Robotica (2002) volume 20, pp. 375–384. © 2002 Cambridge University Press
DOI: 10.1017/S0263574702004071 Printed in the United Kingdom

Reactive approach to on-line path planning for robot manipulators in dynamic environments*

Margarita Mediavilla, José Luis González, Juan Carlos Fraile and José Ramón Perán

Departamento de Ingeniería de Sistemas y Automática, E.T.S.I.I., Universidad de Valladolid, Valladolid (Spain)
E-mail: marga@eis.uva.es

(Received in Final Form: February 1, 2002)

SUMMARY
This paper describes a new approach to path planning of robot manipulators with many degrees of freedom. It is designed for on-line motion in dynamic and unpredictable environments. The robots react to moving obstacles using a local and reactive algorithm restricted to a subset of its configuration space. The lack of a long-term view of local algorithms (local minima problems) is solved using an off-line pre-planning stage that chooses the subset of the configuration space that minimises the probability of not finding collision free paths. The approach is implemented and tested on a system of three Scorbot-er IX five link robots.

KEYWORDS: Path planning; Dynamic environments; On-line motion; Scorbot-er IX robots

1. INTRODUCTION
It is well known that motion planning for robot manipulators with many degrees of freedom is a complex task. This is probably the reason why the research on this area has been mostly restricted to static environments, where the obstacles are still or at least follow known trajectories.[1–4]

Since most path planning is designed for static environments, robot manipulators (in industry as well as in many research projects) are restricted to pre-calculated trajectories and subject to rigid timings. These features make them too rigid and unable to adapt to changes. Increasing the reactivity of robot motion would have important advantages, If the robots had reactive path planning algorithms they could:

- react to unexpected events such as moving obstacles or faults,
- move without a previous model of their environment, based only on sensor information,
- be less dependent on pre-calculated trajectories, since path planning could be done on-line.

In the last four years, our research group has been working on a prototype of a multi-manipulator system[5–8] located in our laboratories of the University of Valladolid*. Our multi manipulator system is designed as an assembling cell that achieves a high production rate by removing the rigid timings and delays from the system. The three robots of our system share a good portion of their working space, but the path planning method that we are developing enables them to operate despite the fact that other robots might also be moving in the same area. The high production rate is achieved by making the robots depend only on the rate of arrival of the supplies. This means that, when a supply comes in, one of the robots starts its motion towards it, irrespective of whether the rest of the robots are at that time already in motion or not. This kind of behaviour needs the robots to have an on-line and reactive planning method.

Path planning approaches for robotic manipulators can broadly be categorized into the two classes of global and local methods: Global methods[9–12] are computationally very expensive, and computational cost increases rapidly as a function of the number of manipulator joints. Furthermore, they are not applicable when the obstacles are unmodeled or subject to uncertainties.

Local methods, on the other hand,[13–15] are reactive and can be used for real-time path planning, but most of them are very limited in their capabilities and easily get trapped in local minima. It is the problem of the local minima that has avoided potential field methods from becoming a valid reactive path planning framework for manipulators.

This paper presents a new approach to reactive on-line path planning for robot manipulators. It is designed for robots that move in a dynamic environment subject to uncertainties. By dynamic environment subject to uncertainties we understand a workplace with still and moving obstacles where the position of the obstacles at each instant of time is known, but where the entire trajectory of the obstacles cannot, because the position is either influenced by external perturbations, or conditioned by our own path planning.

There are not many approaches to on-line motion planning of robot manipulators in the literature. Li and Latombe[16] Ek describe an application of on-line path plannning of two Scara robots. They use a non-reactive method based on global planning. They obtain on-line path

* Part of this work was carried out while the first author was visiting Prof. Gupta at the robotics laboratory, Simon Fraser University, Canada.

* This work has been supported by the Comisión Interministerial de Ciencia y Tecnología. CICYT, TAP95/0092.

planning by reducing the problem to a set of subproblems with three degrees of freedom each. It does not seem possible to extend the use of these simplifications to other types of robots. Hamilton and Dodds[17] use a reactive method based on a set of behaviours. This approach has a problem similar to the local minima of potential fields. The elastic band method of Quinlan and Khatib,[18] combines local and global information by planning the motion of the entire path. Its main drawback is the fact that there must always exist a collision free path between the initial and final configurations of the robot. Seraji and Bon[19] propose an on-line path planning method based on the Cartesian space of the robots. It is based on a set of heuristic "strategies" that simplify the problem. Meng and Yang[20] use neural networks for on-line path planning of mobile robots and manipulators with few degrees of freedom. This approach seems very interesting, but the authors do not train the neural networks, therefore it is difficult to know if this approach is free from problems similar to the local minima of potential fields.

The work by Li and Latombe[16] and the one by Cao and Dodds,[21] are two of the few examples of path planning for teams of robotic manipulators that can be found in the literature. The planning problem we are facing is similar to the one solved by Li and Latombe and by Cao and Dodds. The work by Cao and Dodds obtains off-line path planning of two robots in pick and place tasks, but the path planning method they propose has long computing times (several hours). The approach of Li and Latombe obtains on-line path planning of two Scara robots. Our approach gets the same results as Li and Latombe's work, is on-line path planning for two robots in pick and place tasks, but our approach also has some other useful characteristics that Li and Latombe's does not have. Our method displays reactive behaviour, and we also applied it to three robots.

This paper shows the basic ideas and the first results of a new path planning method. These first results confirm the validity of the ideas proposed, but the full capabilities of the method are currently being explored by the authors. The paper is organized as follows, Section 2 discusses the basic ideas and presents a general description of the method, while Section 3 shows some results. Finally, conclusions are drawn in Section 4.

2. PATH PLANNING BASED ON MOTION STRATEGIES

On-line path planning requires very short computing times. The global methods that explore all the configuration space of the robot, are too slow for this kind of motion. Some sort of simplification must be carried out in order to perform on-line motion for robot manipulators. All the approaches that have dealt with this problem have carried out some sort of simplification, many of them using heuristic rules.[16,18,19] The one we present is based on a simple but systematic method that restricts the degrees of freedom of the planning problem, and uses an off-line stage to ensure that this restriction is acceptable.

The path planning approach presented in this paper is based on what we call *motion stragegies*. Motion strategies are simple and effective ways of moving a robot that are designed for specific kinds of tasks. A motion strategy for a robot arm is implemented by restricting its motion to a subset of its configuration space, what we call *reduced subspace* (R-subspace), or C_R. We need to ensure that this restriction is correctly chosen, which is why we use an off-line stage that optimizes the choice of the reduced subspaces, and minimizes the probability of not finding a collision free path.

Therefore the path planning method we present is based on two stages:

- The off-line stage. This stage decides which R-subspaces (which strategies) are most suitable for thc robots.
- The on-line stage. In this stage the robots look for their path inside the Rsubspace chosen in the offline stage, using a fast and reactive algorithm based on local information.

Reducing the configuration space of a five or six degrees of freedom robot to a smaller subspace is a very strong restriction. One cannot expect that a unique subspace will solve all the motions. Our offline analysis is based on grouping similar conflicts into what we call *motion problems*. Motion problems are general types of conflicts that can be successfully solved using one motion strategy. Thus, when the robot faces a particular motion problem, it would use the strategy that is most suitable for that kind of conflict.

Once the motion problems of the system have been defined, our goal is to assign one strategy to each motion problem. In order to know which strategy is the most suitable for each motion problem we test the strategies by conducting a simulation that will give us an estimation of the probability of finding a collision free path. This stage is the main part of the off-line analysis and we call it *Estimation of the Probability of Faults* (EPF).

The structure of the method is represented in Figure 1. The off line stage starts with the definition of the motion problems based on the tasks that the system is to perform and the geometry of the robots. Next, the EPF analysis is performed and the result is a set of strategies that are suitable for each one of the motion problems. Once the strategies have been chosen, they are used for on-line motion using a local and reactive planning algorithm.

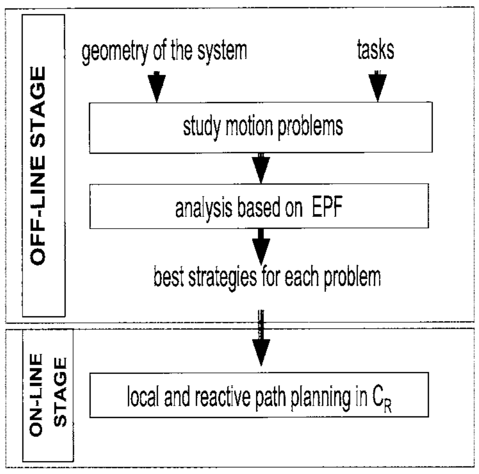

Fig. 1. Basic stages of stragtegy-based path planning.

The following sections describe these stages: First, in Section 2.1 the definition of strategies and R-subspaces is given. Section 2.2 describes the EPF analysis. Finally, Section 2.3 describes the on-line stage.

2.1. Strategies and reduced subspaces

The R-subspaces we have used in this first approach to the method are two dimensional linear subspaces of the configuration space, C. These subspaces have been chosen mainly because of their simplicity, but our method could be adapted to more complex subsets. These subspaces must contain the initial $\mathbf{q}_i$ and final $\mathbf{q}_f$ configurations of the robot, so that the robot can go from $\mathbf{q}_i$ to $\mathbf{q}_f$ inside C_R. R-subspaces are defined by two vectors: $\mathbf{u}_1$, $\mathbf{u}_2 \in C$ such that:

$$C_R = \{\mathbf{q} \in C \mid \mathbf{q} = x_1\mathbf{u}_1 + x_2\mathbf{u}_2\} \tag{1}$$

where x_1 and x_2 are the independent variables of the subspace. Vector $\mathbf{u}_1$ points in the direction of the M-line, which is the line that goes from the initial to the final configuration, and only depends on the task assigned to the robot:

$$\mathbf{u}_1 = \frac{\mathbf{q}_f - \mathbf{q}_i}{\|\mathbf{q}_f - \mathbf{q}_i\|} \tag{2}$$

$\mathbf{u}_1$ is used to implement the motion of the robot towards its goal. On the other hand, vector $\mathbf{u}_2$ describes a direction perpendicular to $\mathbf{u}_1$ that is used by the robot to avoid obstacles.

The vector $\mathbf{u}_2$ is the one that determines the strategy of the robot. For example, for our five-link robots we have used a simple parametrization of $\mathbf{u}_2$, that shows the strategy easily. We use an auxiliar vector $\mathbf{v}_2 = (\nu, \alpha, \beta, \gamma, \delta)$ whose coordinates are chosen by the user. Parameter α, which determines the motion of the first link of the robot inside C_R, is normally left unchanged. Parameter α is chosen by the user and indicates how much change in the motion of link 2 must be allowed in the R-subspace as compared to the change in link 1. In the same manner, parameters β, γ and δ control the motion of links 3, 4 and 5. Since it is easier to work with an orthonormal basis, we find $\mathbf{u}_2$ such that:

$$\mathbf{u}_2 \cdot \mathbf{u}_1 = 0 \tag{3}$$

$$\|\mathbf{u}_2\| = 1 \tag{4}$$

Vector $\mathbf{u}_2$, that fulfills Equations 3 and 4, can be expressed as:

$$\mathbf{u}_2 = \mu\mathbf{u}_1 + \mu\mathbf{v}_2 \tag{5}$$

Parameters μ and η can be calculated using Equations 3 and 4:

$$\mu = -\mu\mathbf{u}_1 \cdot \mathbf{v}_1 \tag{6}$$

$$\mu = \frac{1}{\|((\mathbf{u}_1 \cdot \mathbf{v}_2)\mathbf{u}_1 + \mathbf{v}_2)\|} \tag{7}$$

Using the notation given for a five link robot, we define motion strategies as the set of parameters $\Lambda = (\nu, \alpha, \beta, \gamma, \delta)$. These parameters determine the movements of the robot inside C_R. Motion strategies are chosen in such a way that the corresponding motion of the robot has a physical meaning. In a PUMA type robot, a motion strategy with $\nu = 0$, $\alpha > 0$, $\beta > 0$ y $\gamma > 0$, for example, is what we have called *go up* strategy. It corresponds to a proportional elevation of the second, third and fourth joints of the robot. In our scorbot robots, this strategy is very effective for pick and place tasks, since the robots avoid obstacles by lifting their arms. When a robotic system is composed of several robots, one has to think about a global strategy for the system. This strategy would be determined by the parameters ($\nu_1 \ldots \nu_m$; $\alpha_1 \ldots \alpha_m$, $\beta_1 \ldots \beta_m$, $\gamma_1 \ldots \gamma_m$, $\delta_1 \ldots \delta_m$) of the m moving robots. A strategy is similar to, but not exactly the same, as an R-subspace. One strategy might lead to two different R-subspaces if the initial and final configurations of the robots are different.

2.2. Analysis based on the estimation of the probability of faults

This section describes the off-line stage of our path planning algorithm. This stage can be seen as the training of the robots for specific tasks. The first part of this off-line stage is the choice of the tasks that we want the robots to be trained to do. These kinds of tasks are what we call motion problems. The definition of the motion problems depends on the purpose of the robotic system and must be defined by the user.

The second part of our off-line stage is the search of the strategy that minimizes the probability of fault. We define *fault* in this context as an unsuccessful motion of the robotic system. A motion of the robotic system is considered unsuccesful if any of the robots cannot accomplish its task using our strategy-based path planner, either because of a deadlock, a livelock or because a collision-free path inside C_R does not exist.

In this section we first define the motion problems in Section 2.2.1. The EPF tecnique has been applied by the authors in a simplified motion problem. This application will be described below in Section 2.2.2. Based on this, a description of the aplication of EPF to general motion problems is in Section 2.2.3.

2.2.1. Motion problems. To illustrate the idea of motion problems we shall use the example that has proved this method: the multi-manipulator system of our laboratory.[5–7] It is composed of three Scorbot-er IX robots with five degrees of freedom each and several working platforms. One of the platforms is located in the center and is used by the three robots as the assembly platform. The other three tables are located at both sides of each robot and are used for pieces and product release and storage. Each robot can access two side working areas and the central area.

Each robot has its own controller, that is communicated via RS-232 to a PC computer. The three computers of each robot are connected to each other via ethernet. A communications and control software that runs under Linux in the three computers[6] enables joint operation of the system. The tasks assigned to the system are mainly pick-and-place operations between the working platforms. Only gross motion is considered, and it is assumed that no joint manipulation of pieces is needed.

In our multi robot system we have defined, for example, four kinds of motion problems:

- *Problem 1.* Two of the robots of the working cell are stopped in the middle of the common space. The third robot must work around two obstacles. This problem resembles a breakdown of the system: The two robots that are stopped might have suffered a failure and are still at some point on a fixed trajectory, the third robot continues the degraded operation of the system working alone around two still obstacles.
- *Problem 2.* Two robots are doing pick and place operations moving between their two side tables. They might cross in the middle of the working cell.
- *Problem 3.* Three robots are doing pick and place operations between their side tables. They might cross in common areas.

Figure 2 shows a diagram of these problems. Problem 1 is the simplest one. It does not involve moving obstacles, but notice that the obstacles are not fixed, since the robots can be stopped in many different positions.

2.2.2. EPF for three degrees of freedom. Our first approach to EPF has been based on motion Problem 1, that deals with still obstacles. We have also reduced the degrees of freedom of the robots to three. Since there is only one moving robot with a configuration space of three degrees of freedom and C_R subspaces are two-dimensional, there is one extra degree of freedom. Thus we can choose one parameter, α, that will describe all the C_R subspaces that can be chosen. The vector $\mathbf{u}_2$ of the subspace is chosen in terms of this parameter as:

$$\mathbf{u}_2 = cos(\alpha)\mathbf{n}_1 + sin(\alpha)\mathbf{n}_2 \tag{8}$$

where $\mathbf{n}_1$ and $\mathbf{n}_2$ are two vectors of the null space of $\mathbf{u}_1$ ($\mathbf{n}_1 \cdot \mathbf{u}_1 = \mathbf{n}_2 \cdot \mathbf{u}_1 = 0$) that fulfill: $\mathbf{n}_1 \cdot (0, 0, 1) = \mathbf{n}_2 \cdot \mathbf{n}_1 = 0$. Table I shows several $\mathbf{u}_2$ vectors and the strategies that results from them (basically go-up and fold, since there are only three links). The estimation of the probability of faults has been done by taking several values of the parameter or at regular intervals, and, for each one of them, estimating the probabilty of fault, $P(\alpha)$. The optimum C_R subspace can be estimated graphically taking the one that minimizes $P(\alpha)$.

The estimation of $P(\alpha)$ has been done by thinking of it as a statistical experiment. Our aim is to estimate the probability of an R-subspace, defined by a value of α, leading to a successful trajectory in a task of motion Problem 0. This estimation is calculated by taking N random tasks that belong to Problem 1 and testing if a collision free path can be found for the moving robot inside this R-subspace. These tasks are defined by the position of the robot-obstacles. These robots are stopped along a fixed trajectory, what we call the direct trajectory. We have used two parameters, s_1 and s_2, that vary from 0 at the beginning of the direct trajectory of the robot and 1 at the end. The tasks of Problem 1 are chosen by taking random values of parameters s_1 and s_2, thus randomly varying the positions of the obstacles.

The result of these N tests (for each α) are r faults and $N-r$ successful motions. This is a binomial experiment $B(N, p)$ where N is the number of proofs, and $p(\alpha)$ is the (real) probability of faults (unknown). $p(\alpha)$ is estimated using the sampling probability $P(\alpha) = y = r/N$.

Binomial distributions $B(N, p)$ can be approximated by normal distributions $N(p, \sqrt{p(1-p)/N})$ with

$$Z = \frac{y-p}{\sqrt{p(1-p)/N}} \tag{9}$$

as long as $N \cdot p > 4$ and $N \cdot (1-p) > 4$. An interval approximation of p can be calculated as:

$$P(\alpha) \approx p = y \pm Z_{1-\frac{\xi}{2}} \sqrt{y(1-y)/N} \tag{10}$$

where ξ is the desired confidence interval.

If we want to distinguish between two estimated probabilities p_0 and p_a having type I and type II errors limited by α and β, the minimum number of experiments needed is:[22]

$$N_{min} = \frac{(Z_{1-\alpha}\sqrt{p_0(1-p_0)} + Z_{1-\beta}\sqrt{p_a(1-p_a)})^2}{(p_a - p_0)^2} \tag{11}$$

We have used in this Equation (11) the notation that is most frequent in statistics literature. Type I error is the one made when the hypothesis H_0: $p = p_0$ is true and is rejected. Type II error is made when the hypothesis is false and is accepted. Parameter or should not be confused with the one used in Section 2.1 to describe strategies.

We need to know the probabilities p_a and p_0 to calculate N_{min}. Since p_a and p_0 are not known a priori, the minimum number of experiments is calculated making $p_a = 0.5$ since this value maximices the number of tests.

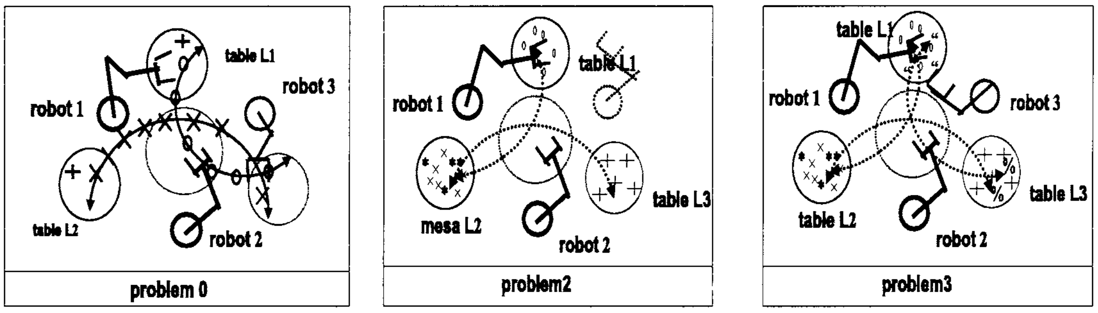

Fig. 2. Motion problems of our multi-manipulator system. Problem 1 involves two still robots acting as obstacles. In Problem 2 two robots cross while they go from one side table to the other. In problem 3 three robots cross.

Table I. Off-line analysis. Some of the strategies used in the off-line analysis can be seen in this table. These strategies are basically *fold* and *go up*.

α (rad)	$\mathbf{u}_2$	strategy	
0.10	(–0.03 0.21 0.97)	$q_2\uparrow q_3\upuparrows$	*go up*
0 52	(–0.01 0.49 0.86)	$q_2\uparrow q_3\upuparrows$	*go up*
1.04	(–0.02 0.86 0.50)	$q_2\upuparrows q_3\uparrow$	*go up*
1.57	(–0.02 0.99 0.00)	$q_2\upuparrows q_3\uparrow$	*go up*
2.09	(–0.02 0.86 –0.50)	$q_2\upuparrows q_3\downarrow$	*fold*
2.61	(–0.01 0.49 –0.86)	$q_2\uparrow q_3\downdownarrows$	*fold*
3.03	(–0.00 0.10 –0.90)	$q_2\uparrow q_3\downdownarrows$	*fold*

Using a number of $N \geq N_{min}$ situations and testing the strategy Λ_j on each of them, an estimation of the probability of fault of that strategy can be calculated as $P(\alpha)=y$. This estimation is possible if the experiment we carry out is binomial. This implies that the N observations must be independent. Since, in motion Problem 1, the position of the obtacle-robots is chosen randomly these experiments are independent.

2.2.3. EPF for more degrees of freedom. When the EPF analysis is applied to robots with more degrees of freedom, one must take into account the fact that the strategies of the system are described not only be a parameter but by a set of parameters. In this case the simple inspection of the function $P(\alpha)$ is not enough, and optimization techniques shall be used in order to find the minimum of P.

Since the function that we want to minimize is a highly nonlinear and not analytical function the optimization method that we have decided to use is the flexible polyhedron search (simplex method by Nelder and Mead)[23] which does not use the derivatives of the function.

If the EPF technique is applied to motion problems such as Problems 2 and 3, which are more interesting than Problem 1, the experiments will be defined in a different way. In these cases we have used a continuous simulation of the robotic system as our statistical experiment. In this simulation the robots are told to perform N tasks (that belong to the specific motion problem) using the strategy whose probability of fault we want to estimate. A task is considered faulty if the time spent on it by the robot is greater than a certain amount. Thus we can discard those strategies that lead to deadlocks or inefficient situations (livelocks). Such situations may happen, for example, if two robots try to avoid each other by lifting their arms. In this way, both robots occupy the upper part of the workspace and become an obstacle to each other. When the robots reach their joint limit both would change local direction (see the algorithm in Section 2.3) and then both would go down, thus they follow each other and are always an obstacle to each other. The robots can be trying to avoid each other forever.

Once the number of bad motions, r, is calculated the probabilty of faults, which will be called $P(\Lambda)$, can be estimated, taking into account the considerations mentioned above. One of the important aspects of this estimation is the fact that the experiments must be independent from each other for our estimation to be valid. Since we are doing a continuous simulation, the tasks might be correlated to each other and therefore, the results of our test might also be correlated. There are ways of breaking this dependency between tasks. We have used two: we choose the initial and final points of the motion of each robot randomly. Thus a source of randomness is introduced. Since the robots have tasks with random length the time they spend doing them changes, therefore, the movements of the three robots will soon be uncorrelated. Another source of randomness can be introduced by making the robots wait a random time before starting a new task. Thus the robots start their motions at random times and meet each other in random locations. These delay times cannot be very long, since otherwise, the robots would not meet each other very often, and our experiment would not be very instructive.

2.3. *On-line stage*

The on-line stage of our method is based on moving the robot inside an R-subspace using a reactive path planning algorithm.[24,25] The planning algorithm described in this section is very simple, but more complete algorithms used in mobile robots, such as the ones described in reference [26] could also be adapted to our method.

Our path planning method is based on a discrete algorithm that calculates discrete points in the configuration space This algorithm has been implemented in the multi manipulator system described in Section 2.2. Discrete points are generated on-line every 0.2 seconds by the control computers of the multi manipulator system. Some interpolation routines generate an interpolated trajectory between those points. We can assume that the robots follow this trajectory and errors are confined to safety areas. The algorithm is based on three behaviours:

- *Go to goal*. This behaviour is used whenever the robot finds obstacles at a distance greater than a certain amount h. This behaviour means that the robot moves following the line that goes from its actual position to the final configuration. That direction is pointed out at each moment by a vector called $\mathbf{u}^F$. If the position of the robot at one instant k is $\mathbf{q}_k$, the next time instant $\mathbf{q}_{k+1}=\mathbf{q}_k+\Delta\mathbf{u}_k^F$, with

$$\mathbf{u}_k^F=\frac{\mathbf{q}_f-\mathbf{q}_k}{\|\mathbf{q}_f-\mathbf{q}_k\|}.$$

- *Go around obstacle*. This behaviour is used when the first one cannot be followed. This behaviour consists of going around the obstacle following one of the two possible local directions (left or right). One of the local directions, the one used as default, is defined by vector $\mathbf{u}_k^T$, that is perpendicular to $\mathbf{u}_k^F$ at each moment k. $\mathbf{u}_k^T$ is perpendicular to $\mathbf{u}_k^F$ and belongs to C_R. It can be easily calculated if $\mathbf{u}_k^F$ is defined in terms of the base $(\mathbf{u}_1, \mathbf{u}_2)$. For example, if $\mathbf{u}_k^F=a\mathbf{u}_1+b\mathbf{u}_2$, the vector $\mathbf{u}_k^T=-b\mathbf{u}_1+a\mathbf{u}_2$ is perpendicular to $\mathbf{u}_k^F$ and indicates one of the two local directions. When the robot follows this behaviour using, for example, the right local direction, it checks the following configurations:

$$\mathbf{q}_k + \begin{cases} \Delta\mathbf{u}_k^F \\ \Delta\mathbf{u}_k^F - \Delta\mathbf{u}_k^T \\ -\Delta\mathbf{u}_k^T \\ -\Delta\mathbf{u}_k^F - \Delta\mathbf{u}_k^T \\ -\Delta\mathbf{u}_k^F \end{cases} \quad (12)$$

and chooses the first of them whose distance to the obstacles is greater than a threshold h. Thus the robot explores the neighbours of $\mathbf{q}_k$ following one local direction and chooses the neighbour that is closest to the goal configuration but whose distance to the obstacle is greater than h.

- *Change local direction*. When the two previous behaviours cannot be followed, the algorithm changes its default local direction, by making $\mathbf{u}^T = -\mathbf{u}^T$. The change of local direction usually tells us that the strategy has not been correctly chosen, since one strategy implies one default local direction. When the robot needs to use this behaviour we say that the path planning algoritm has failed.

3. EXPERIMENTS AND RESULTS

This section shows some of the experimental results obtained with the strategy-based path planning. These results are only some preliminary tests, but they show that the method proposed in this paper can solve on-line path planning of multi-manipulator systems.

Section 3.1 shows the results of the application of the off-line stage to motion Problem 1, as was described in Section 2.2.2. Section 3.2 presents some preliminary tests of on-line path planning for robots with five degrees of freedom.

3.1. Results of the off-line analysis

The off-line analysis described in Section 2 has been applied to our multi manipulator systems described in Section 2.2.2. Since this is the first approach to strategy based path planning we have applied the method to a simple problem. We have thus avoided the problems related to the high number of degrees of freedom of the robots in the first approach to the method. We have only used the first three degrees of freedom of the robots, and the motion problem that we have analysed is Problem 1, the simplest one. All these simplifications make these results less interesting than those presented in Section 3.2 but they show how the EPF analysis works.

The motion experiment we have done has the following characteristics:

- The degrees of freedom of the robots have been reduced to three. Only the first (waist), second (shoulder) and third (elbow) joint of the robot move, the hand is aligned with the forearm.
- We assume that two of the robots (robots B and C) have stopped at any point along their trajectory because of a fault and the third robot (robot A) has to move avoiding those two fixed obstacles. Robot A will be called the planner robot while B and C are the obstacle robots. The obstacle robots are stopped at any point of what we call the *direct trajectory*, which is the one they would follow if there were no obstacles in its path.
- The robots are closer to each other than in the real multi manipulator system. The distance between robots is 0.8 meters, thus the robots can touch each other's base, and the motion problem is a little more difficult.

The direct trajectory of robots B and C is described with parameters s_1 and s_2 that vary continuously from 0 at the starting point in one of the tables, to 1 at the target point in the other table. The positions of the obstacles can be described by a pair of numbers $S=(s_1, s_2)$. Knowing s_1 and s_2 one can calculate the joint coordinates of each of the obstacle robots and its location in the workspace.

This motion problem does not deal with real moving obstacles, but it is useful to teach the robots how to avoid a wide range of fixed obstacles. The trajectory of the obstacle robots can be seen in Figure 3, robot B travels in a normal pick and place trajectory between the two tables, but robot C travels by occupying the upper part of the space. Thus the workspace that robot A has to deal with is a very cluttered one. We have used the parameter or $\alpha \in (0, 2\pi)$ to describe the strategy of the robot, as was explained in Section 2.2.2. Some of the resulting strategies can be seen in Table I.

The EPF analysis has been applied to this motion problem by choosing 30 different R-subspaces and choosing 157 random situations $S_i=(s_1^i, s_2^i)$, $i=1 \ldots 157$ for each one of them. Those 157 tests give a confidence of 70% ($\alpha=\beta=0, 3$) to distinguish fault probabilities that differ by 5%, according to Equation 11 ($p_0=0.5$ and $p_a=0.45$). This confidence is not very high, but the results have been refined in those strategies that had the lowest fault probability (see Figure 4. We choose a different set of 157 situations for each strategy, thus making the experiment more random.

The estimation of the probability of fault in those R-subspaces can be seen in Figure 4. In the right graph of Figure 4 the results of the R-subspaces with the lowest probability of fault are shown in more detail using intervals (90% confidence). The minimum probability of fault is found in R-subspaces of α between 0.5 and 1.5 rad. In the right hand graph of Figure 4 there is a horizontal line between values 0.04 and 0.02, that indicates the estimated probability of finding a situation with no solution at all (in $\mathcal{C}$). This shows that about 4% of the situations cannot be solved by any method. We can also see that the estimated probability of fault for some of the R-subspaces cannot be distinguished fronn the probability of no solution cases. This means that, in terms of failure rate, the strategy based planner and the complete planner cannot be distinguished. In other words, EPF analysis shows that R-subspaces are capable of solving basically all the situations that have a solution for this type of problem.

These results show that in our system, the fixed obstacles created by obstacle-robots can be solved very successfully using a *go up* strategy. Notice that these include two obstacles with very different trajectories. The *go up* strategy is very successful in terms of avoiding fixed and big obstacles, since the robots tend to use the upper part of the workspace that is not occupied by any obstacle.

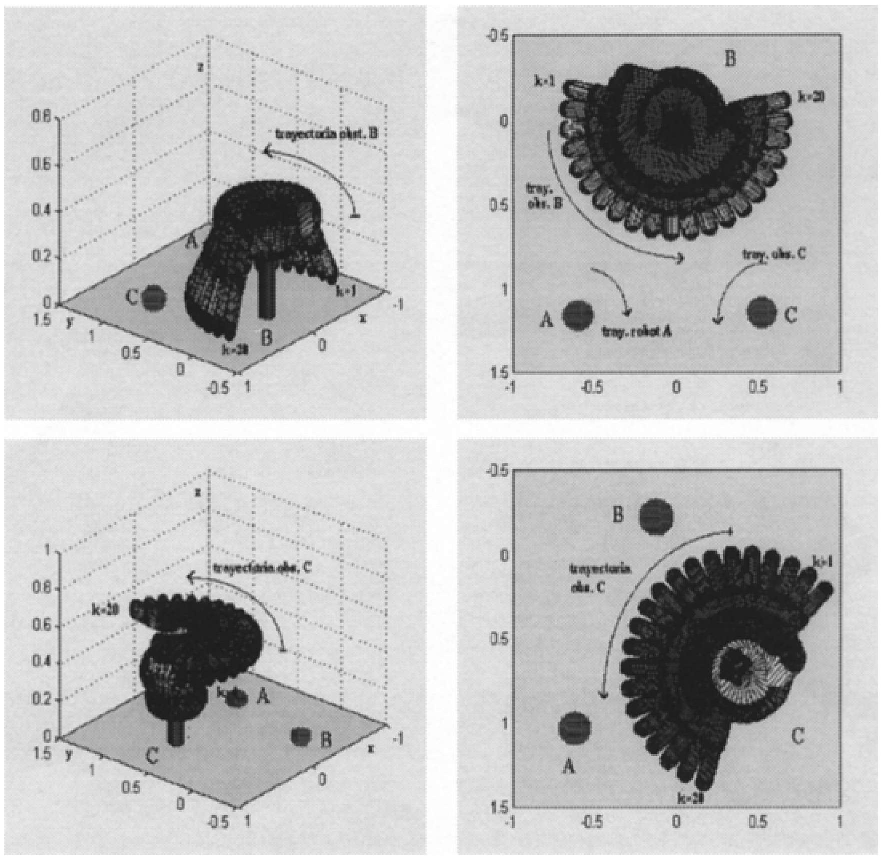

Fig. 3. Off-line analysis for motion problem 1. Situations of the obstacle robots.

3.2. *Results of on-line path planning*

In this section we present the results obtained using on-line path planning based on motion strategies in a system with two and three five-link robots. The strategies have been chosen without using the off-line stage, because off-line results are not ready for five link robots. The strategies were chosen by trial and error and thinking of which kind of motion would be most appropriate for the robots. In any case, these strategies are very successful and 100% success has been reported.

These strategies have been tested on a simulation of the systems operation. The robotic system is the one described in Section 2. The distance between two robot bases is 1.1 meters, while the maximum reach of the robot gripper is 0.846 meters. The first link of the robot (waist) has 300 degrees of joint reach, therefore, 30% of the area that can be

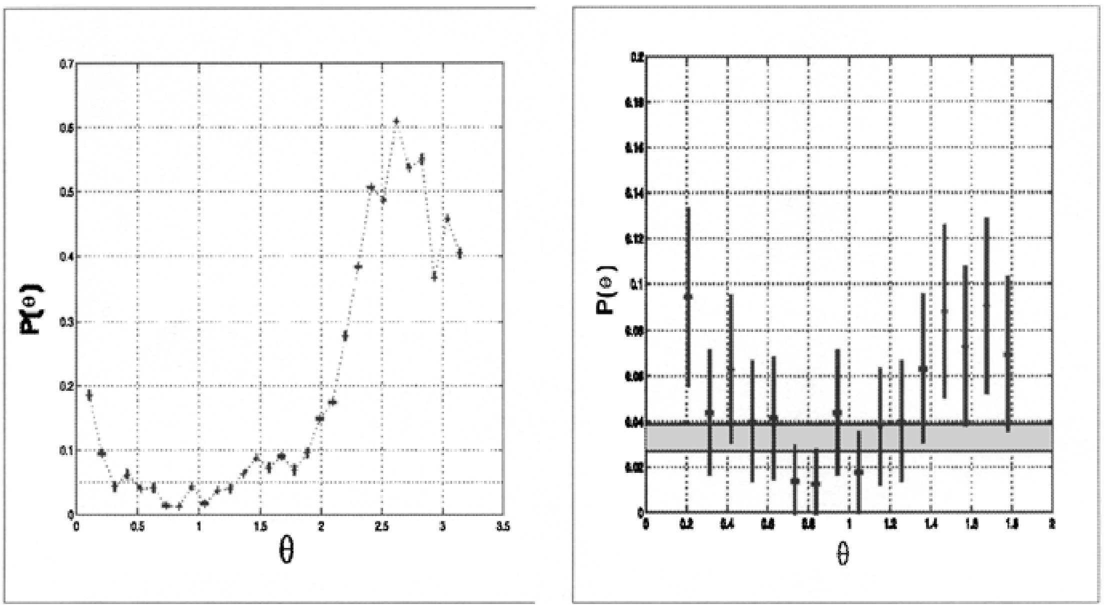

Fig. 4. Off-line analysis. Estimation of the probability of faults for some strategies. The horizontal band represents the probability of not finding a collision free path in *C*.

Table II. Results of several strategies on problem 2 (two robots cross).

ROBOT I	ROBOT 2	no. tests	no. success
go-up v_2=((0, 0.64, 1, 1, 0)	**go-up** $\mathbf{v}_2$=(0, 0.64, 1, 1, 0)	300	300
fold v_2=((0, 1, –1, –1, 0)	**fold** $\mathbf{v}_2$=(0, 0.51, –1, –1, 0)	300	300
fold $\mathbf{v}_2$=(0, 0.51, –1, –1, 0)	**go-up with hand down** v_2=((0, 1.53, 1.53, –2.51, 0)	300	300
go-up $\mathbf{v}_2$=(0, 0.67, 1, 1, 0)	**go-up with hand down** v_2=((0, 1.53, 1.53, –2.51, 0)	300	300

reached by robot 1 can also be reached by robot 2, and 30% of that area can also be reached by robot 3. Although the robots have five degrees of freedom, only the first four are important, since the robots do not carry big objects, and the rotation of the gripper, which is the fifth joint, does not change the shape of the robots. Still, the number of degrees of freedom of the overall system is large, since there are three robots involved. The collision avoidance problem is solved modeling the robots with semi-spherical cylinders.[21]

The motion problems were the ones described in Section 2.2 as Problem 2 and Problem 3. In these problems the robots cross while they are doing pick and place operations between the working areas located at both sides of each robot.

Problem 2: two robots cross. The first experiments described are examples of what we have called Problem 2 (Section 2.2).

In these experiments robot 1 goes back and forth between table L_1 and table L_2 (the final point of one task is the initial configuration of the next), and robot 2 goes back and forth between tables L_2 and L_3. The initial and final configurations are chosen randomly inside a sphere in the configuration space centered around the central configuration of each table, and with 0.1 radiand of radius, so that the robot gripper covers a circular area of 40 cm in diameter of the working table. Thus the initial and final points of each trajectory vary in each motion and the robots always meet in different situations. The motion is considered unsuccessful if the robot takes more that twice the time spent by the non-obstacle trajectory or if the local direction must change. This experiment has been carried out with different strategies, and the results can be seen in Table II.

Problem 3: three robots cross. In this section we compare our method in what we called Problem 3 with potential field methods. The potential fields described in reference [14] have been used in a system of three PUMA robots whose dimensions are the same as the ones in our multi manipulator system. The tasks have been simulated using the software developed by its authors: the ACT program by Aleph technologies. The multi robot system has been simulated by treating it as a single manipulator with 15 degrees of freedom. The tasks used in this experiment are similar to the ones used in the experiments with two robots: the robots go back and forth between working tables and initial and final configurations are randomly chosen inside the working tables. The results can be seen in Table III.

Although the number of experiments performed with potential fields is smaller than the one used with strategies, these data show that potential fields are clearly unable to solve this problem, while motion based on strategies does solve it easily. Potential fields are purely local methods. The only information they use is the one related to the closest obstacles. That is why they very often get blocked into local minima. These local minima make this method incapable of solving even the simplest motion problems. Path planning based on strategies also uses a local planning, but our method has one important advantage: it has the concept of strategy that gives the robot some sort of global information. This information makes the difference with potential fields: the problems that are easily solved with strategies cannot be solved at all with local methods as potential fields. Figure 5 shows one example of motion in our multi manipulator system using path planning based on strategies. The three robots are using the *fold* strategy.

3.3. Discussion concerning the results

The results obtained from the first test using the method show it is possible, using the strategy based method, to solve problems involving on-line path planning of several robots. These results are much better than the ones obtained using potential field methods, as can be seen in the comparison. This means that the method is promising and we hope to explore its full capabilities.

The off-line analysis was only carried our for a simple case and the results are coherent with those obtained on-line. A high percentage of successes are obtained by using strategies to solve pick and place tasks in a multi robot system.

Table III. Comparison of potential fields and strategy-based path planning, problem 3 (three robots cross).

PATH PLANNING METHOD	no. tests	no. success
strategy-based (fold vs. fold vs. fold) robot 1 $\mathbf{v}_2$=(0, 1, –1, –1, 0) robot 2 $\mathbf{v}_2$=(0, 0.51, –1, –1, 0) robot 3 $\mathbf{v}_2$=(0, 0.51, –1, –1, 0)	300	300
potential fields	50	0

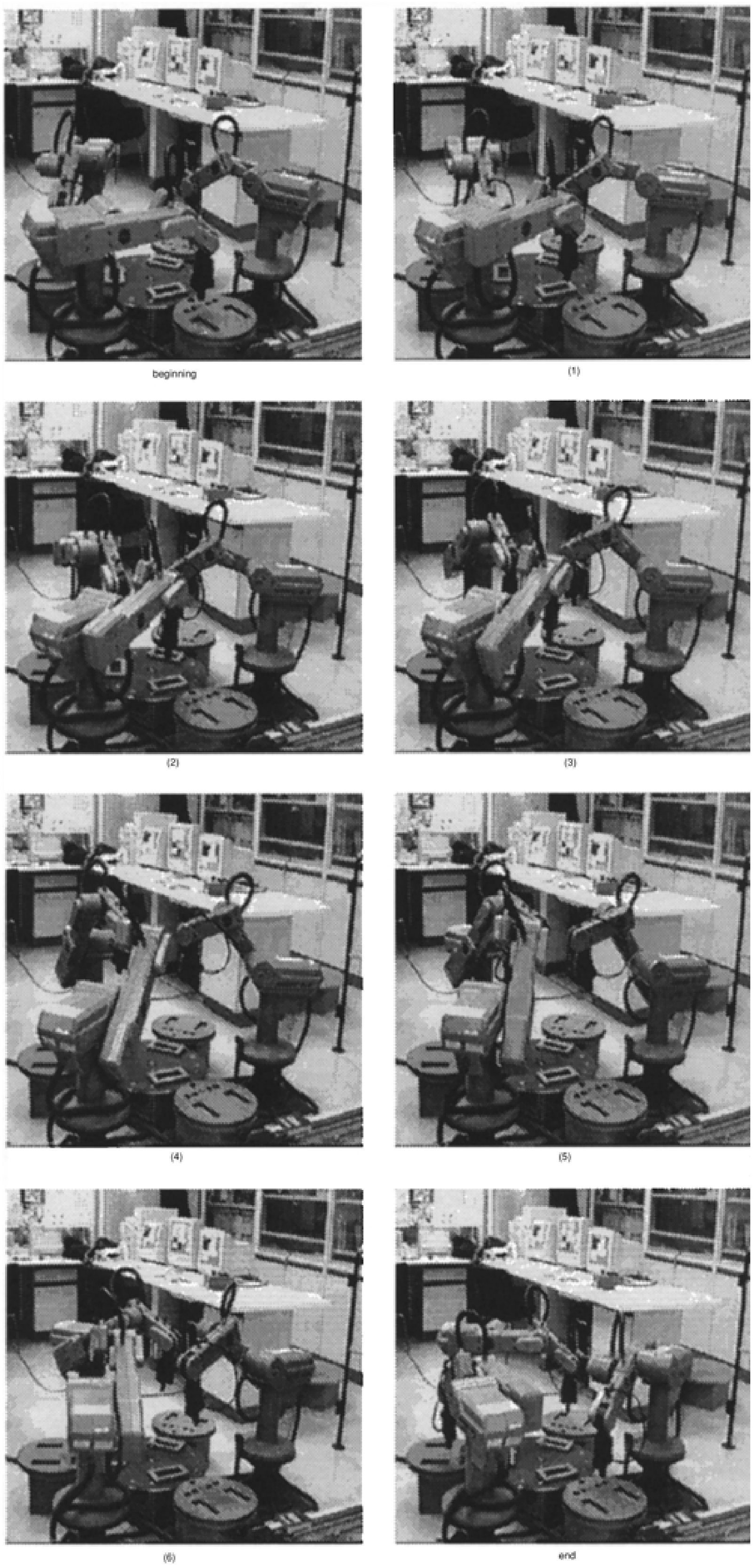

Fig. 5. Results of on-line path planning. Snapshots of a real execution of the multi manipulator system. The three robots use *fold* strategy.

4. CONCLUSION

A new approach to on-line path planning of robot manipulators is developed and demostrated in this paper. This approach achieves on-line and reactive path planning for a system of three five-link robot manipulators. The approach is based on decomposing motion planning into two stages: an on-line path planning stage and an off-line pre-processing stage. The on-line stage achieves very short computing times by restricting the search to a subset of the configuration space of the robot. The off-line stage chooses the subspace that minimizes the probability of not finding a collision free path. This method is a combination of global and local path plannning methods, and has the best qualities of both types of path planning approaches. This on-line path planning method is based on local information and therefore has the advantages of local methods: it is simple, reactive and computationally fast. On the other hand, our approach avoids the drawbacks of local approaches using an off-line stage that minimizes the probability of finding blockages and local minima. The approach is computationally fast since all the expensive calculations are moved into the off-line preprocessing stage. The approach has been tested on a multi-manipulator system of three five-link robots, and the results are successful: on-line path planning of two and three robots in pick and place operations is achieved.

References

1. J-C. Latombe, *Robot Motion Planning* (Kluwer Academic Publishers, Boston, 1991).
2. L. Kavraki, P. Svestka, J-C. Latombe and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation* **12**(4), 566–580 (1996).
3. X. Zhu and K. K. Gupta, "Practical global motion planning for many degrees of freedom: A novel approach within sequential framework," *J. Robotcs Systems* **12**(2), 105–117 (1995).
4. K. K. Gupta and A. P. Pobil (editors), *Practical Motion Planning in Robotics: Current Approaches and Future Directions* (John Wiley and Sons, 1998).
5. M. Mediavilla, J. C. Fraile and G. Dodds, "Optimization of collision free trajectories in multi robots system," *Proceedings of the IEEE International Conference on Robotics and Automation* (1998) pp. 2910–2915.
6. R. M. Merinero, J. C. Fraile and F. J. Garciá, "Design and implementation of a multi-robot system control," *Proceedings of the 3rd Portuguese Conference of Control. Controlo '98. Special Session on Robotics and Automation* (1998) pp. 826–828.
7. J. C. Fraile, C. J. J. Paredis, Cheng-Hua Wang and P. K. Khosla, "Agent-based planning and control of a multi-manipulator assembly system," *Proceedings of the IEEE International Conference on Robotics and Automation* (1999) pp. 1219–1225.
8. M. Mediavilla and J. C. Frail, "A multi robot system of three manipulators with on-line path planning abilities," *Proceedings of the LAAS-CNR5 9th International Symposium on Intelligent Robotic Systems*, Toulouse (2001) pp. 47–58.
9. L. Kavraki, M. N. Koluntzakis and J-C. Latombe, "Analysis of probabilistic roadmaps for path planning," *Proceedings of the IEEE International Conference on Robotics and Automation* (1996) pp. 3020–3025.
10. L. Gouzènes, "Strategy for solving collision-free trajectories problems for movile and manipulator robots," *Int. J. Robotics Research* **3**(4), 51–65 (1984).
11. J. Barraquand, B. Langloids and J-C. Latombe, "Numerical potential field techniques for robot path planning," *IEEE Transactions on Systems, Man and Cybernetics* **22**(2), 224–241 (1992).
12. D. Challou, D. Boley, M. Gini, V. Kumar and C. Olso, "Parallel search algorithms for robot motion planning." **In**: *Practical Motion Planning in Robotics: Current Approaches and Future Directions* (K. K. Gupta and A. P. Pobil, editors) (John Wiley and Sons, 1998).
13. O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *Int. J. Robotics Research* **5**(1), 90–98 (1986).
14. B. Faverjon and P. Tournassoud, "A local based approach for path planning of manipulators with a high number of degrees of freedom," *Proceedings of the IEEE International Conference on Robotics and Automation* (1987) pp. 1152–1159.
15. K. Souccar, C. Coelho, J. Conolly and R. Gruppen, "Harmonic functions for path planning and control," **In**: *Practical Motion Planning in Robotics: Current Approaches and Future Directions* (K. K. Gupta and A. P. Pobil, editors) (John Wiley and Sons, 1998) pp. 277–299.
16. T. Y. Li and J-C. Latombe, "On-line manipulation planning for two robot arms in a dynamic environment," *Int. J. Robotics Research* **16**(2), 144–167 (1997).
17. K. Hamilton and G. I. Dodds, "Reactive planning of robot arms in single and cooperative tasks." *Proceedings of the IEEE International Conference on Robotics and Automation* (1998) pp. 336–340.
18. S. Quinlan and O. Khatib, "Elastic bands: Connecting path planning and control," *Proceedings of the IEEE International Conference on Robotics and Automation* (1993) pp. 802–807.
19. H. Seraji and B. Bon, "Real-time collision avoidance for position-controlled manipulation," *IEEE Transactions on Robotics and Automation* **15**(4), 670–677 (1993).
20. Max Meng and Xianyi Yang, "A neural network approach to real-time trajectory generation," *Proceedings of the IEEE International Conference on Robotics and Automation* (1998) pp. 1725–1730.
21. B. Cao, G. Dodds and G. W. Irwin, "An approach to time optimal, smooth and collision free path planning in a two robot arm enviroment," *Robotica* **14**, Part 1, 61–70 (1996).
22. S. Lemeshow, D. W. Hosmer, J. Kler and S. K. Lwage, *Adequacy of Sample Size in Health Studies* (John Wiley and Sons, 1990).
23. D. M. Himmelblau, *Applied Nonlinear Programming* (McGraw-Hill, 1972).
24. R. C. Arkin *Behavior-Based Robotics* (MIT Press, 1998).
25. R. A. Brooks, "Elephants don't play chess," *Designing Autonomous Agents* (MIT Press, 1990) pp. 145–167.
26. H. Noborio and T. Yoshiaka, "Sensor-based navigation of a mobile robot under uncertain condition," **In**: *Practical Motion Planning in Robotics: Current Approaches and Future Directions* (K. K. Gupta and A. P. Pobil, editors) (John Wiley and Sons, 1998) pp. 325–347.