculations would be line-line rather than line-ellipse. Similarly, defining a manipulator link as a line is a simplification which can be justified by acknowledging the existence of various algorithms which can grow obstacles, globally or locally to take account of real link dimensions.

2.1 Intersection between link and obstacle

The line and ellipse equations are defined as follows;

$$y = a \cdot x + b \tag{7}$$

$$\frac{(x-c)^2}{u^2} + \frac{(y-d)^2}{v^2} - 1 = 0 \tag{8}$$

The line equation (7) presents the line from minus infinity to plus infinity and is collinear with the manipulator link. When any of the links is moved, the corresponding line equation for that link is constructed (Figure 2). The coefficients a and b in equation (7) are computed using the start point of the link $p(x_p, y_p)$ and the end point of the link $e(x_e, y_e)$ on the line;

$$a = \frac{y_p - y_e}{x_p - x_e} \qquad b = \frac{x_p \cdot y_e - x_e \cdot y_p}{x_p - x_e} \tag{9}$$

The solution to the above system of equations, that is the intersections, is two points called $s(x_s, y_s)$ and $k(x_k, y_k)$ (Figure 3);

$$x_s = \frac{M+H}{K} \qquad y_s = \frac{a \cdot (M+H)}{K} + b \tag{10}$$

$$x_k = \frac{M-H}{K} \qquad y_k = \frac{a \cdot (M-H)}{K} + b \tag{11}$$

where

$$M = -2 \cdot u^2 \cdot a \cdot b + 2 \cdot u^2 \cdot d \cdot a + 2 \cdot v^2 \cdot c$$

$$K = 2 \cdot (n^2 + u^2 \cdot a^2)$$

$$H = 2 \cdot u \cdot v \cdot \sqrt{\begin{array}{r} -2 \cdot a \cdot b \cdot c + 2 \cdot d \cdot a \cdot c - d^2 - b^2 \\ + 2 \cdot d \cdot b + v^2 - a^2 \cdot c^2 + a^2 \cdot u^2 \end{array}}$$

2.2 Detecting intersections

If parameteric forms of the ellipses and manipulator lines are not used it is necessary to construct further vectors to establish intersection. These are the vector ***pk***, when x is

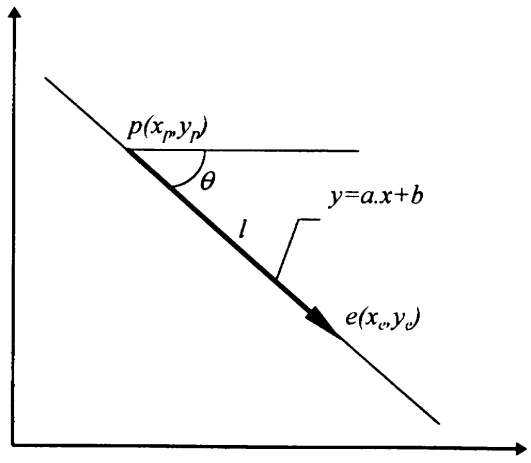

Fig. 2. Description of the vector ***pe*** on the link.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

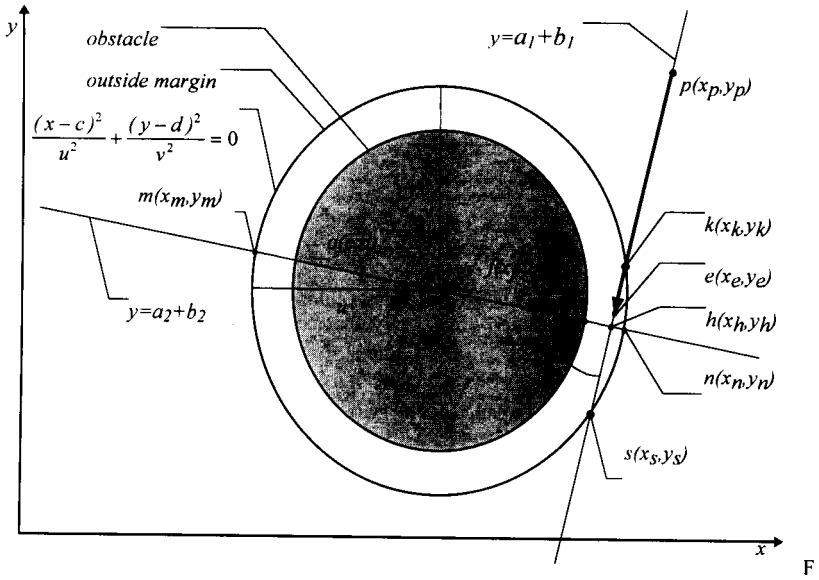

Fig. 3. Description of the obstacle, outside margin, lines and interaction of link and obstacle.

in the first and forth quadrants, and ***ps*** in the second and third quadrants, Figure 4a. The sense of the x component of the vector ***pe*** is compared with the one of the vector ***pk***. If the sense of ***pe*** is opposite the sense of ***pk***, then the link does not intersect with the margin ellipse since there is a distance between the link and the ellipse. In the case of the same sense in the same direction, there are three possible cases that will be encountered. The first one is shown in Figure 4b. The length of the vector ***pe*** is less than the length of the vector ***pk***. Still, there is no intersection. As for the cases in Figure 4c and Figure 4d, the length of ***pe*** is greater than the length of ***pk***. In both

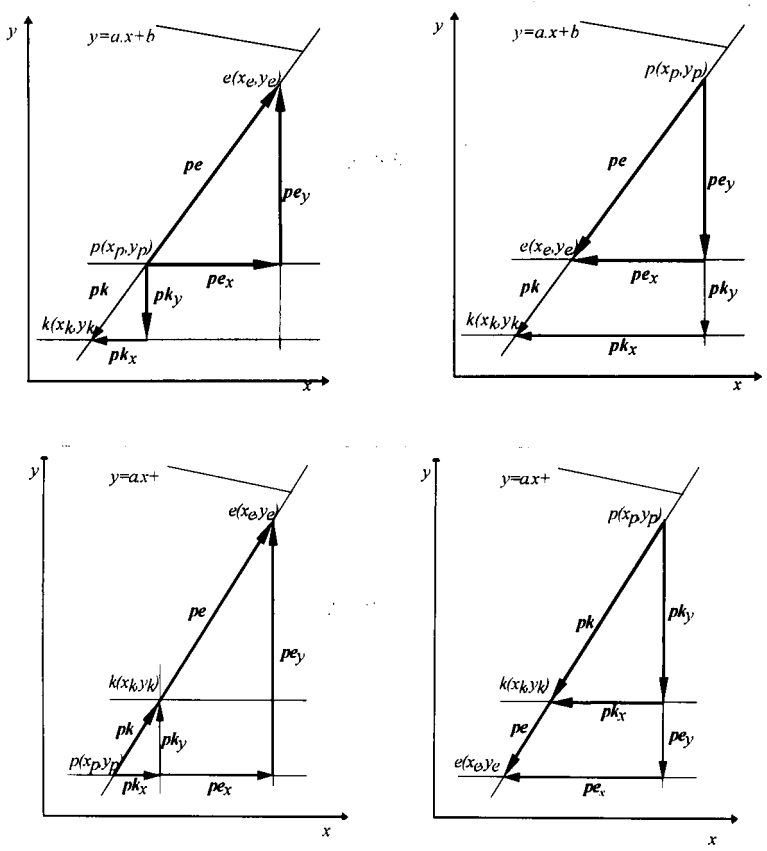

Fig. 4a–d. Intersection detection.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

cases, there is an intersection and the difference between the length of two vectors gives the distance that enters the ellipse.

Once an intersection has been detected the corresponding link must be moved away from the obstacle. The middle point $h(x_h, y_h)$ between the two intersection points $k(x_k, y_k)$ and $s(x_s, y_s)$ can be calculated using

$$x_h = \frac{x_k + x_s}{2} \qquad y_h = \frac{y_k + y_s}{2} \tag{12}$$

As the co-ordinates of the point $h(x_h, y_h)$ and the orientation of the link are known, another line perpendicular to the line which is collinear with the link can be easily drawn. The intersection points $m(x_m, y_m)$ and $n(x_n, y_n)$ are determined by means of the equations (7) and (8). Hence, other two vectors, ***hm*** and ***hn*** are found. Comparing the magnitudes of the vectors, ***hm*** and ***hn***, it can be decided where the link is to be moved. If the difference is less than zero and the angle of the link, θ, is less than zero, then the link is moved so that the absolute value of its angle decreases. If the angle of the link is greater than zero, then the link is moved so that the value of its angle increases. If the difference between ***hm*** and ***hn*** is greater than zero and θ is also less than zero, then the link is moved so that the absolute value of its angle increases. Otherwise, it is moved so that the value of its angle decreases. When the link angle θ is in the second or third quadrant, the same procedure is followed except that the point s is taken instead of the point k. The flowchart of this function called ELLIPSE is shown in Figure 5.

2.3 Interaction between links

Having described a simple algorithm for individual links it is necessary to consider the effect of link motion on distal links. Considering an n link planar manipulator, Figure 6, when the kth link is moved a fixed amount $\Delta\theta_k$, the maximum displacement of the end point of the nth link Δs_n is simply calculated as:

$$\Delta s_n = \Delta\theta_k \cdot l_t \tag{13}$$

where l_t is the straight line from the start point of kth link to the end point of nth link.

If kth link is taken as the first link, the maximum displacement which can occur at the end-point of the

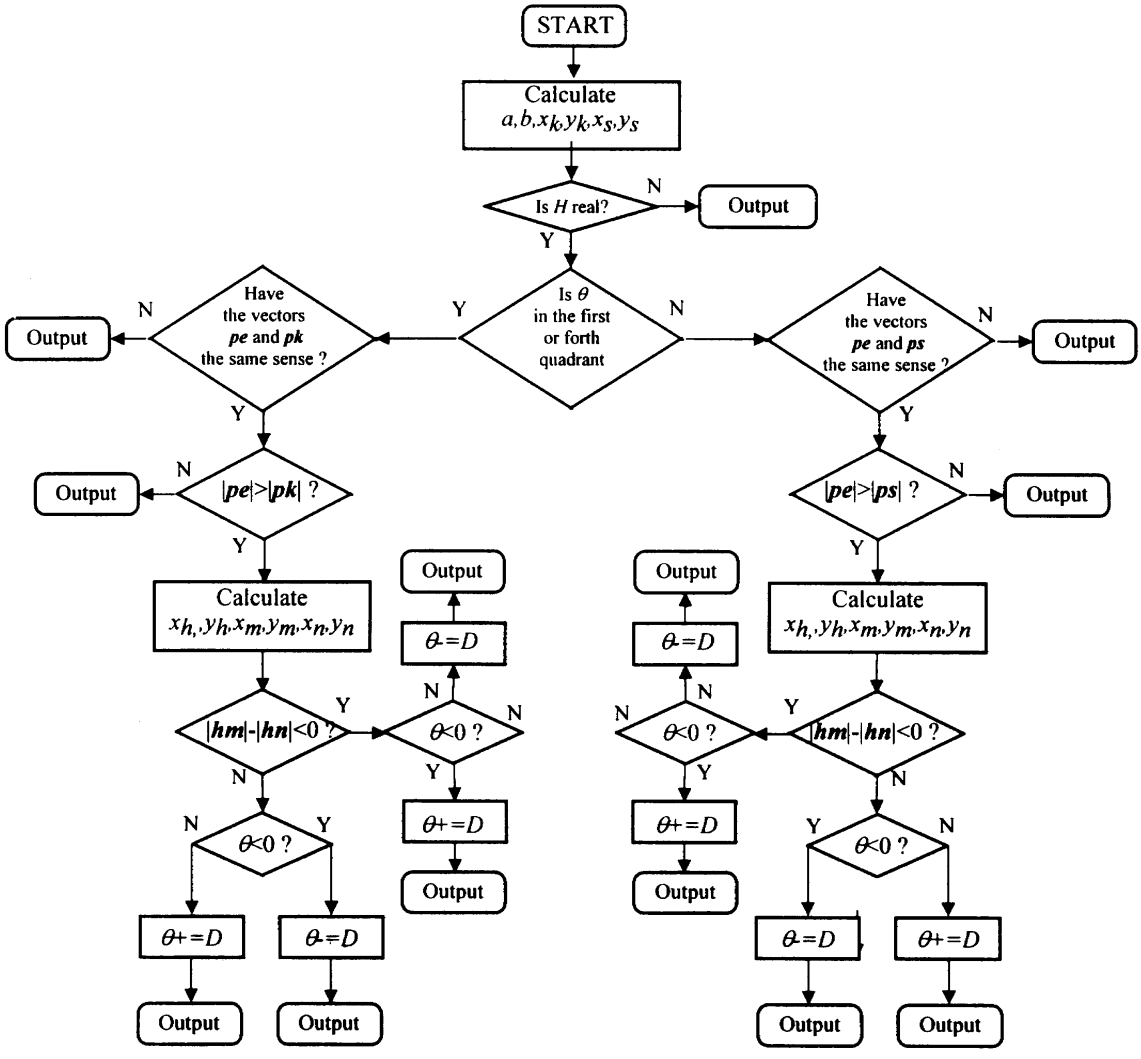

Fig. 5. The flowchart of the function ELLIPSE.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

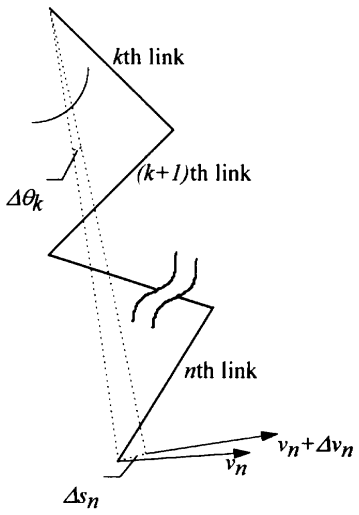

Fig. 6. Determining the velocity of the link n.

nth link is obtained. It is clear that the distance Δs_n should not be able to jump the manipulator from one side of an obstacle to the other, as this would guarantee a collision. However, this leads to unnecessarily wide ellipse margins which if too cautious could make an opening impassable. To reduce this effect the margin ellipses are assigned dynamically so that the maximum moveable distance, Δs_n, will not cause a collision. The effect of this is to slow down the motion as the device extends. This is not unreasonable since the performance of a real system would attenuate with extension. This dynamic re-evaluation of the ellipses leads to the possibility of the new ellipse being larger than the old, which could mean that the link would start within the margin and could then collide with the obstacle if the potential field algorithm moved the link towards the obstacle. This is avoided by increasing the margin ellipse, but again with the result of narrowing the opening.

If the kth link is moved to avoid an intersection, the algorithm assumes that the kth link should be returned to its position defined by the settling algorithm. The nth link should therefore be rotated in the opposite direction to the movement of kth link. To achieve a similar end point displacement backwards, the nth link angular displacement $\Delta\theta_n$ must be greater than $\Delta\theta_k$. This value is calculated using

$$\Delta\theta_n = \$^{-} \qquad (14)\$^{10000l_k} \qquad (14)$$

where l_k is the length of nth link. The end-point of the nth link will not come back to the previous location. However, the aim is not to take the end-point exactly to the previous location but to move the further links away from obstacles to prevent collision through appropriate angular displacements.

Although the angular displacement is fixed and the same for each link while the robot is in the free space, an angular displacement for each link will be determined individually in the case of intersection. Therefore, each link will be moved back with the angular displacement corresponding to its location with respect to the other links. This is implemented in a recursive function called BACK (Figure 7). If the kth link is moved, BACK checks from the most distal link, n, then $n-1, n; n-2, n-1, n; \ldots; k+1, \ldots, n-2, n-1, n$. The flowchart of the whole control of robot motion is shown in Figure 8.

3. COMPUTER SIMULATION

The proposed algorithm for obstacle avoidance through intersection detection with margin ellipses around obstacles has been implemented in the C on a 100 Mhz Pentium. Three examples are included to demonstrate the capabilities of the method. In the first example, a six link planar manipulator is required to reach the goal point through a narrow passage where tight manoeuvring is necessary through an environment cluttered with six obstacles. The link lengths are 150 (unit irrelevant). The co-ordinates of the base of the robot and goal point are $(580, 730)$ and $(496, 255)$ successively. The working envelope is 700×800. The potential field grid size is 100. The angular displacement that the potential field uses is 0.005 radian. The minimum and maximum ellipse margins are 1.5 and 2.5, respectively. The co-ordinates of the obstacles and the radii of margin ellipses are given in Table I. The series of diagrams in Figure 9a, b, c show the device reaching the goal with the obstacle avoidance algorithm activated for all obstacles and links. The pre-computation of the potential field takes 0.54 seconds, with only 7 iterations, with a running time of 7.08 seconds. Figure 9d shows the results of using just the settling algorithm but with a well developed 100 iteration potential field and 14 control points per link. The final configuration shown indicates that the path was not collision free.

A similar six link planar manipulator is to pass through very thin long obstacle modelled by ellipses as seen in Figure 10a, b, c. Again, tight manoeuvring is required. The link lengths are 210. The co-ordinates of the base of the robot and goal points are $(400, 50)$, $(390, 890)$ respectively. The working envelope is 600×950. The grid

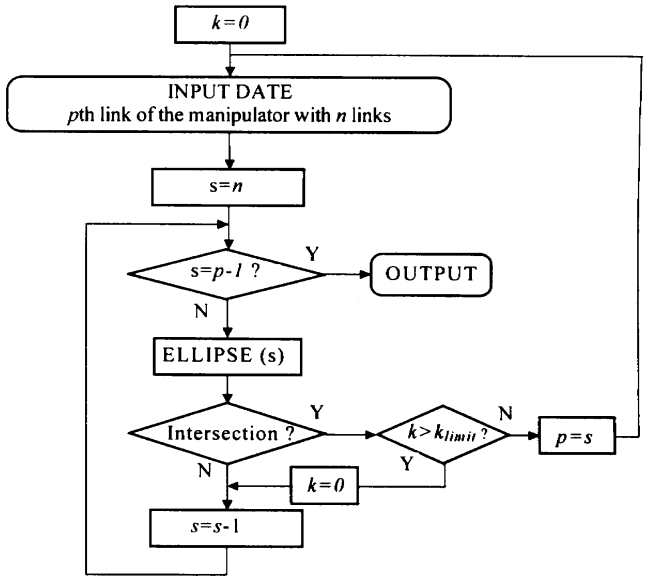

Fig. 7. The BACK function.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

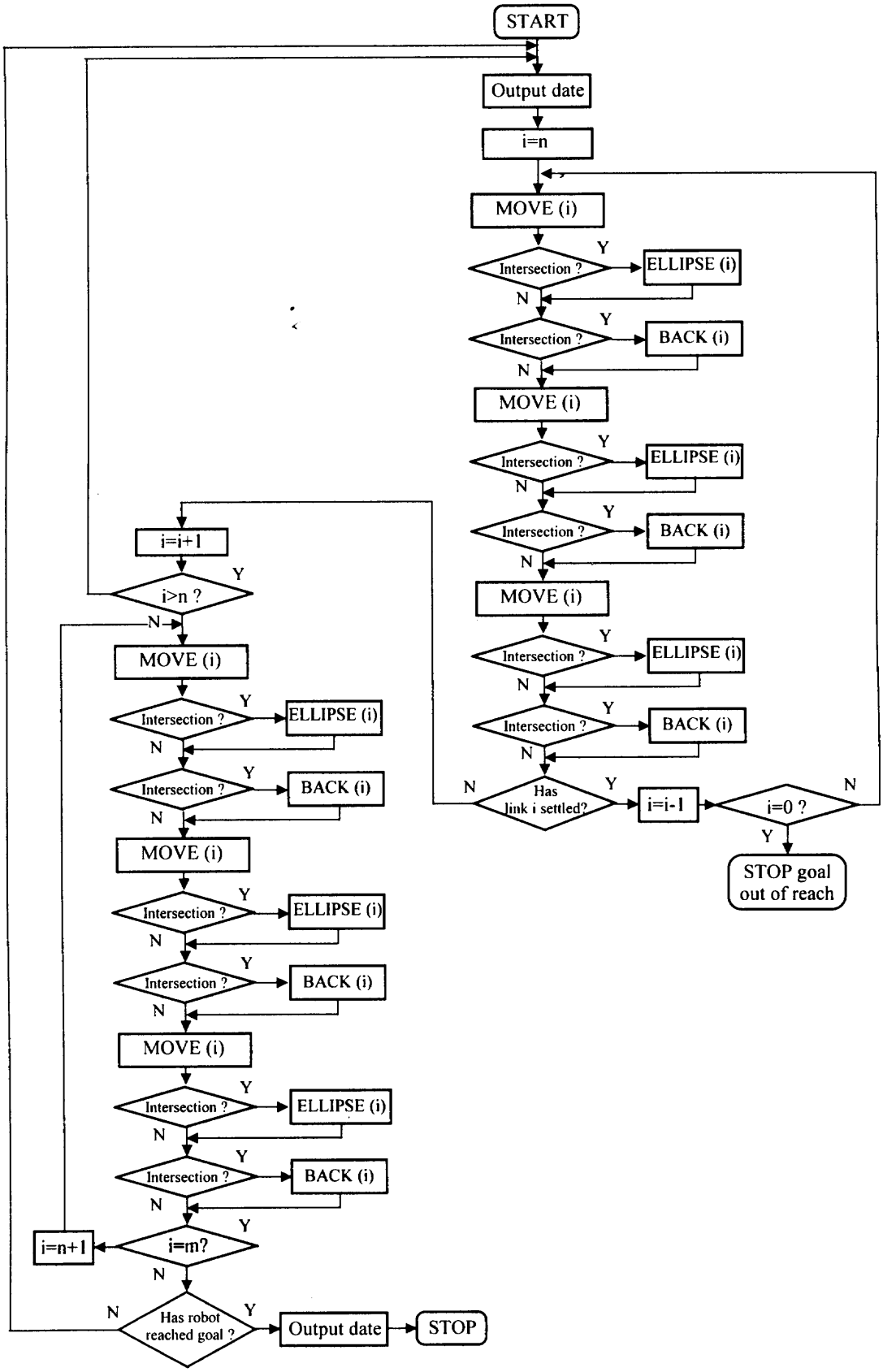

Fig. 8. Flowchart of robot control.

size is 100. The angular displacement that the potential field uses is 0.006 radian. The total minimum margin value is 1.5 while the total maximum value is 4.4. The co-ordinates of the obstacles and the radii of margin ellipses are given in Table II. Pre-computation for seven iterations takes 0.27 second while running time takes 3.50 seconds. Figure 11 shows a magnified view of an obstacle and a link using a fixed margin ellipse value.

The algorithm is independent of the number of links, with the third example showing the use of a twelve link

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

Table I. Co-ordinates of the centres and radii of the obstacles for the first example

	x	y	the radius u of the obstacles	the radius v of the obstacles
1. obstacle	300	692	88	88
2. obstacle	300	490	78	88
3. obstacle	370	250	68	88
4. obstacle	190	300	98	103
5. obstacle	540	470	158	88
6. obstacle	82	540	68	138

manipulator, Figure 12. The parameters for this example are: link lengths, 100; base coordinates, (580, 700); goal coordinates, (330, 260); working envelope 700×800; grid size 100; potential field angular displacement, 0.005 radians, minimum and maximum ellipse margins, 1.2, 2.5. The obstacle coordinates are given in Table III. Precomputation takes 0.83 seconds for 7 iterations, with a running time of 29.27 seconds.

4. DISCUSSION

The algorithm that has been presented, has been illustrated with some arbitrary obstacle fields. The key questions that this raises are: ''What are the situations in which it fails?'', ''How does the algorithm compare with other techniques?'' and, ''Can the algorithm cope with a dynamic environment?''

4.1 Failure modes

Failure is a natural state, but the frustration of knowing that an item is just out of reach is usually followed by the preparation of a plan B. This is very true of environments that are changing. However there are situations in which the algorithm is unable to find a route where one does exist. These are discussed below.

4.1.1 Intersections proximal to the joint. The displacements of the end-points of the links have been considered so far. Though, not only the end point of the link but also the whole link must avoid obstacles. For instance, after moving the kth link, the angular displacement which is necessary to move the middle point of the nth link is higher than the angular displacement for the endpoint of the nth link using equation (13). The nearer the point is to the start point of the link, the higher angular displacement for the link is required. If the angular displacement is determined with respect to one of the intersection points of the link with the obstacle, then very high angular displacements may be obtained and may cause very fast movements of

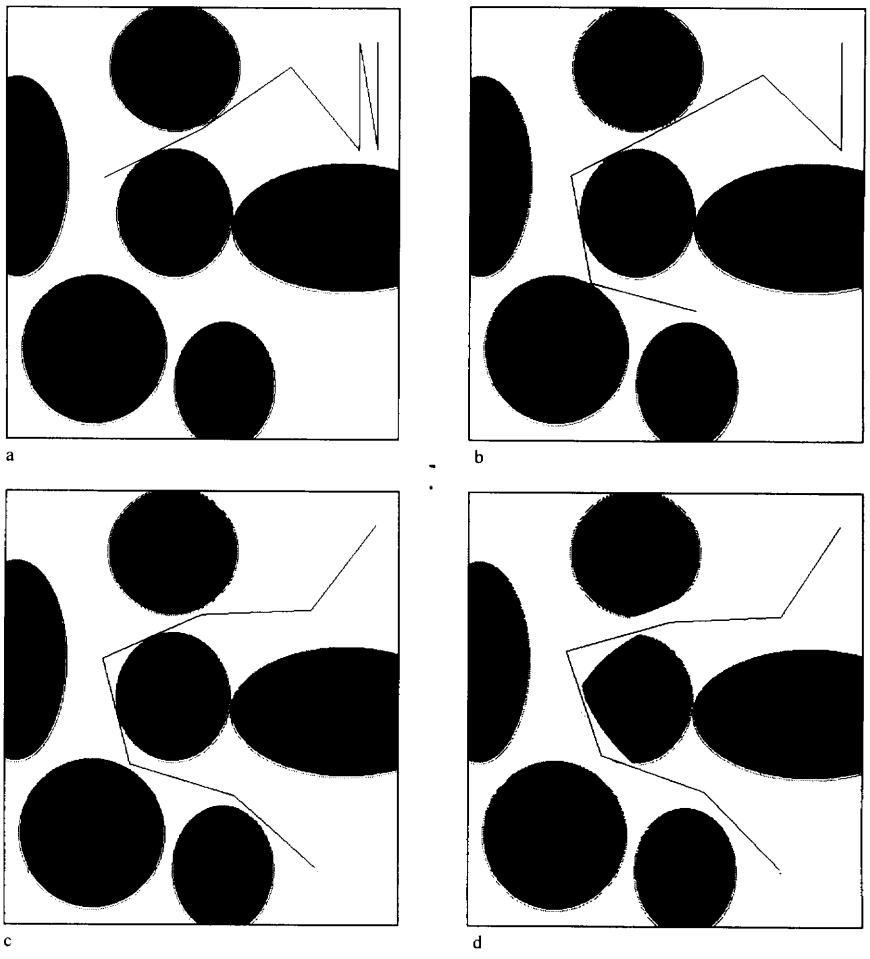

Fig. 9a–d. Example 1, six link manipulator.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

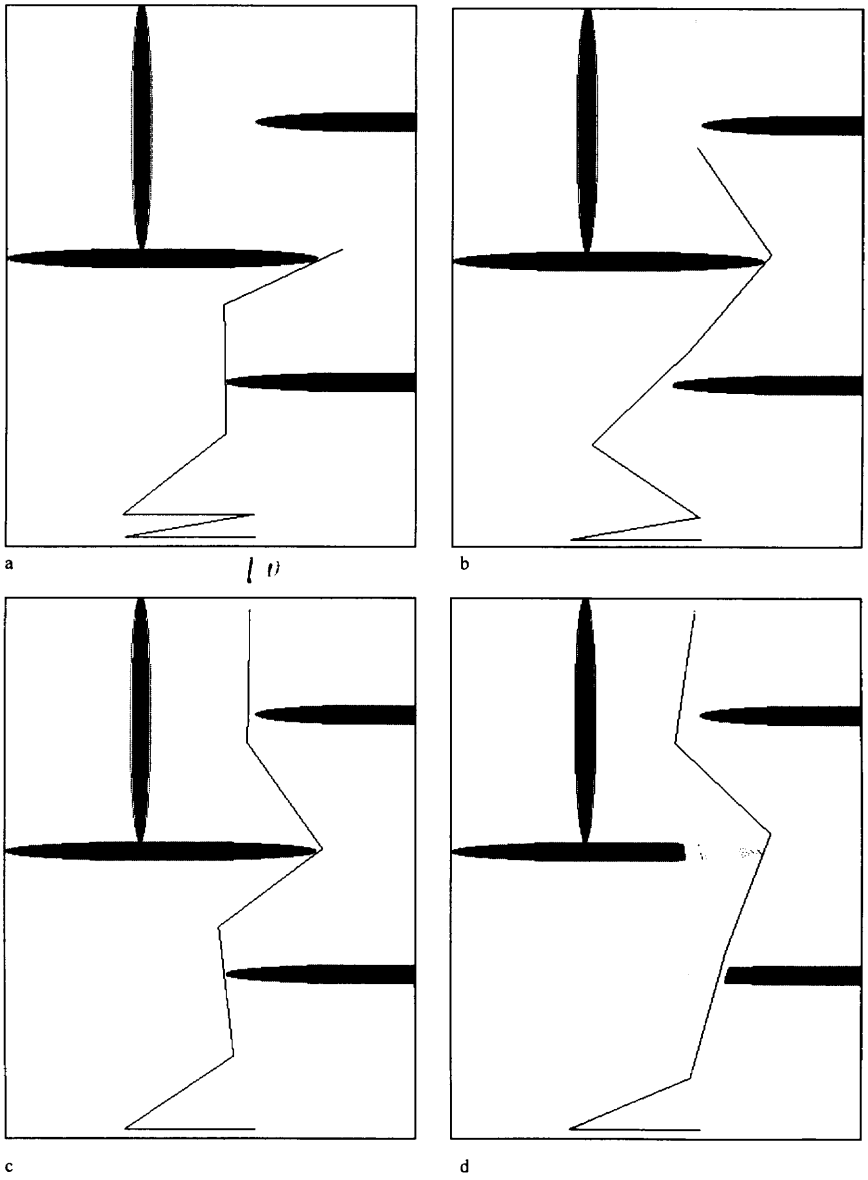

Fig. 10a–d. Example 2, six link manipulator.

the links. Yet, such movements are not needed at all. First, because the links follow the smooth path created by the potential field and never tend to move in different directions which may result in very fast movements for the links which intersect with the outside margin ellipse. Second the BACK function, when necessary, repeatedly moves the appropriate link back from the obstacle ensuring that no intersection will be left in a loop without being dealt with. Determining the angular displacements using only end-points of the links is enough to avoid obstacles.

Table II. Co-ordinates of the centres and radii of the obstacles for the second example

	x	y	the radius u of the obstacles	the radius v of the obstacles
1. obstacle	600	720	198	16
2. obstacle	250	500	248	16
3. obstacle	600	300	248	16
4. obstacle	100	716	16	198

4.1.2 Double intersections. Situations can occur when one link intersects with more than one margin ellipse

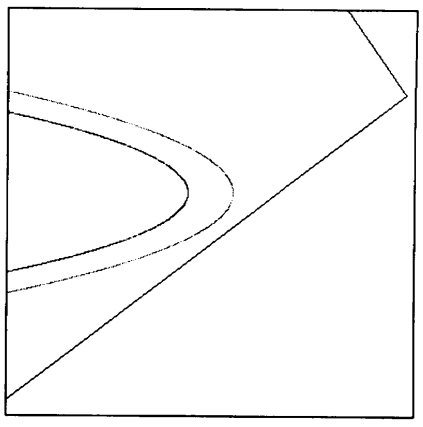

Fig. 11. Magnified view of an obstacle and a link.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

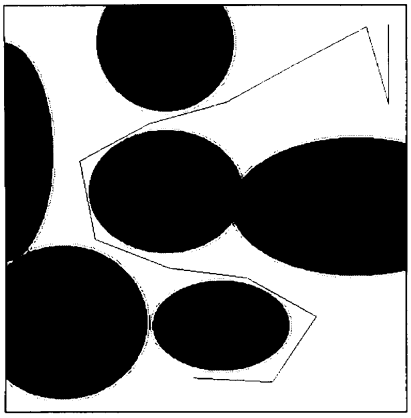

Fig. 12. Example 3, 12 link manipulator.

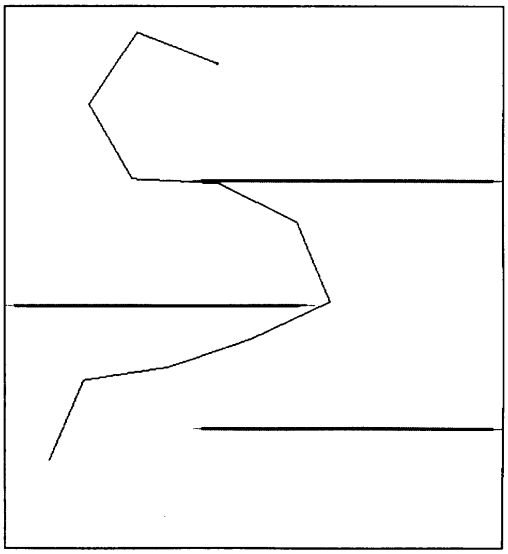

Fig. 13. The benchmark environment.

particularly when the passage is narrow. The link is checked for the first obstacle and moved accordingly. Because of the recursive feature of the BACK function, without moving any other link, it is checked for the second obstacle and moved in the opposite direction to the first movement, returning the link to the original position. To prevent this infinite loop, a count is made of the number of iterations, and the function is left after a specified number.

4.2 Comparison with other techniques

One difficulty with these results is that it is difficult to compare the performance with other techniques. It is usual that papers show situations that work, rather than cases that do not! There is therefore a need for a benchmark which can be used across the field to compare different algorithms.

The benchmark environment shown in Figure 13 is proposed as a standard that could be adopted. The key features are the constant corridor width of 50 and the 180 degree bend with a zero radius of curvature. The maximum link length that can be navigated is 100.

It is possible to analyse other environments by identifying the corridor width and evaluating the cornering required. However there are other variables that also contribute. For the algorithm given in this paper the position of the goal, the position of the fixed base of the manipulator and the number of links all have an effect.

Table III. Co-ordinates of the centres and radii of the obstacles for the third example

	x	y	the radius u of the obstacles	the radius v of the obstacles
1. obstacle	300	674	87.5	87.7
2. obstacle	300	490	97.5	77.5
3. obstacle	370	320	87.5	57.5
4. obstacle	170	325	107.5	97.5
5. obstacle	540	470	157.5	87.5
6. obstacle	95	540	62.5	137.5

The algorithm in this paper achieves this benchmark path with a maximum link length of 35. One significant reason for this length being comparatively short is that the pivot point of each link is located at the joint. For optimum behaviour the pivot point should be able to move, allowing each link to rotate near the obstacle intersection point. This would require a modification to the ELLIPSE function to modify the proximal as well as distal links of the manipulator.

4.3 Physical implementation

The idea of an algorithmic soft contact approach naturally leads to a physical implementation that allows real soft contact and uses such contact data, from appropriate sensors, to control motion. This is the approach being taken by Reznik and Lumelsky.[23]

The limitations of real sensors, e.g. finite range and inability to see through obstacles, is advantageous to the computational aspects, since the working field is greatly reduced in size. The sensors effectively blur the boundary between the use of global and local techniques.

This real time approach can be linked with off-line planning in which a wider working field may be available for human assessment of best routes. Routes could be blocked by using virtual objects, forcing the manipulator to take a ''better'' path.

Regarding the mechanical design of such a device a key issue is the balance between flexibility and accuracy of motion. There are very few examples in nature of long thin structures that are unsupported and flexible, which is not encouraging for devices working in air, unless soft contact can be used. Devices supported in a fluid, e.g. blood will be much simpler to design.

When designing thin devices, thin is taken as less than 10 mm diameter, there is the additional influence of the size of actuators. This leads to the need for externally actuated systems, driven through wires, or the use of

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

distributed actuation as is found with shape memory alloy or flexible pneumatic designs.

4.4 Dynamic environments

The issue of a changing environment is simply a matter of rate of change. For example, the human eye operates at about 25 Hz, which is matched in order of magnitude to the response of our motion systems (excluding LOW LEVEL reactions). A human is therefore unable to see or move to avoid a speeding bullet, but a thrown tennis ball can be easily caught. Taking another example: When we are confronted by a rotating door, we automatically pause to estimate the arrival of the gaps and then take the plunge. But if the fequency of rotation were variable most people would probably go and find another doorway.

This illustrates that the issue of a dynamically changing environment is meaningless, unless the changes are predictable within the response time of the device. The proposition of recalculating a potential field taking into account movement of obstacles (including other manipulators) is perfectly reasonable considering the availability of computing power. The part that is not trivial is the recognition that the field may have changed such that the manipulator is no longer on the correct path, but this is a re-statement of the backing-up task.

Therefore a dynamically changing environment brings only one extra requirement; that of being able to start from the wrong place.

5. CONCLUSION

An enhancement to a potential field route finding and path following algorithm has been presented. The enhancement is based on a soft-contact repulsion technique form geometrically defined obstacles and links. This overcomes some of the difficulties of potential field techniques which have been used for path finding, and allows the manipulator to use more of the available manoeuvring space, rather than being constrained to following a specified line as closely as possible.

The enhancement establishes a margin ellipses around an obstacle and models manipulator links as lines. When an intersection between a link and ellipse occurs the corresponding link is moved away from the obstacle. The soft-contact safety zone is sized with the maximum joint motions per increment to avoid actual collision. Moreover, all distal links to the link moved are checked and if necessary moved to establish a stable collision free configuration.

This method reduces the need for a well developed potential field, which in turn reduces the pre-computation time dramatically. The method also avoids the possibility of an obstacle being disregarded, and therefore hit, because of the finite number of potential field control points used along any link. Both the potential field technique and the new technique use the Cartesian space avoiding the high dimensionality of configuration space. There are no singularities and no local minima. The combination of a global based route finding algorithm and path following algorithm with the locally based collision avoidance algorithm provides a robust solution. The effectiveness of the proposed method has been shown in three computer simulations.

A method of benchmarking different techniques has also been presented that should enable these results to be compared with other path following algorithms for highly redundant manipulators.

References

1. E.S. Conkur & R.O. Buckingham, ''Comparison of Jacobian based techniques with potential field technique for controlling highly redundant robots'' *2nd Int. Mechatronics Design and Modelling Workshop,* Ankara, Turkey (Nov., 1995) pp. 161–170.
2. P. Chiacchio & S. Chiaverini, ''Closed-loop inverse kinematic schemes for constrained redundant manipulators with task space augmentation and task priority strategy'' *Int. J. Robotics Research* **10,** No. 4, 440–425 (1991).
3. Y. Nakamura, *Advanced Robotics Redundancy and Optimisation* (Addison-Wesley Pub. Company, Reading, Mass., 1991).
4. A. Graham & R. Buckingham, ''Real time collision avoidance of manipulators with multiple redundancy'' *Mechatronics* **3,** No. 1, 89–106 (1993).
5. S. Ma, S. Hirose & H. Yoshinada, ''Development of a hyper-redundant manipulator for maintenance of nuclear reactor'' *Advanced Robotics* **9,** No. 3, 281–300 (1995).
6. S. Seereeram & J.T. Wen, ''A global approach to path planning for redundant manipulators'' *IEEE Transactions on Robotics and Automation* **11,** No. 1, 152–160 (1995).
7. R.J. Schilling & G. Walker, ''Path tracking with the links of a planar hyper-redundant robotic manipulator'' *J. Robotic Systems* **12,** No. 3, 189–197 (1995).
8. D.N. Nenchev, ''Restricted Jacobian matrices of redundant manipulators in constrained motion tasks'' *Int. J. Robotic Research* **11,** No. 6, 584–597 (1992).
9. C. Klein & C. Huang, ''Review of pseudoinverse control for use with kinematically redundant manipulators'' *IEEE Trans. Sys. Man Cybernet.* **SMC 13,** 245–250 (1983).
10. A.A. Maciejewski & C.A. Klein, ''Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments'' *Int. J. Robotics Research* **4,** No. 3, 109–117 (1985).
11. N. Rahmanian-Shahri & I. Troch, ''Collision-avoidance control for redundant articulated robots'' *Robotica,* **15,** part 2, 159–168 (1995).
12. R.V. Mayorga, F. Janabi-Sharifi & A.K.C. Wong, ''A fast approach for the robust trajectory planning of redundant robot manipulators'' *J. Robotic Systems* **12,** No. 2, 147–161 (1995).
13. A Bagchi & H. Hatwal, ''Fuzzy logic-based techniques for motion planning of a robot manipulator amongst unknown moving obstacles'' *Robotica* **10,** part 6, 563–573 (1992).
14. M. Sharir, ''Algorithmic motion planning in robotics'' *IEEE Computer* **22,** No. 3, 9–20 (1989).
15. E.S.H. Hou & D. Zheng, ''Mobile robot path planning based on hierarchical hexagonal decomposition and artificial potential fields'' *J. Robotic Systems* **11,** No. 7, 605–614 (1994).
16. A. Denker & D.P. Atherton, ''No-overshoot control of robotic manipulators in the presence of obstacles'' *J. Robotic Systems* **11,** No. 7, 665–678 (1994).
17. A. Schweikard, ''A simple path strategy based on calculation of free sections of motions'' *Engng. Applic. Artif. Intell.* **5,** No. 1, 1–10 (1992).
18. K.K. Gupta & Z. Guo, ''Motion planning with many degrees of freedom: sequential search with backtracking'' *IEEE International Conference on Robotics and Automation* (1992) pp. 2328–2333.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press

19. K.K. Gupta & X. Zhu, ''Practical global motion planning for many degree of freedom: a novel approach within sequential framework'' *J. Robotics Systems* **12,** No. 2, 105–107 (1995).
20. J.Z. Li & M.B. Trabia, ''Adaptive path planning and obstacle avoidance for a robot with a large degree of redundancy'' *J. Robotic Systems* **13,** No. 3, 163–176 (1996).
21. D. Reznik & V. Lumelsky, ''Sensor-based motion planning for highly redundant kinematic structures: II The Case of a Snake Arm Manipulator'', *IEEE International Conference on Robotics and Automation* (1993) pp. 889–894.
22. D. Reznik & V. Lumelsky, ''Sensor-based motion planning for highly redundant snake robot'' *Advanced Robotics* **9,** No. 3, 255–280 (1995).
23. O. Khatib, ''Real-time obstacle avoidance for manipulators and mobile robots'' *Int. J. Robotics Research* **5,** No. 1, 90–98 (1986).
24. J. Tim & P. Khosla, ''Real-time obstacle avoidance using harmonic potential functions'' *IEEE International Conference on Robotics and Automation* (1990) pp. 790–796.
25. D.E. Koditschek, ''Exact robot navigation by means of potential functions: some topological considerations'' *IEEE International Conference on Robotics and Automation* (1987) pp. 1–6.

https://doi.org/10.1017/S0263574797000532 Published online by Cambridge University Press