March 1993

COLLISION FREE PATH PLANNING FOR A PLANAR REVOLUTE MANIPULATOR

Zohreh Erfan
Purdue University School of Electrical Engineering

Shaheen Ahmad
Purdue University School of Electrical Engineering

COLLISION FREE PATH PLANNING FOR A PLANAR REVOLUTE MANIPULATOR

ZOHREH ERFAN
SHAHEEN AHMAD

TR-EE 93-13
MARCH 1993

SCHOOL OF ELECTRICAL ENGINEERING
PURDUE UNIVERSITY
WEST LAFAYETTE, INDIANA 47907-1285

COLLISION FREE PATH PLANNING FOR A PLANAR REVOLUTE MANIPULATOR

Zohreh Erfan, Shaheen Ahmad
Real-Time Robot Control Laboratory,
School of Electrical Engineering,
Purdue University,
West Lafayette, Indiana 47907-1285, U.S.A.

Abstract

In this paper, we address the problem of path planning for a revolute manipulator, operating in a workspace filled with obstacles whose boundaries are enveloped by circle or possibly any other shape. The path planning approach presented here is developed in the manipulator joint space and consists of two strategies. The first strategy is used in tightly packed free spaces located between the workspace obstacles, this approach relies on constrained optimization. The second strategy is used whenever the manipulator is operating in free space regions which are not surrounded by the workspace obstacles, this strategy does not use optimization but relies on a global but safe approximation of the free space regions of the joint space. The manipulator uses the second strategy to perform collision-free gross motions. This division of the path planning approach reduces the chances that the manipulator will get stuck in a geometrical local minimum if the constrained optimization procedure is used to entirely determine the path planning procedure. With the approach taken here, the chance of finding a path for the manipulator from its given start point to its desired goal point in an automated manner is high in comparison to one which uses just one of the above two strategies.

1 Introduction

A manipulator with no ability to avoid obstacles in its workspace has to be taught every point on its trajectory so that the arm may be free from collision as the end-effector moves from a start point to a goal point. This path is stored during the teaching session and used each time the manipulator is moved from the start to the goal points. Obviously, this method is only good in cases of repetitive tasks where there is no variation in the position of either the start or

the goal point. A new path must be taught to the manipulator whenever the start and goal points change locations or obstacles positions are changed. The purpose of the research reported here is to devise an automatic path-planner. As two link revolutely jointed planar manipulators are used in large numbers in assembly systems, transfer lines and in automatic machine loading, we developed trajectory planning schemes for such robots.

The robot motion planning problem ([1-19], [22-25]) has been extensively addressed in the literature. Different aspects of this problem including the path planning problem has been investigated by computer scientists, engineers and mathematicians. Theoretical investigations have mostly led to some general solutions to this problem [22] which would require tremendous amount of computations. Algorithms which are practically implementable ([1-19], [23]) have also emerged. Brooks [3] approached the path planning problem for a moving polygon with translational and rotational movements among polygonal obstacles by representing the cartesian free space in between the obstacles as a set of free space corridors (generalized cones), followed by determination of the range of orientations which the moving object may safely take along each cone axis ("spine"), of these corridors. A solution path was found by connecting the given start point to the desired goal point, the path consisted of a sequence of connected corridors, the translational movements were performed along the spines, and rotational movements were performed at the spines intersections. Later, Brooks [4] extended this free space representation to a three degree revolute manipulator with an additional rotational degree of freedom at the end-effector. The arm was moving among polyhedral obstacles, and planning of collision-free motions of the manipulator for pick and place operations were performed. Lozano-Perez [1] approached the path planning problem for a cartesian manipulator among polyhedral obstacles in the configuration space of the robot, note that the configuration space of a cartesian manipulator is the same as its three dimensional cartesian workspace. The path planning problem was then reduced to that of path planning for a point among the grown configuration space obstacles. The grown configuration space obstacles were formed by shrinking the moving object to a point and growing the original obstacles by the size and shape of the original moving object. A path was then found by a graph search algorithm [2] which determined the shortest path in between the start and goal points by connecting these points through the

vertices of the grown configuration space obstacles. The path planner only allows those path segments which do not pass over any of the grown obstacles, i.e. a path segment is allowed to be on an edge of the grown configuration space obstacles but must not pass through the configuration obstaclss. Faverjon [6] found a collision-free path for a three degree of freedom revolute manipulator among polyhedral obstacles by building an octree which hierarchically represented the joint space configurations of the robot. The octree was then searched to find a collision-free path for the manipulator. Brooks and Lozano-Perez [9] also developed a path-finding algorithm for a polygonal object with two translational and one rotational movements among polygonal obstacles, based upon using a hierarchical subdivision of the three dimensional configuration space of the moving object. Luh and Campbell [5] determined a collision-free path for the first three degrees of freedom of a Stanford arm with a prismatic joint among stationary polyhedral obstacles. They determined infeasible regions of the workspace which become unavailable to the manipulator's end-effector, due to the prismatic link of the arm colliding with an obstacle. These infeasible regions were approximated by polyhedral obstacles and were called 'pseudo-obstacles''. As a result, the original stationary workspace obstacles and the pseudo-obstacles of the workspace had to be considered in order to determine a collision free path for the arm. An algorithm was developed by Luh and Campbell [5] which found the shortest path from the initial position of the end-effector to its desired goal point, along the edges of the original and the pseudo obstacles in the workspace. Khatib [10] addressed the issue of real-time obstacle avoidance for an articulated manipulator in an environment of static obstacles. The end-effector model was chosen to be rotation-invariant. Collision between the manipulator and the obstacles were prevented by forces which repelled the arm from making contact with the obstacle surfaces. The repelling forces were a function of the minimum distance between the arm and the obstacles. Khatib [10] also used an attractive field to guide the manipulator's end-effector to its goal point. The sum of the attractive and repelling forces determined the end-effector motions in the workspace. If the sum of the attractive and repelling forces becomes zero, the end-effector experiences no further motions and thus becomes locally trapped at a point of minimum force which is not the goal point. By appropriate adjustment of the repelling and attractive forces these local force

minima on the end-effector can be avoided. In another approach to the problem of path planning, Lozano-Perez [13] used the configuration space of a revolute manipulator with three degrees of freedom, in order to plan a collision-free path for the robot among polyhedral obstacles. In order to build the configuration space (C-space) for this robot, Lozano-Perez first built the C-space for the first two joints, ignoring the third joint initially. Next he approximated the free space regions of this two dimensional C-space, and found a collision-free path for the first two links of the robot. Next, he built a subset of the complete C-space for the robot (i.e. for all the joints (q_1,q_2,q_3)) assuming that the first two joints move according to the path found in the above two dimensional C-space. Once again he performed a search. This was done in the above subset of the C-space, in order to find a collision-free path for the third link of the robot. Now looking in the area of multiple robot collision avoidance, Freund and Hoyer [15] devised a real time heuristic trajectory modification scheme. Their trajectory modification scheme was based on hierarchical master-slave strategy. The master arm's trajectory was never altered, however if the slave arm would enter the master arm's wokspace, then its path would be altered to avoid a possible collision based on a number of rules which was dependent on the robot kinematics. Erdmann and Lozano-Perez [11] also addressed the multi-robot collision avoidance by expanding the C-space collision avoidance planning strategy developed earlier [1], to include time. Then one of the robots could be regarded as an obstacle while the other arm's trajectory was planned in each C-space time slices. Stonier [18], and Erfan and Ahmad [19] also developed Lyapunov based on-line control strategies to coordinate multiple robot trajectories. Both Stonier's and Erfan and Ahmad's control laws prevented the collision of two arms, while each arm was able to move to its goal.

1.1 Work Addressed in this Paper

In this paper, we will address the problem of determining a collision-free path for a planar revolutely jointed manipulator, in a workspace filled with obstacles. We will assume the initial and final configurations of the manipulator are collision-free. In this paper, we subdivide the path planning into two strategies, the global path planning and the local path planning. Path planning is performed in the joint space by iteratively calling the global and the local path planners. The approach developed here uses an approximate but safe global

view of the workspace obstacles in the joint space. This information is used to find a large manoeuver from the current configuration to a collision-free configuration which is closer to the goal point joint configuration than the current configuration, and eventually to the goal point. Whenever the robot is operating in a cluttered region, a local path is generated to steer the manipulator away from the boundaries of the obstacles. The local path planner uses the exact knowledge of the joint space obstacles boundaries. The local planner guides the arm to the joint space goal configuration or to an intermediate collision-free configuration at which the global motion planner will take over the path planning operations. In case the goal point joint configuration is situated in some other cluttered region, then the global planner will hand the path planning task over to the local planner, after the global planner has determined a path to an intermediate target point near the goal joint space configuration. This process of iteratively calling the local and global motion planners are carried out until a path to the goal point is reached.

This paper is organized into eight sections. In section one, we have presented a brief overview of some of the collision avoidance work completed to date for a manipulator working in a workspace filled with stationary obstacles. The manipulator kinematic models as well as the descriptions of the obstacles are presented in section two. In section three, the calculations of the forbidden or infeasible regions in the joint space which result in a collision with an obstacle are presented. The characteristics of the infeasible regions are described in section four. In section five, a rectangular approximation to an infeasible region and its equations are derived. The polygons which result from taking the union of the rectangles which bound the infeasible regions are also described in section five. In section six, the path planning procedure which consists of a global path planning scheme and a local path planning scheme are described. Several simulations which show how the two path planning schemes cooperatively find a collision-,free path from a given collision-free joint space configuration to a desired collision-free joint space goal configuration are given in section seven. A summarizing and concluding discussion with the needs for future work is presented in section eight.

2 Manipulator Kinematic Models and Obstacle Descriptions

2.1 Robot Kinematic Models

Planar SCARA (Selective Compliance Articulated Robot Arm) robot arms are used in large numbers for machine loading and unloading and in assembly. Typically such robots have four degrees of freedom (see Figure 1) made up of two revolute joints working in the horizontal plane, followed by a third prismatically jointed link allowing vertical motion, followed by a fourth revolute joint with an attached gripper. For collision avoidance purposes, this robot can be viewed as a two degree of freedom manipulator, as the Z-axis prismatic motions correspond directly to the height above an obstacle in the workspace. The fourth revolute joint is used to orient the parts, as the orientation maneouvers can be carried out above the obstacles or in the free space of the XY plane, it is possible in many cases to perform the path planning by modelling the SCARA arm as a planar two link revolutely jointed manipulator. Notice that although we are motivating and illustrating the collision avoidance strategy here with SCARA type robot arms, the main idea expressed in this work related to global-local planning is also applicable to robots of other types.

The kinematic model of the robots under consideration is thus that of a two degree of freedom planar robot with revolute joints. Let us represent the end-effector workspace position as X_e, the corresponding joint angles of the arm is $q = (q_1, q_2)^T \in \mathcal{R}^2$. We assume the robot has links of lengths l_1 and l_2 respectively for its first and second links (see Figure 1). The position of the end point of the first link of the manipulator is given by $X_{e1} = (l_1 \cos q_1 , l_1 \sin q_1)^T$, then the end-effector position is given by $X_e = X_{e1} + l_2 (\cos q_2 , \sin q_2)^T$. For an end-effector position $X_e(x,y)$, given we denote the difference $(q_2 - q_1)$ in joint angles by q_{2-1}, the arm angles are calculated as follows:

$$q_{2-1} = \pm \cos^{-1} \frac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2}, \quad (1)$$

$$q_1 = \text{atan2} (- l_2 s_{2-1} x + (l_1 + l_2 c_{2-1}) y , (l_1 + l_2 c_{2-1}) x + l_2 s_{2-1} y), \quad (2)$$

$$q_2 = q_{2-1} + q_1 , \quad (3)$$

given $s_{2-1} = \sin q_{2-1}$ and $c_{2-1} = \cos q_{2-1}$. With the negative sign of the arc-cosine function, one obtains an angle q_2 which corresponds to a left hand configuration (LC) of the arm, and the positive sign results in a right hand configuration (RC)

of the arm. Any of the two joint angles pairs (q_1,q_2) as calculated from above may be used to reach X.

2.2 Obstacle Descriptions

Obstacles may have polygonal or any other shape. However, it is not particularly desirable to have the manipulator pass very close to the obstacle boundaries, and thus the smallest circle which bounds an original non-circular obstacle is used to approximate the obstacle. Every point on the manipulators' links has to be located outside the circular obstacle to ensure a collision free trajectory with that obstacle. There are some interesting properties which arise if the obstacle is assumed to be circular due to its symmetric nature, furthermore a circular obstacle can be represented by one function as opposed to a number of discontinuous functions needed to represent an obstacle of any arbitrary shape. These features facilitate implementation and demonstration of the planning procedure developed in this paper. The planning procedure developed here is equally valid if the obstacle is circular or otherwise.

2.3 Calculating the Infeasible Joint Space Regions

All the objects which are within a radius of (l_1+l_2) from the base of the manipulator are obstacles, these obstacles have to be considered when we determine a trajectory for this robot. The infeasible regions are defined to be those regions of the joint space which cause a collision of the arm with the obstacles. There exists one region for each circle present in the XY workspace of the arm, provided that the circle is an obstacle to the arm.

3.1 Range of Collision for Link one

Consider a circular obstacle O_i of radius r_i and center (x_{ci},y_{ci}) with respect to the base of the manipulator (see Figure 2). Let R_i denote the distance from the base to the obstacle's center, i.e. $R_i=\sqrt{x_{ci}^2+y_{ci}^2}$. Now given $R_i \geq r_i$, let $t_{1i} = \sqrt{R_i^2 - r_i^2}$ be the length of the tangent line drawn from the origin of the workspace (the manipulator's base), to the obstacle. The collision ranges for link one can be determined from one of the three cases below, such that $q_1^{Li} \leq q_1 \leq q_1^{Ui}$, where q_1^{Li} and q_1^{Ui} are respectively the lower and upper q_1 boundaries of the infeasible region for the obstacle O_i. These boundary angles

are calculated by intersecting two circles: one circle is the trace of the end point of the first link, and the other circle has its center located at the center of obstacle O_i and a radius equal to (r_i+l_2).

Case 1: $l_1 \geq t_{1i}$

In this case, the range of collision for link one with obstacle O_i is, $\theta_i - |\theta_{\delta i}| \leq q_1 \leq \theta_i + |\theta_{\delta i}|$, where, $\theta_i = \text{atan2}\,(y_{ci}, x_{ci})$ and $\theta_{\delta i} = \text{atan2}\,(r_i, t_{1i})$.

Case 2: $t_{1i} > l_1 \geq R_i - r_i$

In this case, the range of collision of link one with obstacle O_i is given by,

$$q_1^{Li} = \theta_i - |\theta_{\delta i} - \theta_{\Delta i}| \leq q_1 \leq \theta_i + |\theta_{\delta i} - \theta_{\Delta i}| = q_1^{Ui} \qquad (4)$$

where $|\theta_{\delta i} - \theta_{\Delta i}| = \cos^{-1}\left((R_i^2 - r_i^2 + l_1^2) / 2R_i l_1\right)$. Note that at point $P(x_{e1}, y_{e1})$ (Figure 2), $q_1 = q_1^{Li}$ and the end of the link one is in contact with obstacle boundary TMT'. The point $P'(x'_{e1}, y'_{e1})$ is symmetrically located on the obstacle about the line oc_i.

Case 3: $l_1 < R_i - r_i$

In this case, link one cannot reach obstacle O_i and therefore there is no collision for this link with this obstacle, at any orientation.

3.2 Range of Collision for Link Two

Let the range $\{q_{1fR}\}$ be the ranges of joint angle q_1, which is free from link one collision with any obstacle. To calculate the range of collision for link two, we calculate at each discrete q_1 from the set $\{q_{1fR}\}$, the ranges of q_2 values which would cause a collision between link two and the obstacles in the workspace. For each obstacle within the reach of the arm, a pair of 'q2 coordinate values, q_2^L and q_2^U, is found such that $q_1 \in \{q_{1fR}\}$. The range of q_2, $q_2^L \leq q_2 \leq q_2^U$ is the collision range for link two, at that q_1 coordinate value. For the arm to be collision free at a collision free q_1 angle, the orientation of link two must be outside link two's collision range.

As the end point of the first link at a feasible q_1 angle, is $(x_{e1}, y_{e1})^T$ and the end point of the second link is $(x_{e2}, y_{e2})^T$, then given $s_i = \sin q_i$ and $c_i = \cos q_i$ for $i=1,2$, the distance d_{1i} between link one end point and the obstacle O_i, is given by,

$$d_{1i}^2 = (x_{ci} - x_{e1})^2 + (y_{ci} - y_{e1})^2 = (x_{ci} - l_1 c_1)^2 + (y_{ci} - l_1 s_1)^2 . \quad (5)$$

The ranges of angles q_2 that causes link two collisions with obstacle O_i, while link one is fixed at a $q_1 \in \{q_{1fR}\}$, can be found from d_{1i} and is given below:

Case 1: $d_{1i} - r_i \le l_2 \le \sqrt{d_{1i}^2 - r_i^2}$

In this case, link two's end point touches the obstacle, thus,

$$(x_{ci} - x_{e2})^2 + (y_{ci} - y_{e2})^2 = r_i^2 \quad (6)$$

$$\text{where, } (x_{e2}, y_{e2}) = (x_{e1} + l_2 c_2, y_{e1} + l_2 s_2) = (l_1 c_1 + l_2 c_2 , l_1 s_1 + l_2 s_2) . \quad (7)$$

Substituting for (x_{e2}, y_{e2}) from equation (7) and also substituting equation (5), into equation (6), we obtain,

$$(x_{ci} - l_1 c_1) c_2 + (y_{ci} - l_1 s_1) s_2 = \frac{l_2^2 + d_{1i}^2 - r_i^2}{2 l_2} \quad (8)$$

Given $c\beta = \cos \beta_i$ and $s\beta = \sin \beta_i$ where β_i is some angle, from equation (5), we may let $(x_{ci} - l_1 c_1) / d_{1i} = s\beta$ and $(y_{ci} - l_1 s_1) / d_{1i} = c\beta$, which gives us $\beta_i = \text{atan2}(x_{ci} - l_1 c_1, y_{ci} - l_1 s_1)$. This allows us to rewrite equation (8) of the form, $s\beta\, c_2 + c\beta\, s_2 = (l_2^2 + d_{1i}^2 - r_i^2) / (2 l_2 d_{1i})$. Thus we can solve for q_2 as,

$$q_2 = \text{atan2}\left(d_1 , \pm \sqrt{(2 l_2 d_{1i})^2 - d_1^2}\right) - \&, \quad (9)$$

where $d_1 = l_2^2 + d_{1i}^2 - r_i^2$. The range of collision for the second link is given by,

$$\text{atan2}\left(d_1 , \sqrt{(2 l_2 d_{1i})^2 - d_1^2}\right) - \beta_i \le q_2 \le \text{atan2}\left(d_1 , -\sqrt{(2 l_2 d_{1i})^2 - d_1^2}\right) - \beta_i . \quad (10)$$

Case 2: $l_2 > \sqrt{d_{1i}^2 - r_i^2}$

In this case, link two becomes tangent to the obstacle O_i when it first collides with it. Let the coordinates of this point be (x_m, y_m), then $(x_{ci} - x_m)^2 + (y_{ci} - y_m)^2 = r_i^2$, and $(x_m, y_m) = (x_{e1} + (\sqrt{d_{1i}^2 - r_i^2})\, c_2 , \; y_{e1} + (\sqrt{d_{1i}^2 - r_i^2})\, s_2)$.

Substituting the coordinates (x_m, y_m) into the circle equation and following a similar derivation to case 1 results in,

$$q_2 = \text{atan2}\left(\sqrt{d_2} , \pm r_i\right) - \beta_i \quad (11)$$

where $d_2 = d_{1i}^2 - r_i^2$, and $\beta_i = \text{atan2}(x_{ci} - l_1 c_1, y_{ci} - l_1 s_1)$. The range of collision for the second link is given by,

$$\text{atan2}\left(\sqrt{d_2} , r_i\right) - \beta_i \le q_2 \le \text{atan2}\left(\sqrt{d_2} , - r_i\right) - \beta_i . \quad (12)$$

Case 3: $d_{1i} - r_i > l_2$

There are no collisions for the second link with obstacle O_i and all orientations of angle q_2 are feasible as the object is out of reach of link two.

3.3 Equations of the Infeasible Joint Space Region

From the above case descriptions we can functionally describe the interior and the boundary of the infeasible region IR_i in the joint space, if $R_i > (r_i + l_1)$ (no collision with link one), by two functions in each case, as follows,

Case 1: $g_i(q) = -q_2 + \text{atan2}(d_1, +\sqrt{(2 l_2 d_{1i})^2 - d_1^2}) - \beta_i \le 0$,

or, $g_i(q) = q_2 - \text{atan2}(d_1, -\sqrt{(2 l_2 d_{1i})^2 - d_1^2}) + \beta_i \le 0$. (13)

Case 2: $g_i(q) = -q_2 + \text{atan2}(\sqrt{d_2}, +r_i) - \beta_i \le 0$,

or, $g_i(q) = q_2 - \text{atan2}(\sqrt{d_2}, -r_i) + \beta_i \le 0$. (14)

4 Properties of the Infeasible Regions

Lemma 1: As the manipulator's inverse kinematic function is a one-to-two map, thus for each infeasible point in the joint space, there exists another infeasible point which is symmetrically located about the point $(q_1, q_2) = (\theta_{ci}, \theta_{ci})$ where $\theta_{ci} = \text{atan2}(y_{ci}, x_{ci})$ and (x_{ci}, y_{ci}) are the coordinates of the obstacle O_i's center in the workspace (Figure 3).

Proof: In Figure 3, the two triangles oAc_i and $oA'c_i$ share a common side, and the two other sides are equal in length, i.e. $||\overline{oA}|| = ||\overline{oA'}|| = l_1$ and $||\overline{Ac_i}|| = ||\overline{A'c_i}|| = l_2 + r_i$, thus the two triangles are the same. Thus their interior angles are equal, i.e. $\angle Aoc_i = \angle A'oc_i$, and $\angle Ac_io = \angle A'c_io$. Consider the joint angle q_1 for the ***left*** hand arm ***configuration*** (LC) at P, $q_{1P\,LC} = \theta_{ci} + \angle Aoc_i$ and for the ***right*** hand arm ***configuration*** (RC) at P', $q_{1P'RC} = \theta_{ci} - \angle A'oc_i$, thus the two angles are symmetrically located about the angle $q_1 = \theta_{ci}$. Now if we draw two lines L and L' starting at point A and A', respectively, with each line parallel to the radial line oc_i, then at any point F selected on the line L and point F selected on L', as shown in Figure 3, we have $\angle Ac_io = \angle c_iAF$. Similarly $\angle A'c_io = \angle c_iA'F'$. Thus $\angle c_iAF = \angle Ac_io = \angle A'c_io = \angle c_iA'F'$. Therefore the left hand q_2 joint angle ($\angle X'Ac_i$) with end-effector at P is $q_{2P\,LC} = \theta_{ci} - \angle c_iAF$ and the

right hand q_2 joint angle ($\angle X''A'c_i$) with end-effector at P' is $q_{2P'RC} = \theta_{ci} + \angle c_iA'F'$. Thus $q_{2P\,LC}$ and $q_{2P'RC}$ are symmetrically located about the angle $q_2 = \theta_{ci}$. Therefore for a point P on the object, there exists a point P' on the object, such that $(q_{1P\,LC}, q_{2P\,LC})$ and $(q_{1P'RC}, q_{2P'RC})$ are symmetric to one another with respect to the angular coordinates $(\theta_{ci}, \theta_{ci})$ in the joint space. That is the (q_1, q_2) coordinates of the points P and P' are symmetrically located about the point $(\theta_{ci}, \theta_{ci})$, as proved above. AAC

Corollary 1: The centroid of the joint space infeasible region IR_i is located on the point $(\theta_{ci}, \theta_{ci})$. Furthermore assuming that all the workspace obstacles O_i $i=1,...,m$ are within the reach of the arm, then all of these obstacles have infeasible regions whose centroids are located on the $\theta_D=\pi$ line in the joint space, where θ_D is the angle in between the two links. For an arm configuration with joint angles (q_1, q_2), $\theta_D = \pi+q_1-q_2$ (for RC) and $\theta_D = \pi-q_1+q_2$ (for LC).

Proof: From lemma 1, given θ_{ci} = atan2 (y_{ci}, x_{ci}) is the angle that the center of the obstacle O_i (x_{ci}, y_{ci}, r_i) makes with the positive X-axis, then the point $(\theta_{ci}, \theta_{ci})$ in the joint space is the centroid of the infeasible region for obstacle O_i. It is easy to see that for the joint angles pair $(\theta_{ci}, \theta_{ci})$, the angle in between the two links, θ_D, is equal to π radians, as the q_1 and q_2 joint angles are both measured with respect to the positive X-axis. As the joint space infeasible region for the workspace obstacle O_i i=l,...,m, has its centroid at the point $(q_1, q_2)=(\theta_{ci}, \theta_{ci})$, thus the centroid of all the infeasible regions IR_i $i=1,...,m$ fall on the $\theta_D=\pi$ radians line. Note that the half-plane above this line represent the right hand configurations (RC) of the arm and the half-plane below it represent the arm left hand configurations (LC) (see Figure 4). We should also note that the infeasible regions may overlap despite the fact that their corresponding obstacles do not overlap in the workspace. AAC

Corollary 2: For any point $P \in O_i$ reachable by the arm, assume that the arm may reach P using a left hand configuration with the joint angles $(q_{1P\,LC}, q_{2P\,LC})$, and similarly P may be reached by the arm using a right hand configuration with the joint angles $(q_{1P\,RC}, q_{2P\,RC})$, then $(q_{1P\,LC}, q_{2P\,LC})$ and $(q_{1P\,RC}, q_{2P\,RC})$ are symmetric to each other about the point $(q_1, q_2)=(\theta_{cP}, \theta_{cP})$, where θ_{cP} is the angle $\overline{oP}$ makes with the positive X axis.

Proof: Using a similar proof as the one given for Lemma 1, one may show that (q_{1PLC}, q_{2PLC}) and (q_{1PRC}, q_{2PRC}) are symmetrically located about $(q_1,q_2)=(\theta_{cP},\theta_{cP})$, or $\theta_{cP} = 1/2\,(q_{1PLC}+q_{1PRC}) = 1/2\,(q_{2PLC}+q_{2PRC})$. This results in $-q_{2PRC}+q_{1PRC} = q_{2PLC}-q_{1PLC}$, which leads to the fact that the arm configurations (q_{1PLC}, q_{2PLC}) and (q_{1PRC}, q_{2PRC}) have the same θ_D angular value.

As a result of lemma 1, we only need to calculate half the boundary points of the infeasible region. The other half may be readily obtained using the symmetric characteristic of the infeasible region. The method of calculation of the boundary points to infeasible regions is iterative. The boundary points are calculated at each discrete feasible values of q_1.

41 The Feasible Joint Solution Space

The feasible joint solution space is the space in which the joint angles $q(q_1,q_2)$ are free from collision. Joint limits also restrict the feasible solution space. We notice that when manipulator passes $\theta_D=0$ (on line L_1 or L_2 in Figure 4), it flips the arm configuration and effectively links collide with each other, which in reality is infeasible. To avoid confusion, we therefore limit the feasible region to the region in between the lines L_1' and L_2' where $\theta_D>0$. The interior to the regions in between L_0 and L_1 and also in between L_0 and L_2 denote the RC and LC arm configurations, respectively. Both the RC and LC regions satisfy the range $\theta_D>0$. Note that RC region is characterized by $0< q_2-q_1 <\pi$ and the LC region has $-\pi< q_2-q_1 <0$.

5 Rectangular Approximation of the Infeasible Region

The shape and the size of the infeasible region varies with the location and size of the obstacle and the link lengths. A safe rectangular approximation for each infeasible region will be used to aid the path planning scheme (see Figure 5). The sides of the rectangles will be made parallel to one of the two axes θ_T and θ_D, with the positive direction of the θ_T axis given by the unit vector $\hat{\theta}_T = \blacksquare\ (1,1)^T$. The θ_D axis is along the direction $\pm 1/\sqrt{2}\,(1,-1)^T$ in joint space and is orthogonal to the θ_T axis. As before, the θ_D angle denotes the interior angle between the two links when the arm is at the joint angle pair (q_1,q_2), and may be calculated as follows,

$$\theta_D = \begin{cases} \pi - q_1 + q_2 & \text{if } q_1 > q_2 \quad \text{(LC)} \\ \pi + q_1 - q_2 & \text{if } q_1 < q_2 \quad \text{(RC)} . \end{cases} \tag{15}$$

The two sides of the **rectangle** which are parallel to the θ_D axis, are each a distance of d_{max} on each side of the infeasible region's centroid $(\theta_{ci},\theta_{ci})$. The other two sides of this rectangle are parallel to the θ_T axis and they are located **at** $\theta_D=\theta_{DBi}$ where θ_{DBi} **is** the largest θ_D angle which bounds the infeasible region IR_i, as shown in Figure 3. The θ_{DBi} angle is the interior angle between the two links, when the end-effector is in contact with the obstacle at point **M** on the line oc_i (see Figure 3). Notice that the entire boundary of the infeasible region IR_i, for the workspace obstacle O_i, is bounded along the θ_D axis, by the angle θ_{DBi} only if $R_i>(r_i+\max(l_1 , | l_1-l_2 |))$. Assuming that this condition is satisfied, then it is easy to see that a smaller θ_D angle than θ_{DBi} does not cause a collision between the arm and the obstacle, and a larger θ_D angle than the boundary angle θ_{DBi} causes a collision with the obstacle. Notice that if the links are kept at this $\theta_D=\theta_{DBi}$ angle, while q_1 is varied between $[-\pi,\pi]$, only one point of contact **M,** with the obstacle contour, results at the end-effector. The point M on the obstacle can be calculated as, $x_M = x_{ci} - r_i \cos\theta_{ci}$ and $y_M = y_{ci} - r_i \sin\theta_{ci}$, where $\theta_{ci} = \text{atan2}(y_{ci} , x_{ci})$. The boundary angle θ_{DBi} for the obstacle O_i (and the infeasible region IR_i) can be calculated from (15) after the joint angles (q_{1M},q_{2M}) corresponding to the end-effector at point M has been found from the inverse kinematics.

The equation of the line L which passes through the centroid of the infeasible **region,** $(\theta_{ci},\theta_{ci})$, and is **parallel** to θ_D axis **is** given by, $q_1 + q_2 = 2\,\theta_{ci}$ (Figure 5). The slope of the line L is -1, and the point B is located at $(0,2\theta_{ci})$. The unit normal to the line L is $\vec{n} = \pm(1/\sqrt{2})\,(1,1)^T$. Take any point P with coordinates (q_1,q_2) on the boundary of the infeasible region IR_i as shown in Figure 5. Thus the maximum distance d_{max} between point P on the infeasible region boundary $q \in IR_i$ and the line L is then calculated as,

$$d_{max} = \max_{q \in IR_i} \left\{ \frac{|\vec{BP} \cdot \vec{n}|}{\|\vec{n}\|} \right\} = \max_{q \in IR_i} \left\{ \frac{|q_1 + q_2 - 2\,\theta_{ci}|}{\sqrt{2}} \right\} . \tag{16}$$

5.1 Equations of the Boundaries of the R

The points P_n and P_n' on the lines LP_n and LP_n' have coordinates $P_n\,(\theta_{ci} + (d_{max})/\sqrt{2}\,,\ \theta_{ci} + (d_{max})/\sqrt{2})^T$ and $P_n'\,(\theta_{ci} - (d_{max})/\sqrt{2}\,,\ \theta_{ci} - (d_{max})/\sqrt{2})^T$ (see Figure 5). The equations of the lines LP_n and LP_n' which bound the infeasible region and pass through P_n and P_n' respectively, are $q_1 + q_2 = 2\,\theta_{ci} + \sqrt{2}\,d_{max}$, and $q_1 + q_2 = 2\,\theta_{ci} - \sqrt{2}\,d_{max}$, respectively. The equations of the lines S_1 and S_2 (Figure 5) are given by $q_2 - q_1 = -\pi + \theta_{DBi}$ and $q_2 - q_1 = \pi - \theta_{DBi}$. Using the equations of the lines LP_n, LP_n', S_1 and S_2, we can find the coordinates of the vertices A, B, C and D (see Figure 5). They are A($(t_1-t_0)/2$, $(t_1+t_0)/2$), B($(t_1+t_0)/2$, $(t_1-t_0)/2$), C($(t_2+t_0)/2$, $(t_2-t_0)/2$) and at last D($(t_2-t_0)/2$, $(t_2+t_0)/2$), where t_i with $i=0,1,2$ are respectively the intercepts of the lines S_2 , LP_n and LP_n' on the θ_2 axis, and their values are given as, $t_0=\pi-\theta_{DBi}$, $t_1=2\theta_{ci}+\sqrt{2}d_{max}$ and $t_2=2\theta_{ci}-\sqrt{2}d_{max}$, where θ_{DBi} is the θ_D boundary angle of the obstacle O_i.

5.2 Merging Infeasible Regions for a Global View on the Obstacles

In order to develop an approximate but safe global understanding of the infeasible regions for path planning purposes, given there are m workspace obstacles reachable by the arm, the m infeasible region rectangles are merged together and the resulting polygons (r_0 of them) are called poly(n) n=l,..,r_0. As a result of merging all the rectangular approximations of the infeasible regions, a convex or a concave polygon or a union of disjoint such polygons results. Collectively all the polygons, poly(n), n=1,....,r_0, will be known as POLY.

The 'global view'' of the joint space can be developed through sets of POLY vertices. Consider the infeasible region rectangles as shown in Figure 6. The complete set of POLY vertices in $(\theta_T\,,\,\theta_D)$ coordinates, in any of the configurations region (RC or LC), is given by the ordered set $\{V_1,V_2,.....,V_{14}\}$.

The global planner (by the help from the function Make-Traj-Vertex(.,.) to be introduced later) uses the ordered subset of POLY vertices, $POLYX=\{V_2,V_3,V_6,V_7=V_8,V_9,V_{12},V_{13}\}$ to plan a collision free path. Notice that a straight-line path that connects any two consecutive vertices of POLY or POLYX is collision-free. In some circumstances, the path planning which uses POLYX

vertices is advantageous over path planning which uses POLY vertices, because some safe but extraneous arm motions are avoided.

Note that any two POLY vertices with the same label in Figure 6 correspond to the same point in the workspace (according to corollary 2), and in fact the two vertices have the same (θ_T,θ_D) coordinates. The function Change-Configuration which will be described later, will generate a joint space trajectory which will make a change in the configuration of the arm, from RC to LC or vice versa, using the function Make-Traverse(.), as to be seen later.

5.3 Definition of a Free Space Corridor (FSC)

Given m objects in the reachable workspace of the SCARA arm, assume that the objects are ordered by the sequence of $\theta_{c1}<\theta_{c2}<.....<\theta_{cm}$, where $\theta_{ci}=\text{atan2}(y_{ci},x_{ci})$. Then if a point $P(q_1,q_2)$ is such that $2\theta_{ci-1} \le (\theta_T=q_1+q_2) < 2\theta_{ci}$ for $i=2,...,m$, then the point P is said to be in the free space corridor FSC_i, otherwise if $\theta_T < 2\theta_{c1}$ then P is said to be in the free space corridor FSC_1, and if $\theta_T \ge 2\theta_{cm}$ then P is said to be in the free space corridor FSC_{m+1} (see Figure 7). Notice that two points which are in the same FSC_i may not necessarily be connected. Further notice that a point P which is in FSC_i $i=1,...,m+1$ is not necessarily in a collision free region.

6. Path Planning Procedure

In order to move the robot end-effector from the point $S(\theta_{TS},\theta_{DS})$ in $\theta_T\theta_D$ space to the point $G(\theta_{TG},\theta_{DG})$, in the presence of m workspace obstacles, we want to determine a sequence of joint angles which comprise a path as, $q(\hat{k}) \in \{s_1\}=\{q / g_i(q) > 0 \text{ for } i=1,..,m\}$ where $\hat{k} \in Z^+$ (where Z^+ is the set of positive integers) is a parameter used to parameterize the path traced from S $(\hat{k}=0)$ to G $(\hat{k}=k_f)$. Here $\{s_1\}$ is the space of joint angles outside the infeasible regions IR_i for $i=1,...,m$. The path planning problem then can be posed as a constrained optimization problem, find the sequence $q(\hat{k})$ such that,

$$\min_{q(\hat{k}) \in \{s_1\}} \quad F(\theta_T(\hat{k}),\theta_{TG},\theta_D(\hat{k}),\theta_{DG})$$

is obtained at every step $\hat{k}$ of the minimization, where $F(.,.)$ is some function used to denote the measure of distance to the goal point and also to keep the manipulator at a safe distance from the obstacle boundaries. This constrained

optimization problem of finding a feasible collision free path in the set $\{s_1\}$, from $q(\hat{k}=0)=q_S$ to $q(\hat{k}=k_f)=q_G$, can be solved by the barrier method [20]. By applying this technique, once the optimization process is initiated at q_S, in the course of the optimization, the path proceeds toward q_G, and eventually reaches the joint space goal configuration, at step $\hat{k}=k_f$, if a solution path has successfully been found (i.e. no trapping in a local minimum has occured). We should note that most optimization schemes get stuck in local minimums. This is because numerical search schemes employing the gradient or descent search methods do not have a global view of the function being optimized. In order to prevent the path planning process from being stuck in a local minimum both the local and global view of the obstacle boundaries will be employed. Further a number of intermediate target goal points will be found and will be used in the optimization to reduce possibility of getting stuck in local minimas.

The path planning developed here will be a hybrid strategy based on two separate schemes, the global path planner is called every time the current point on the path, X, is outside POLY, and a local path planner is used every time X is inside POLY. The global path planner uses the information on the infeasible regions found from POLY, while the local path planner will use detailed local information of the infeasible regions given by the functions $g_i(q)$. The two schemes work together to move the arm through a number of intermediate target points to reach the desired goal point.

6.1 Solution to the Local Path Planning

The barrier method can be used to solve a constrained optimization problem such as that described in the previous section. The local path planning can be approximated by the following unconstrained minimization problem:

$$\underset{q}{\text{Minimize}}\ \phi(\mu(k),q) = f(q) + \frac{1}{\mu(k)} B(q)\ , \tag{17}$$

where the parameter $\mu(k)$ is a positive scalar and $q=q(k)$, with k as the iteration step in the optimization. As the parameter $\mu(k) \rightarrow \infty$, then the unconstrained optimization solution approaches that of the constrained optimization (see [20] for details on the barrier method). The joint trajectory is the sequence of joint angles $\{q(k)\}$ generated in the optimization steps. The cost function $f(q)$, is chosen as measure of distance to an intermediate target point X_T with coordinates $(\theta_{TXT},\theta_{DXT})$ and, $f(q)$ is given by,

$$f(q) = (\theta_D - \theta_{DXT})^2 + \omega\,(\theta_T - \theta_{TXT})^2\,, \qquad (18)$$

where ω is a constant positive weight. In the next section, we will explain how the intermediate target points are determined. Given the solution space $\{s_1\} = \{q / g_i^*(q) > 0\}$, the barrier function $B(q)$ is selected to avoid m obstacles in the environment as given by,

$$B(q) = \sum_{i=1}^{m} \frac{1}{g_i^*(q)}\,, \qquad (19)$$

where the constraint $g_i^*(q)$ is a continuous function which measures the distance of the current point on the path, with coordinates (q_1,q_2), from the boundary of the infeasible region IR_i,

$$g_i^*(q) = \begin{cases} |g_i(q_1,q_2)| & \text{if } q_1^{LBi} < q_1 < q_1^{UBi} \\ \left|\sqrt{g_i^2(q_1^{LBi},q_2) + (q_1^{LBi} - q_1)^2}\right| & \text{if } q_1 \le q_1^{LBi} \\ \left|\sqrt{g_i^2(q_1^{UBi},q_2) + (q_1 - q_1^{UBi})^2}\right| & \text{if } q_1 \ge q_1^{UBi} \end{cases} \qquad (20)$$

Here $q_1^{UBi} = \max_{q_1}\{q_1 / g_i(q)=0\}$ and $q_1^{LBi} = \min_{q_1}\{q_1 / g_i(q)=0\}$ and q_1^{LBi} and q_1^{UBi} are the lower and upper bounds of the projection of IR_i on the q_1-axis. Note that from section 3, $g_i^*(q_1,q_2)$ represents the distance along q_2 axis to the boundary of IR_i if $q_1^{LBi} \le q_1 \le q_1^{UBi}$, notice further that if $q_1 > q_1^{UBi}$ or $q_1 < q_1^{LBi}$ then g_i^* still represents a continuous measure of distance to the obstacle, this is accomplished by the second term of g_i^*. The distance measures g_i^* are not the minimum distance to the obstacle, and it is sufficient for g_i^* to be continuous for the success of the optimization. Note that the choice of ω and initial values of μ determine the direction in which the trajectory evolves during the optimization, as μ influences the repulsion from the infeasible regions boundaries and ω influences the attraction to the target point X_T.

6.2 The Path Planner

In order to explain the overall trajectory planning strategy the (θ_T,θ_D) coordinates of point X will be denoted by $(\theta_{TX},\theta_{DX})$, and the coordinates of G is $(\theta_{TG},\theta_{DG})$. The global view of the joint space obstacles used by the function **Make-Traj-Vertex(.,.)** of the Global-Plan is represented by POLYX (described in section 5.2). Let the variable p be the number of elements of POLYX, now if no θ_T side of the rectangles IR_i overlap, then we have **p=2m**, otherwise **p<2m**,

where m is the number of the infeasible regions. Now let the jth element of POLY be $V_j=(\theta_{TV}(j),\theta_{DV}(j))$ with $j \in J=\{1,2,...,2p\}$. The trajectory planner explained below, makes use of several functions, these are now described. **Make-Traj-Vertex(X,X_T)** generates a sequence of straight lines in joint space from X to X_T, (where X_T is on the boundary of POLY), passing through the nearest vertices of POLY without changing the arm configuration. **Make-Traj-LocOptim**(X,X_T) generates a local joint trajectory from X to X_T using the local optimization method as described in the previous section.

The function **Find-Vrtx-Indx(X)** returns an index $\hat{i}$ such that, $\hat{i}=2$ if $(\theta_{TX}-\theta_{TV}(2))<0$, or $\hat{i}=2p-1$ if $(\theta_{TX}-\theta_{TV}(2p-1))\geq 0$, otherwise $\hat{i} = \underset{j \in J}{\arg}((\theta_{TV}(j+1)-\theta_{TX})>0)$ and $(\theta_{TV}(j)-\theta_{TX})\leq 0))$. Whenever an intermediate target point is specified on the boundary of POLY, with a θ_T coordinate in the range of $\theta_{TV}(j) < \theta_T < \theta_{TV}(j+1)$, $_{j, j+1 \in J}$ then note that this point will be located on an edge of POLY, at the θ_D angle $\theta_{DV}(j)$, and this angle is equal to the θ_D boundary angle θ_{DBi}, of obstacle $i \in \{1,...,m\}$.

The function **config(X)** returns the numerical value assigned to the configuration of the arm for the point X, this is either +1 (for LC) or -1 (for RC). Suppose we are asked to move from X to X_T and the two points do not share the same configuration of the arm i.e. $config(X_T)=-config(X)$. Then a change in the configuration of the arm becomes necessary, and the function **Change-Configuration** plans such a trajectory, the function will be described in the pseudo-code presented later.

As the infeasible regions IR_i of the obstacles are aligned along the θ_T axis or the $\theta_D=\pi$ line, the path planner exploits this property by planning the joint motion from S to G approximately along the θ_T axis to a sequence of intermediate target points X_T until $\theta_{TXT}=\theta_{TG}$. Next the local path planner is employed if $G \in$ POLY to reach G by motions approximately along θ_D axis. If however G is outside POLY then the global planner determines a path by moving along the θ_D axis to the goal point G. In a case where the current point on the path, X is located either on the boundary or in the interior of POLY, and X and G are in the same FSC, then if G is in POLY but X and G differ by configuration, two target points are set for the local planner, first to reach the $\theta_D=\pi$ line, the second to reach G. If at any point $X \in$ POLY and X and G are not

in the same FSC, the local planner generates motions approximately along θ_D axis to exit POLY, then the global planner continue the path outside POLY to an intermediate target point on the boundary of POLY, with the same FSC as the goal point G. Also note that in order to determine accurately whether point X is inside or outside POLY, i.e. $X \in$ POLY or $X \notin$ POLY, we have used the complete set of POLY vertices $\{V_1, V_2, \ldots\ldots, V_{14}\}$ (Figure 6), as it was first given in section 5.2.

Let $J_1 = \{1,2,\ldots,m+1\}$ denote the set of joint space corridors when there are m obstacles and no two workspace obstacles share the same central line oc_i in the workspace. Also we assume that the values assigned to X (the current point on the path), io (the total number of target points assigned by the Local-Plan), and cnt (the variable used to count the number of entry points to POLY. An entry point to POLY is a point located on POLY 's boundary, used by the Local-Plan as a starting point for generation of a path inside POLY) will be known to all of the functions of the trajectory planner.

The local trajectory plan is now described in the below pseudocode, note that X_{T1} and X_{T2} are intermediate target points.

```
Local-Plan          /* plans trajectories inside POLY, using optimization. */
   if1 ( (X and G) ∈ FSC(j) ) then
                   j ∈ J1
    { i = 1         /* i keeps track of which target is next to be reached. */
      if2 ( config(X) = config(G) ) then
        { io = 1          /* io is the total number of targets. */
          XT1 = G }
      else2
        { io = 2
          XT1 = (θTG,π)        /* first a configuration change by going to the */
                               /* θD=π line. */
          XT2 = G )
        endif2 }
   else1
    { i = Find-Vrtx-Indx(X)
      io = 1
      XT1 = (θTX,θDV(i))   P intermediate target point on POLY 's boundary. */
```

```
      Make-Traj-LocOpt(X,X_T1) }          /* make a path to X_T1. */
    endif1
    for L1  i=1,io,1
     { Make-Traj-LocOpt(X,X_Ti)           /* make a path to the ith target.'/
      if3 ( θ_DX≠θ_DXTi ) then            /* has the ith target been reached? */
        { if4 ( X ∈ POLY ) then           /* if no, and still inside POLY, then the */
            { print 'Stuck'               /* optimization process is stuck in a */
                                          /* local minimum. */
              STOP }                      /* the gains ω and μ need adjustment, */
                                          /* to avoid local minimum. */
          else4
            go to L2
          endif4 }
      endif3 }
L1  continue
L2  return
```

The global trajectory planner is described in the below pseudo-code, given X, the current arm position, is outside POLY and X and G correspond to the same arm configuration.

```
Global-Plan  /* X ∉ POLY and the arm configuration for X and G are the same */
    î = Find-Vrtx-Indx(X)
    ĵ = Find-Vrtx-Indx(G)
    if1  ( î ≠ ĵ ) then
        { Make-Traj-Vertex(X,X_V(ĵ))    P X_V(ĵ) is the ĵ-th vertex of POLY. */
          X = X_V(ĵ) }
    endifl
    if 2  (G ∉ POLY)  then
          Make-Traj-Line(X,G)
    else2
          { X_T3 = (θ_TG,θ_DV(ĵ))       /* X_T3 is on the boundary of POLY, and is */
                                        /* chosen in the same configuration region as X. */
          Make-Traj-Line(X,X_T3)
```

```
            X = X_T3 }
        endif2
        return
```

The overall trajectory planning is performed by Path-Planner, it achieves this by calling Global-Plan and Local-Plan and Change-Configuration functions:

```
Path-Planner        /* Overall path planner. */
    cnt = 0         /* cnt counts the number of entry points to POLY. */
    while1  ( X ≠ G ) then
      (if2  ( X ∈ POLY)  then
          Local-Plan
       else2                /* X ∉ POLY */
        { if (G ∉ POLY)   Change-Configuration
         Global-Plan
         if3 (G ∉ POLY) then
            if ( X ≠ G ) print 'error'     /* here G must be reachable from */
                                           /* X, unless X and G are not */
                                           /* feasibly connected */
          else3  /* here G ∈ POLY */
               { if4  (cnt = 0 ) then
                   cnt = cnt+1             /* the first entry point is X_T3, */
                                           /* as assigned by Global-Plan. */
                 else if4 ( cnt=1 ) then
                   { Change-Configuration  /* if path not completed in one */
                                /* configuration, then change configuration, */
                                /* and enter POLY from another direction. */
                                /* The next entry point as assigned by
                                /* Change-Configuration is either X_TC or X_T3. */
                     cnt = cnt+1 }
                 endif4 }
          endif3 }
        endif2 }
    endwhile1
    print  ' A path is complete. '
```

stop

Note that Path-Planner calls the Change-Configuration function only when $X \notin$ POLY. Let $d(.,.)$ represent the distance between any point $X_1(\theta_{TX1},\theta_{DX1})$ and $X_2(\theta_{TX2},\theta_{DX2})$, i.e., $d(X_1,X_2) = \sqrt{(\theta_{TX1}-\theta_{TX2})^2+(\theta_{DX1}-\theta_{DX2})^2}$. Suppose the arm is required to go from the point X to the point X_T, and that $\text{config}(X) = \text{config}(X_T)$. In this case, the function Change-Configuration first calculates the lengths of two paths with distances d_1 and d_2, and next selects the path with the shorter length in order to produce a change in the arm configuration.

Given $\hat{i}$ = Find-Vrtx-Indx(X) and $\hat{j}$ = Find-Vrtx-Indx(X_T) and $V_{(i)}=(\theta_{TV}(i),\theta_{DV}(i))$ is the ith vertex of POLY, in RC or LC arm configuration, then let

(1) l_1 to be the closest index to the point X, on the right hand side of X (increasing θ_T), at which $\theta_{DV}(l_1+1)=\pi$, and

(2) l_2 to be the closest index to the point X, on the left hand side of X (decreasing θ_T), at which $\theta_{DV}(l_2-1)=\pi$, now the distances d_1 and d_2 are calculated as follows,

$$d_1 = \begin{cases} d(V_{(\hat{i}+1)},X)+\sum_{i=\hat{i}}^{l_1-1}[d(V_{(i+1)},V_{(i)}] + s_1+t_{11}+d(V_{(\hat{j}+1)},X_T) & \text{if } l_1 \geq \hat{j}+1 \\ d(V_{(\hat{i}+1)},X)+\sum_{i=\hat{i}}^{i=i+1}[d(V_{(i+1)},V_{(i)}] + s_1+t_{12}+d(V_{(\hat{j})},X_T) & \text{if } l_1 \leq \hat{j} \end{cases}$$

$$d_2 = \begin{cases} d(V_{(\hat{i})},X)+\sum_{i=l_2+1}^{\hat{i}}[d(V_{(i)},V_{(i-1)}] + s_2+t_{21}+d(V_{(\hat{j})},X_T) & \text{if } l_2 \leq j \\ d(V_{(\hat{i})},X)+\sum_{i=\hat{i}}^{l_2+1}[d(V_{(i)},V_{(i-1)}] + s_2+t_{22}+d(V_{(\hat{j}+1)},X_T) & \text{if } l_2 \geq \hat{j}+1 \end{cases}$$

where the expressions for s_1, s_2, t_{1k} and t_{2k} for $k=1,2$ are:

$$t_{11}=\sum_{i=l_1}^{(\hat{j}+2)}[d(V_{(i)},V_{(i-1)})], \qquad t_{12}=\sum_{i=l_1}^{(\hat{j}-1)}[d(V_{(i+1)},V_{(i)})], \qquad t_{21}=\sum_{i=l_2}^{(\hat{j}-1)}[d(V_{(i+1)},V_{(i)})],$$

$$t_{22}=\sum_{i=l_2}^{(\hat{j}+2)}[d(V_{(i)},V_{(i-1)})], \qquad s_1=2(\pi-\theta_{DV}(l_1)), \qquad s_2=2(\pi-\theta_{DV}(l_2)).$$

```
Change-Configuration     /*  X ∉ POLY  */
     if1  ( ( G ∉ POLY ) and ( config(X) = config(G) ) )  then
          go to L1
     else1
          { j̄ = Find-Vrtx-Indx(G)
            X_T = (θ_TG, θ_DV(ĵ))   /* note that X_T is chosen in the arm configuration */
                                    /* region which would make config(X_T) = - config(X). */
            calculate l_1 and l_2
            calculate d_1 and d_2
            if2  (d_1<d_2)  then
                 Make-Traverse(X,X_T,î+1,l_1)
            else2
                 Make-Traverse(X,X_T,î,l_2)
            endif2 }
     endif1
L1   return
```

Given the configuration of the arm at point X is different than the configuration required by the arm at point X_T, then the function Make-Traverse(.) first generates a path from X to an intermediate point located on the boundary of RC and LC regions, using Make-Traj-Vertex(.,.) and Make-Traj-Line(.). This path makes a change of configuration for the arm from **config(X)** to **config(X_T)**, before X_T can be reached. If the path gets close to G (the path has entered the FSC in which G is located) and G ∈ POLY, then a new intermediate target point, X_{TC}, is next created on the boundary of POLY. If X_{TC} is created, then Make-Traverse(.) generates a path to X_{TC} and stops at X_{TC}. Otherwise Make-Traverse(.) stops at the above boundary point of RC and LC regions.

```
Make-Traverse(X,X_T,i,l)          /* config(X_T) = - config(X) */
     Make-Traj-Line(X,V_(i))
     Make-Traj-Vertex(V_(i),V_(l))
     Make-Traj-Line(V_(l),(θ_TV(l),π))
     if1  ( (both V_(l) and G) ∈ FSC_j  and (G ∈ POLY) )  then
                                j ∈ J_1
```

```
        { /* create a new intermediate target point, X_TC = (θ_TC,θ_DC). */
      θ_TC = θ_TV(l)
      θ_DC = θ_DG
      X_TC = (θ_TC,θ_DC)      /* note that X_TC is chosen in the arm configuration */
                              /* region which would make config(X_TC) = config(X_T). */
      Make-Traj-Line( (θ_TV(l),π) , X_TC)   /* take the path to X_TC. */
      X = X_TC                              /* the current point on the path is X_TC. */
      go to L1 }
    endifl
    X = (θ_TV(l),π)           /* the current point on the path is X=(θ_TV(l),π) with */
                              /* config(X)=config(X_T). */
    Global-Plan
L1  return
```

Lemma2:

If (S and G) e IR_i for i=l,..,m, and J_1 denotes the set of the free space corridors, and if there exists a connected path $\{q(\hat{k})$ with $\hat{k} \in Z^+\} \in \{s_1\}$, from S to G, then the above path planner will find a path if we utilize appropriate values of gain ω and μ every time the local path planner is called.

Proof: Suppose that there exists a connected path from S to G, and the portion of the path devised by the local planner does not enter the infeasible regions i.e. $\{q(\hat{k})\} \in \{s_1\} \notin IR_i$ $i \in \{1,..,m)$, this is achieved with appropriate selection of the gain μ. The path planner uses the Local-Plan, the Global-Plan and possibly Change-Configuration to find a path from S to G. If a configuration change to reach the goal point is needed, it is accomplished using the Change-Configuration function, followed by a sequence of local and global plans, in order to reach the goal point.

Given (S and G) e POLY, the Global-Plan can find a trajectory from S to G, if S and G are feasibly connected in the joint space. If S $\in$ POLY and G e POLY, then as long as we choose gains ω and μ, such that from S to an intermediate target point X_T e POLY, a trajectory can be generated by the Local-Plan, then X_T to G trajectory can be determined by the Global-Plan. Similarly if G $\in$ POLY and S e POLY, for an appropriate ω and μ, we can move from the boundary of POLY at point X_T to G, having got to X_T from S by using the global plans. If both conditions (both Sand G) $\in$ POLY and (both S and G) $\in$ FSC_j are satisfied for any $j \in J_1$, we are required to find a

sequence of ω' s or one ω and an initial gain μ which will generate a trajectory from S to G. If (both S and G) $\in$ POLY but are in different FSC, then the path devised by the Local-Plan first exits POLY, followed by the Global-Plan taking the path closer to the goal point to an intermediate target point X_T at which the local planner is called to enter POLY, with X_T as the entry point to POLY. Therefore the success of the planner depends on the choice of ω sequence and the initial value of the sequence of μ, used in the Local-Plan. Thus we conclude that if the point S and G are both in $\{s_1\}$ and they are connected, given a sequence of gains w and an initial value of μ, the above path planner will generate a trajectory from S to G. AAC

We should note here that the trajectory generated by the above planner is one of many, the trajectory is not necessarily the shortest or necessarily the smoothest. The generated trajectory can be modified to satisfy such additional considerations. As we noted earlier the entire trajectory can be generated by performing an optimization. However under such cases the chances of getting stuck in a local minimum is higher than that is possible using the strategy devised here. The strategy devised here relies on the observation that the IR_i can be approximated into rectangles which line up on the $\theta_D=\pi$ line, therefore trajectory plan can consist of two decoupled movements, one along the θ_T axis outside POLY and one approximately along θ_D axis to exit and enter POLY. Therefore a number of intermediate targets which consist of motions along θ_T axis and along θ_D axis are generated to reach the goal point. This reduces the burden on picking appropriate values for the gains w and μ, by determining a sequence of intermediate target points.

7 Simulation Results

Notice that in cases where the free regions of the joint space are disjointed, no path connecting two points each located in a separate disjointed region is feasible. Therefore we have assumed here that $\max(l_1, |l_2-l_1|) < ||(x_{ci},y_{ci})|| - r_i \le l_1+l_2$. In all the three examples below, the link lengths are selected as, $l_1=4$ and $l_2=3$. The two obstacles in the examples below have radius $r_1=r_2=1$, their centers are at $(x,y)=(2,5.3)$ for O_1, and at $(x,y)=(-1,5)$ for O_2.

Example1:

In this example, S is at $q_S=(0.5,2.3)$ radians, and the goal G is at $q_G=(1.4,2)$ radians. Figure 8 shows the infeasible regions due to the two obstacles. The generated path is also shown in Figure 8. Planning started with a call to the local planner, as S ϵ POLY. This generated a trajectory to point A, located outside POLY. The global planner was next activated which generated a trajectory to **B** with coordinates $(\theta_{TG},\theta_{DV}(\hat{j}))$ with $\hat{j}$=Find-Vrtx-Indx(G). At point **B,** the local planner is called, which generated a trajectory to G. The weights chosen for the operation of the local planner were $\omega=0.5$ and initial $\mu=100$, for both calls to this planner. The manipulator motions in the workspace amongst the obstacles are also shown in Figure 8.

Example 2:

The location and size of the workspace obstacles as well as S and G configurations are the same as in example 1. In this example, however, the initial weight μ was increased to $\mu=500$ while $\omega=0.5$ was chosen the same as in example 1. Figure 9 shows the collision-free path, it was devised entirely by the local path planner from S to G. Notice the path never leaves POLY, due to the fact that the repelling force $(1/\mu)$ **B** from the infeasible regions boundaries is smaller here than that in example 1, and thus the local planner devises a path very close to the boundary of the infeasible regions. The pull toward the goal point as produced by the attractive force **f(q)** makes the path go toward the goal point, as it goes over the top of IR_1. This example clearly shows that the trajectory path is dependent on the choice of ω and μ. Further the number of times the global planner is called is as a result dependent on these parameters. The robot end-effector motions in the workspace is also shown in Figure 9. We see the joint motions are smoother because there is no joint space discontinuities due to **global/local** trajectory plan change as in example 1.

Example 3:

Both the start and the goal points were chosen outside POLY. Figure 10 shows the collision-free path which was planned in between these two points. This path has been planned entirely by the global planner, by effectively using the vertices of POLY. Here we are not required to select any gains μ and ω as the trajectory lies entirely outside POLY. The robots motions in the workspace

is also shown in Figure 10. Notice that in order for the arm to pass by obstacle O_1 with no collision, initially the first link rotates backward as the second link bends further on link one, once the boundary angle θ_{DB1} has been reached for the interior angle in between the links, then the arm moves forward, and after the arm passes both obstacles safely, then the arm stretches out in order to reach to the goal joint angles.

8 Summary and Conclusions

In this paper we developed a path planning strategy for SCARA robots which makes use of exact and approximate but safe knowledge of the infeasible joint space regions due to the obstacles in the workspace. The exact knowledge of the infeasible regions are used when the arm is close to the obstacles or is surrounded by them in which case it utilizes constrained minimization techniques to generate a path to a sequence of intermediate target goals to reach a final goal. The intermediate target goals are generated from the global approximate (but safe) view of the infeasible regions,, When the arm is far from obstacles the approximate but safe knowledge of the infeasible regions is used to generate a trajectory which drives the arm to the goal point or a point close to it. The work here is different from the past work in the sense that it makes use of optimization techniques as well as global information on the approximate obstacle boundaries. The iterative calls to the 'global and local path planning procedures with a continuous change of target points allows us to reduce the chance that a purely optimization technique would become stuck in some local minima, and a purely global plan would be unable to find a path.

The trajectories generated by the planner described here can be modified to minimize the distance travelled or to improve the smoothness of the trajectory along the path. This can be done by rounding the corners of the generated path at the locations where the path has passed through the POLY vertices or when there is a change over between **local/global** plan. Even the trajectories generated by the Local-Plan may be smoothed by including appropriate constraints in the set of constraints of the optimization problem.

The ideas developed in this paper can be further expanded to obstacles of any shape, although determining the joint space infeasible regions would become more computationally intensive as more than two functions would be required to describe these regions. We would also have to develop

differentiable functions to smooth out the discontinuities of the infeasible regions boundary. Note however that only when the robot is required to initially start at a point inside one of the smallest enveloping circles which bound the non-circular obstacles, or when the robot is required to finish at a goal point inside these enveloping circles, it would become necessary to take the actual shapes of the obstacles into consideration. However it would still be feasible to use the path planning strategies developed here, but the path planning would have to be subdivided into those which has to take place in the interior of a circle and into those that are outside the enveloping circles. We should also note that the physical dimensions of the robot links may be incorporated into the infeasible region obstacles, by growing the obstacles appropriately and then assuming the robots can be represented by line segments.

Many robots that are used in industry and research have six or more degrees of freedom. The trajectory planning strategy developed here can be developed further to popularize its use in higher joint space and workspace dimensions. Such developments are indeed feasible, this would require us to develop obstacle infeasible regions in multi-dimensional space by numerical techniques. The search schemes and the global obstacle modelling described here would have to be extended into those higher dimensional spaces.

REFERENCES

[1] Lozano-Pbrez, T., "Spatial Planning: A Configuration Space Approach," IEEE Transactions on Computers, 32(2), pp. 108-120, 1983.

[2] Lozano-Pbrez, T., and Wesley, M.A., "An algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles," Communications of the ACM, Vol. 22, No. 10, pp. 560-570, October 1979.

[3] Brooks. R.A., "Solving the Find-Path Problem by Good Representation of Free Space," IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-13, No. 3, pp. 190-197, March/April 1983.

[4] Brooks, R.A., "Planning Collision-Free Motions for Pick-and Place Operations," The International Journal of Robotics Research, Vol. 2, No. 4, pp. 19-44, Winter 1983.

[5] Luh J.Y.S., and Campbell, C.E., "Minimum Distance Collision-Free Path Planning for Industrial Robots with a Prismatic Joint," IEEE Transactions on Automatic Control, Vol. AC-29, pp. 675-680, August 1984.

[6] Faverjon, B., "Obstacle Avoidance Using an Octree in the Configuration Space of a Manipulator," Proceedings IEEE International Conference on Robotics and Automation, Atlanta, GA, March 1984.

[7] Gouzenes, L., "Strategies for Solving Collision-Free Trajectories Problems for Mobile and Manipulator Robots," The International Journal of Robotics Research, Vol. 3, No. 4, pp. 51-65, Winter 1984.

[8] Donald, B.R., "Motion Planning with Six Degrees of Freedom," Technical Report AIM-791, MIT Artificial Intelligence Laboratory, 1984.

[9] Brooks, R.A., and Lozano-Pérez, T., "A Subdivision Algorithm in Configuration Space for Findpath with Rotation," IEEE Transactions on Systems, Man And Cybernetics, Vol. SMC-15, No. 2, pp. 224-233, March/April 1985.

[10] Khatib, O., "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots," International Journal of Robotics Research, Vol. 5, no. 1, pp. 90-98, Spring 1986.

[11] Erdmann, M., Lozano-Pdrez, T., "On Multiple Moving Objects," Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, April 1986, pp. 1419-1424.

[12] Kant, K., Zucker, S.W., "Toward Efficient Trajectory Planning: The Path-Velocity Decomposition," The International Journal of Robotics Research, Vol. 5, no. 3, pp. 72-89, Fall 1986.

[13] Lozano-Pdrez, T., "A simple Motion-Planning Algorithm for General Robot Manipulators," IEEE Journal of Robotics and Automation, Vol. RA-3, No. 3, pp. 224-237, June 1987.

[14] Lumelsky, V.J., "Dynamic Path Planning for a Planar Articulated Robot Arm Moving Amidst Unknown Obstacles," Automatica, Journal International Federation on Automatic Control (IFAC), Pergamon Press, September 1987.

[15] Freund, E., Hoyer, H., "Real-Time Pathfinding in Multi-Robot Systems Including Obstacle Avoidance," The International Journal of Robotic Research, Vol. 7, no. 1, pp. 42-70, Feb. 1988.

[16] Buckley, C.E., "A Foundation for the "Flexible-Trajectory" Approach to Numeric Path Planning," The International Journal of Robotics Research, Vol. 8, No. 3, pp. 44-64, June 1989.

[17] Chang C., Chung M.J., and Bien Z., 'Collision-Free Motion Planning for Two Articulated Robot Arms Using Minimum Distance Functions," Robotica, Vol. 8, pp. 137-144, 1990.

[18] Stonier, R.J. 'Use of Liapunov Techniques for Collision-Avoidance of Robot Arms," Control and Dynamic Systems, Vol. 35, .Academic Press, Inc., pp. 185-214, 1990.

[19] Erfan, Z. and Ahmad, S., "Lyapunov Based Collision Avoidance and Control for Multiple Revolute Manipulators," submitted to IEEE Transactions on Systems, Man and Cybernetics.

[20] Luenberger, D.G., 'Linear and Nonlinear Programming," ,Second Edition, Addison Wesley, Boston, MA 1989.

[21] Fiacco A.V. and McCormick G.P., "The sequential Unconstrained Minimization Technique for Non-linear Programming, A Primal-Dual Method," Management Science, Vol. 10, No.2, pp. 360-366, January 1964.

[22] Schwartz, J.T., and Sharir, M., "On the Piano Movers' Problem I, II, III" Technical Reports 39, 41, 52. Dept. Computer Science, Courant Institute of Mathematical Science, New York Univ., New York, NY, 1981-1983.

[23] Schwartz, J.T., and Sharir, M., "Efficient Motion Planning Algorithms in Environments of Bounded Local Complexity," Courant Institute, Robotics Lab., N.Y.U., New York, NY, June 1985.

[24] O'Dunlaing, C., Sharir, M., and Yap C.K., "Retraction: A New Approach to Motion-Planning," Proceedings 5th Annual ACM Symposium on Theory of Computing, 1983.

[25] Reif, J., and Sharir, M., "Motion Planning in the Presence of Moving Obstacles," Proceedings of Symposium on Foundation of Computer Science, Portland, Oregon, pp. 144-154, October 1985.

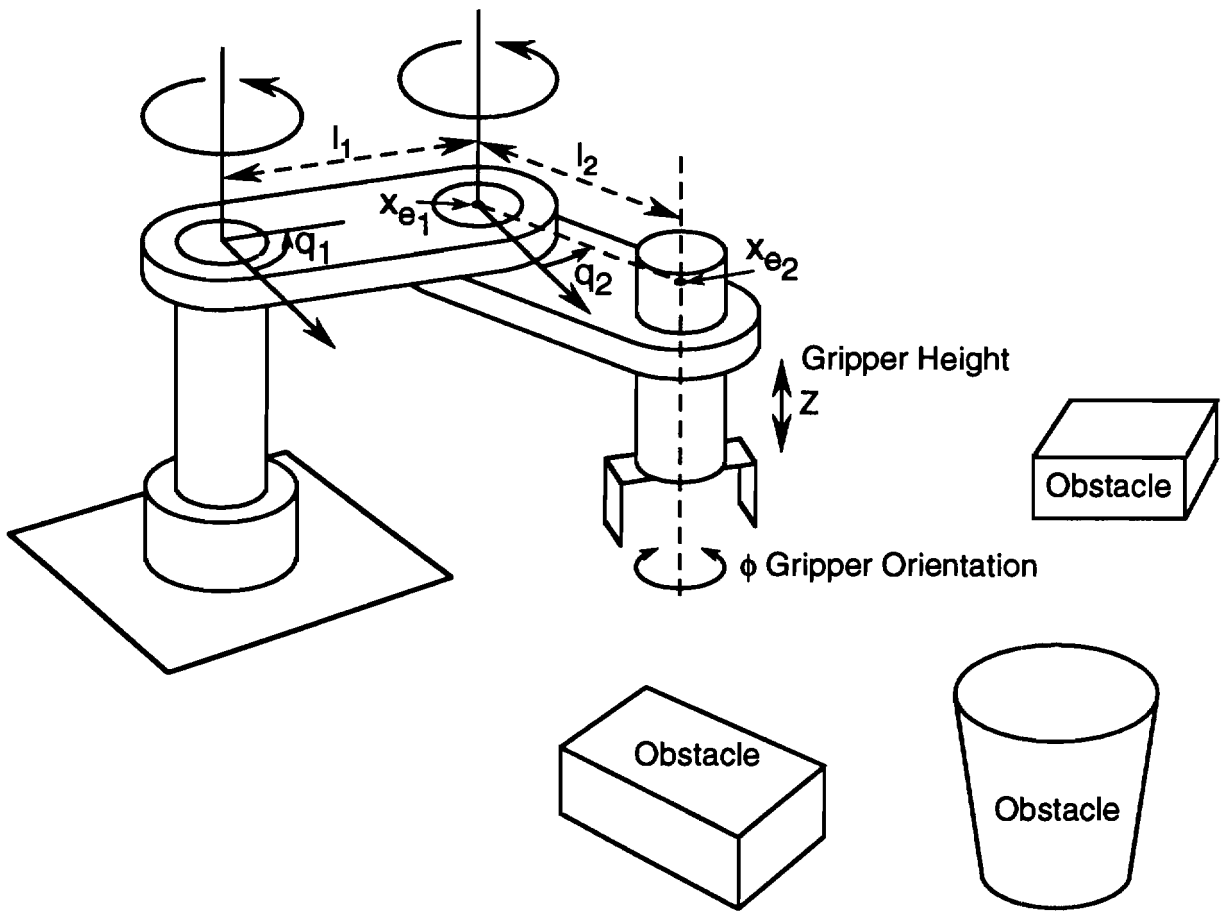


Figure 1. A SCARA Manipulator

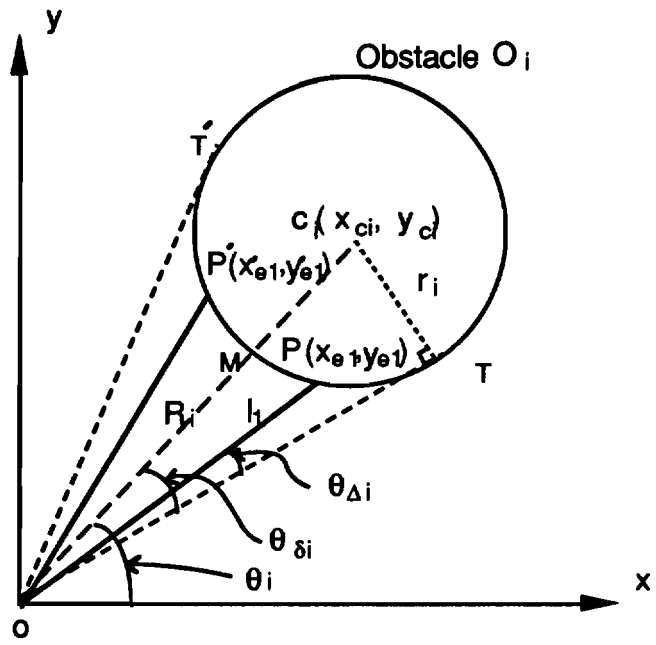


Figure 2 Range of collision for link one, the position of the end point of the first link on the arc MT and a symmetrical position of this end point on MT′.

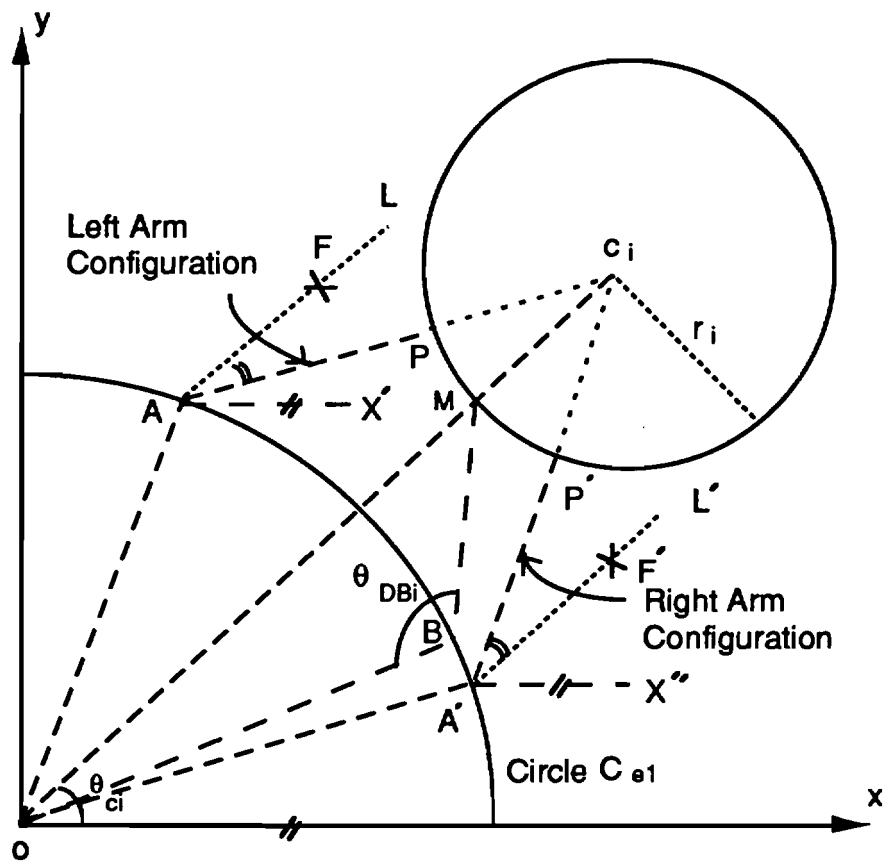


Figure 3 The symmetrical characteristic of the infeasible region about $(q_1,q_2) = (\theta_{ci},\theta_{ci})$ as evident in the workspace.

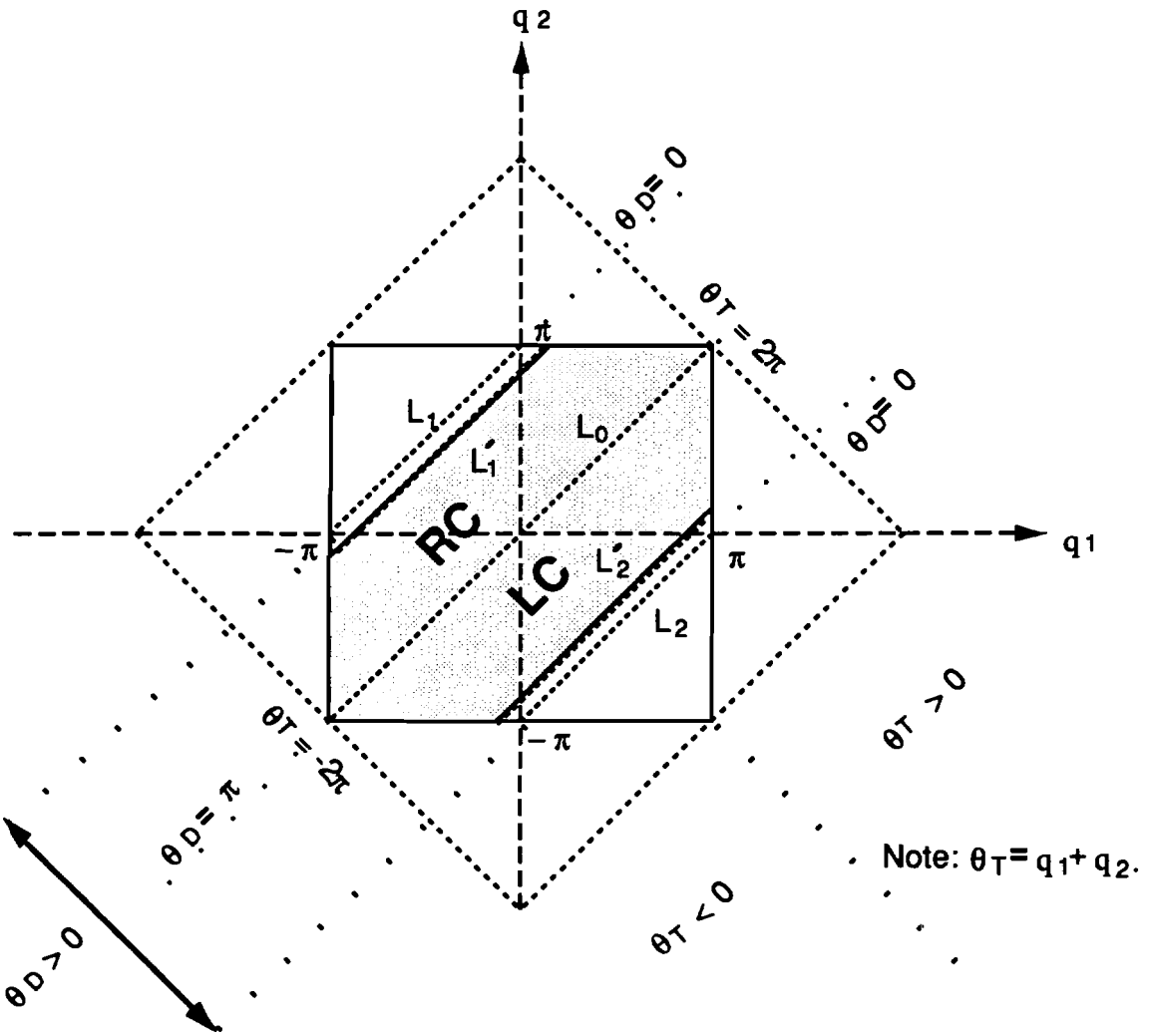


Figure 4 The joint space and the regions of this space corresponding to Right-hand Configurations (RC) and Left-hand Configurations (LC) of the manipulator.

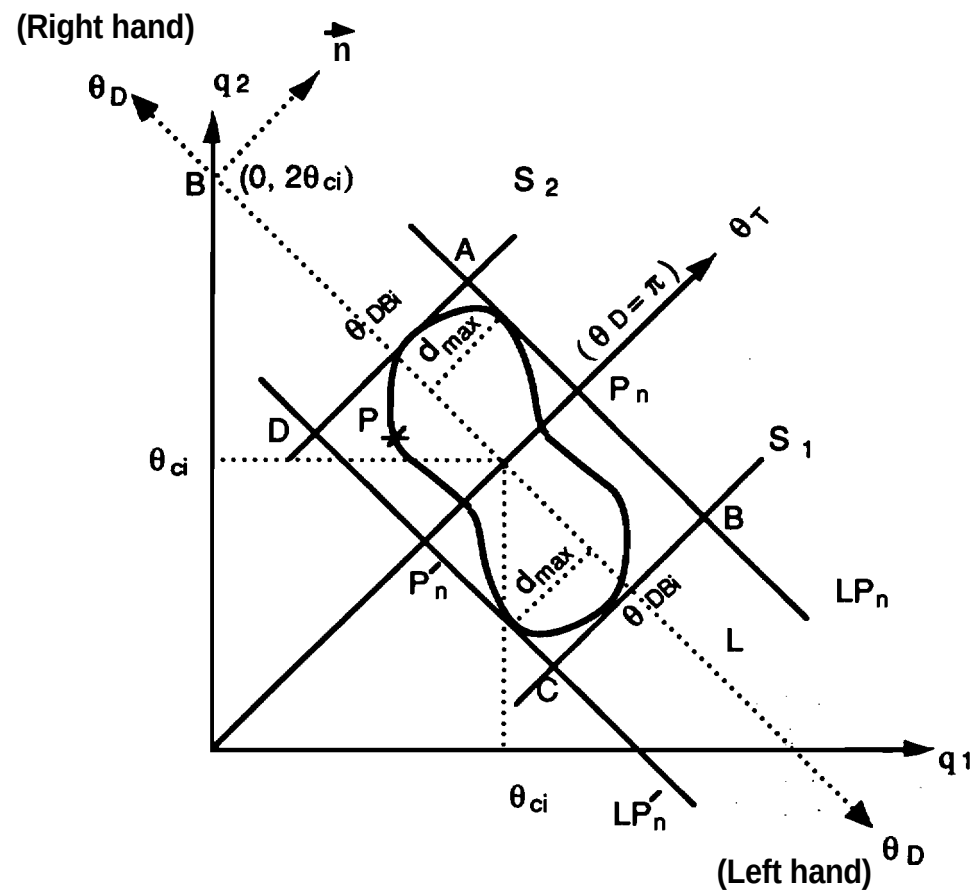


Figure 5 The rectangular approximation to the infeasible region IR_i.

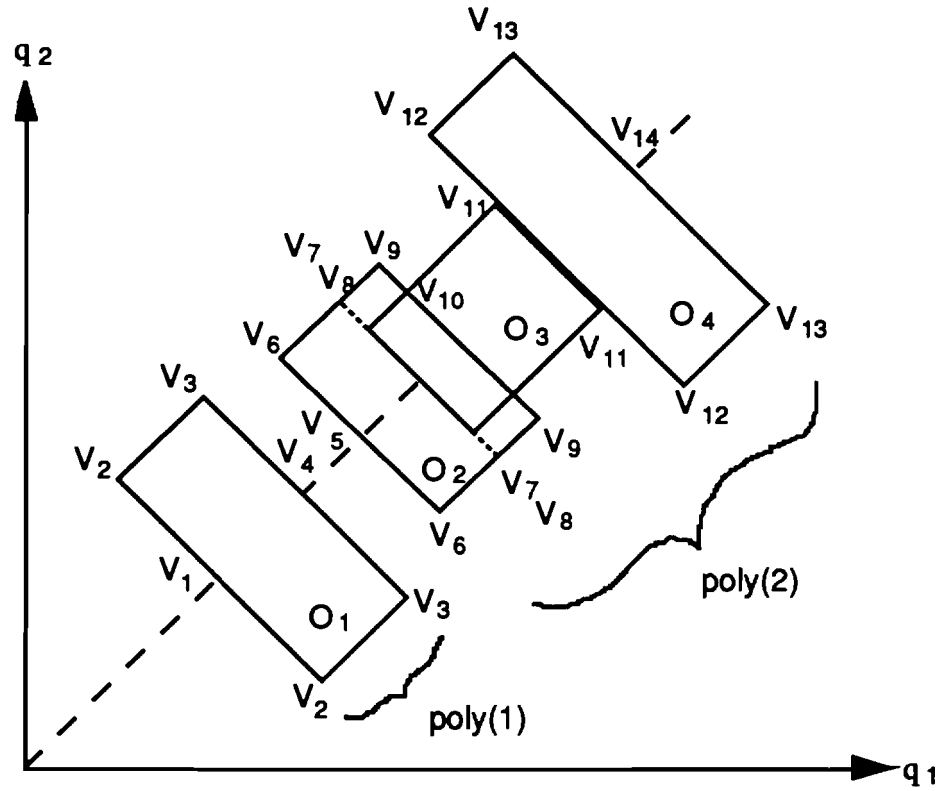


Figure 6 The global view of the obstacles in the joint space, poly(n) n=1,2.

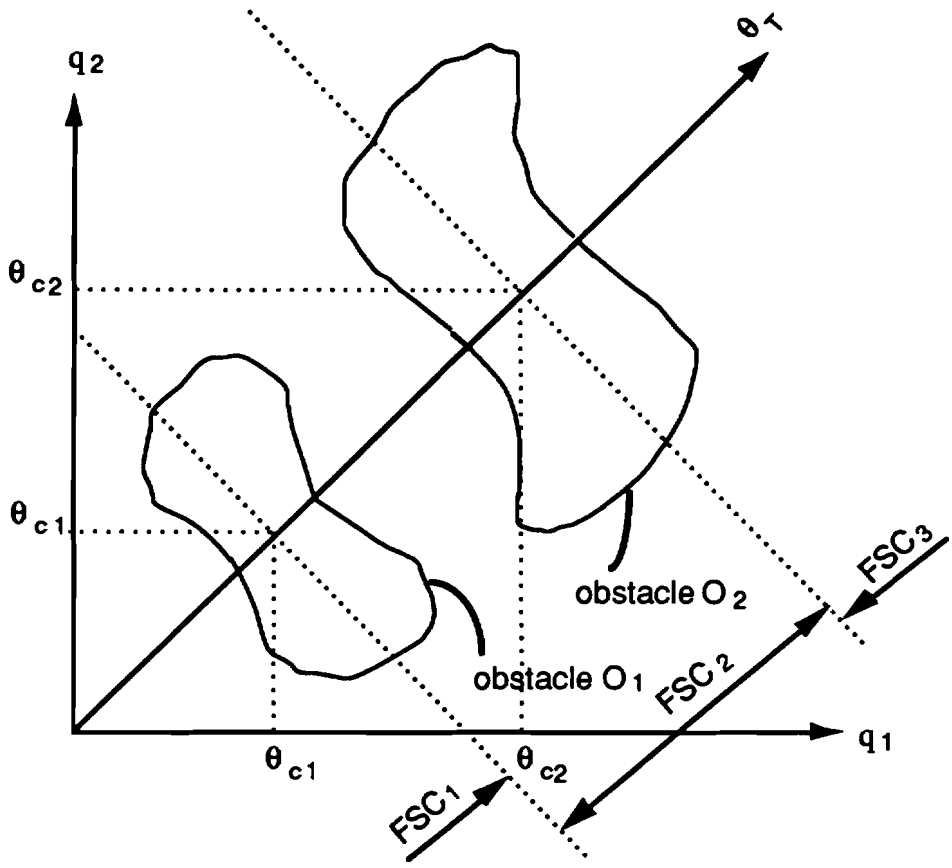
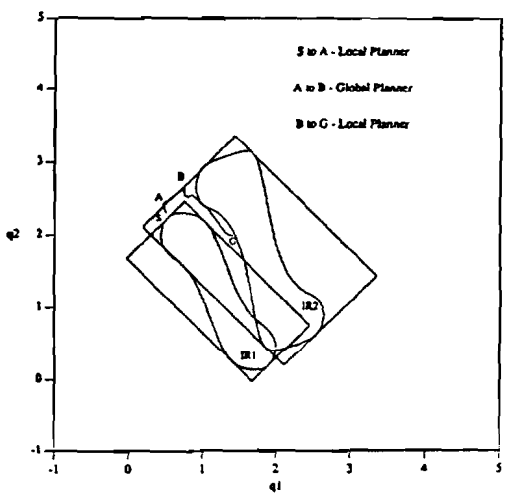
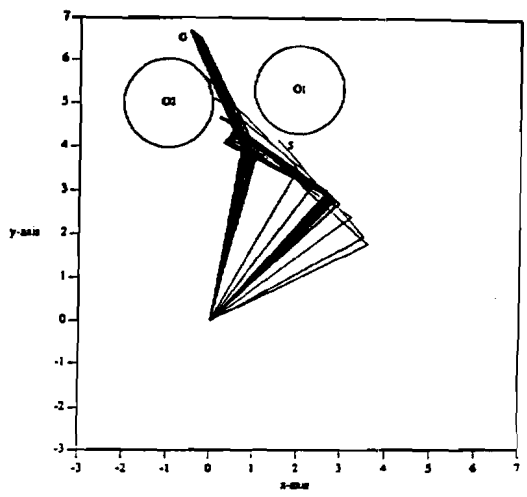


Figure 7 The free space corridors FSC_i, i=1,2,3.



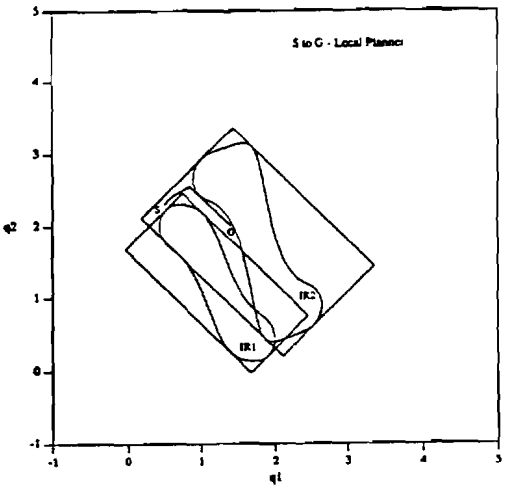
The joint space path.



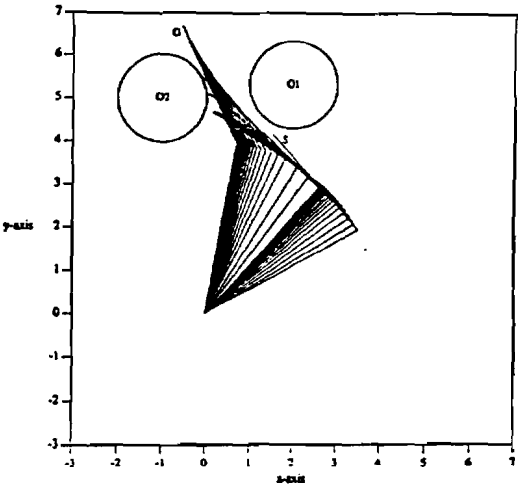
The workspace trajectory of the arm.

Figure 8 The joint space path and the workspace trajectory of the arm for example 1.

Figure 8, continued



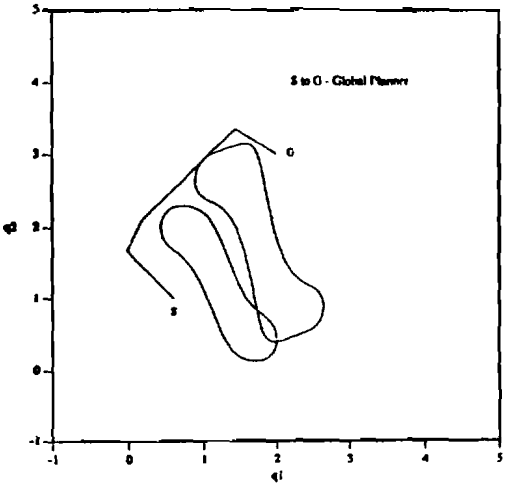
The joint space path.



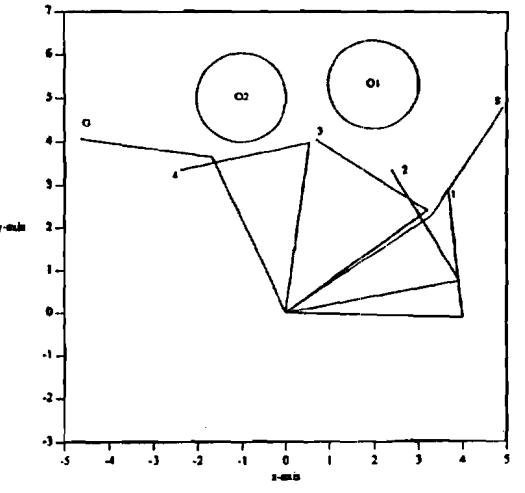
The trajectory of the arm.

Figure 9 The joint space path and the workspace trajectory of the arm for example 2.

Figure 9 continued



The joint space path.



The trajectory of the arm.

Figure 10 The joint space path and the workspace trajectory of the arm for example 3.

Figure 10 continued