ON USING STOCHASTIC AUTOMATA FOR TRAJECTORY PLANNING OF ROBOT MANIPULATORS IN NOISY WORKSPACES

B.J. Oommen* **S. Sitharam Iyengar**** **Nicte Andrade***

*School of Computer Science, Carleton University, Ottawa, Canada : KIS 5B6.

**Dept. of Computer Science, Louisiana State University, Baton Rouge, LA:70803

ABSTRACT

We consider the problem of a robot manipulator operating in a noisy workspace. The robot is assigned the task of moving from $\mathbf{P}_i$ to $\mathbf{P}_f$. Since $\mathbf{P}_i$ is its initial position, this position can be known fairly accurately. However, since $\mathbf{P}_f$ is usually obtained as a result of a sensing operation, possibly vision sensing, we assume that $\mathbf{P}_f$ is noisy. We propose a solution to achieve the motion which involves a new learning automaton, called the Discretized Linear Reward-Penalty (DL_{RP}) automaton. The strategy we propose does not involve the computation of any inverse kinematics. Alternatively, an automaton is positioned at each joint of the robot, and by processing repeated noisy observations of $\mathbf{P}_f$ the automata operate in parallel to control the motion of the manipulator.The advantages and the possible disadvantages of the scheme are also discussed.

I. INTRODUCTION

Robotics is one of the most fascinating and interesting areas of engineering and computer science. Not only is it an area of great importance economically, but as a research area, robotics encompasses such fields as kinematics, mechanics, computational geometry, controls and language design.

One of the most interesting areas in robotics is the study of the problem of navigating a robot (or a manipulator) within a workspace. When the robot has no obstacles to avoid and is operating in a noise-free workspace, the problem is essentially a control problem. Solutions usually involve joint interpolated motions, when the trajectory is not necessarily linear, and motions computed using recursive algorithms (such as Taylor's algorithm) if the path desired is linear. However, the problem is far more complex if the robot (or manipulator) has to plan its motion when there are obstacles in its workspace or if the workspace is noisy.

At the next level of complexity the problems that are considered are those involving motion planning amidst obstacles.Since the literature in this field is so extensive, we refer the reader to a comprehensive survey of the papers and results in the area by Whitesides [15]. The subsequent question of much importance has been that of moving multiple objects. This problem is currently being studied by a fair number of researchers. Suffice it to say that the general problem of coordinating the motion of multiple independent objects was shown by Hopcroft, Schwartz and Sharir [8] to be PSpace-Hard. From a practical view point Grossaman et al. [7] of IBM considered the value of using multiple independent robot arms. Clearly, this "value" depends on the criterion function used to evaluate the performance of the multiple robots. Based on the criteria that they investigated, Grossman *et al*[7] concluded that in both one and two dimensions there is "little merit" in having more than **two** arms. A brief survey of the work done in all of the above areas is found in [23].

Although the body of work done in the area of robotics and motion planning is so extensive, the work done is still in its infancy when it concerns operating robots in real life workspaces subject to noisy and inaccurate measurements. Indeed, as Lozano-Perez remarked in an opening address of the 1985 SIAM Conference on Robotics and Geometric Modeling[13]: "Nothing is ever where it is supposed to be - is the first law of Robotics". He went on to say that "one is lucky to find one paper " on the topics of planning error and planning sensing strategies. In another context, in a personal communication to the first two authors, Lozano-Perez wrote "Error is the central problem in robotics, but it often gets left behind in the problem formulation", and this indeed is true.

I.1 Problem Statement

In this paper we consider the problem of a multi-link robot manipulator operating in a noisy workspace in which the joints of the robot can be prismatic and revolute. The robot is positioned at a configuration $\mathbf{P}_i$, which fully describes the position and orientation of its end-effector. The robot is commanded to move to a configuration $\mathbf{P}_f$, inside the workspace. $\mathbf{P_f}$ represents the desired ultimate position and orientation of the end-effector. Since $\mathbf{P_i}$ is completely defined by the joint angles of the manipulator, it is not unrealistic to assume that it can be obtained to any desired degree of accuracy. However, since the accuracy of $\mathbf{P_f}$, the goal position, cannot be arbitrarily specified by the designer of the manipulator, it is conceivable that the robot may be asked to move to a noisy version of the configuration, $\mathbf{P_f}$. This is especially true if the latter configuration is obtained as a result of a sensing process - customarily a vision sensing process. The problem which we tackle in this paper is indeed that of moving the robot from $\mathbf{P_i}$ to $\mathbf{P_f}$, where $\mathbf{P_f}$ is a fixed but unknown vector, which, furthermore, is unobservable. On the other hand, what is observable is a sequence $\{\mathbf{Q_f}(n)\}$, where,

$$\mathbf{Q_f}(n) = \mathbf{P_f} + \eta$$

and η is an i.i.d. random vector. We intend that the controller operates by processing $\mathbf{P_i}$ and $\{\mathbf{Q_f}(n)\}$. This problem of adaptively controlling a robot has been studied earlier (and that, especially in non-noisy environments). A review of the existing solutions reported in the literature [1-3,9,10,12,14] is given in [23] but omitted here for the sake of brevity.

We shall suggest a solution to the problem and consciously try to disengage ourselves from the traditional concepts of closed-loop feedback control theory. By this, we do not imply that the latter schemes are inferior. However, we aim to arrive at a solution which involves **absolutely no** estimation of parameters, **absolutely no** inverse kinematic computations, and **no** feedback control which is "hardware" oriented (i.e., which requires the tuning of servo-controllers etc.) More importantly, apart from the scheme being computationally attractive, the solution is highly parallelizable.

The strategy which we propose to employ involves using learning automata. Learning automata have been extensively studied in the literature and have been used to model biological mechanisms. They have also been used in pattern recognition, optimization, game playing and more recently even in object partitioning. These automata interact with an environment, and based on the responses of a noisy environment they attempt to learn the optimal action offered by the environment. We propose to use a learning automaton at every joint of the robot manipulator. This indeed involves merely maintaining a Finite State Machine at each joint, which dictates the motion that the particular joint has to make. Without using any other feedback arrangement except repeated noisy observations of $\mathbf{P_f}$, we propose to control the motion of the manipulator. Further, the control of the individual joints is achieved by having the automata operate **in parallel**. Finally, the feedback computations involved are of an elementary sort -- they involve updating the states of the Finite State Machine.

II. LEARNING AUTOMATA

Learning automata have been extensively studied by researchers in the area of adaptive learning. The intention is to design a learning machine which interacts with an environment and which dynamically learns the optimal action which the environment offers. The literature on learning automata is extensive. We refer the reader to a review paper by Narenda and Thathachar [18] and an excellent book by Lakshmivarahan [16] for a review of the various families of learning automata. The latter reference also discusses in fair detail some of the applications of learning automata.

By far, most of the research in this area has involved the category of machines called Variable Structure Stochastic Automata (VSSA). Automata in this category possess transition and output functions which evolve as the learning process proceeds. A VSSA is completely defined by a set of action probability updating functions [16,18,22].

VSSA are implemented using a Random Number Generator (RNG). The automaton decides on the action to be chosen based on an action probability distribution. Nearly all the VSSA discussed in the literature permit probabilities which can take any value in the range [0,1]. To minimize the requirements on the RNG **and to increase the speed of convergence** of the VSSA the concept of discretizing the probability space was recently introduced in the literature [19,20]. As in the continuous case, a discrete VSSA is defined using a probability updating function. However, as opposed to the functions used to define continuous VSSA, discrete VSSA utilize functions that can only assume a **finite** number of values. These values divide the interval [0,1] into a finite number of subintervals. If the subintervals are all of equal length the VSSA is said to be linear. Using these functions discrete VSSA can be designed - the learning being performed by updating the action probabilities in discrete steps.

In this paper we shall present a new discretized automaton which is the Multi-Action Discretized Linear Reward-Penalty (DL_{RP}) automaton. We shall prove that the Two-Action machine is ergodic and ε-optimal in all random environments whenever $c_{min} < 0.5$. Indeed this is the only known symmetric linear reward-penalty automaton which is ε-optimal in **any** random environment. We also show that the general Multi-Action Discretized Linear Reward-Penalty automaton is expedient. We shall then proceed to propose their application to the particular robotics problem.

II.1 Fundamentals and Learning Criteria

The automaton considered in this paper selects an action a(n) at each instant 'n' from a finite action set $\{ a_i \mid i = 1 \text{ to } R \}$. The selection is done on the basis of a probability distribution p(n), an R x 1 vector where, $\mathbf{p}(n) = [p_1(n), p_2(n), \ldots, p_R(n)]^T$ with, $p_i(n) = \Pr[a(n) = a_i]$, and,

$$\sum_{i=1}^{R} p_i(n) = 1 \qquad \text{for all } n. \tag{1}$$

The selected action serves as the input to the environment which gives out a response b(n) at time 'n'. b(n) is an element of B = {0,1}. The response '1' is said to be a 'penalty'. The environment penalizes the automaton with the penalty c_i, where, for all i,

$$c_i = \Pr[b(n) = 1 \mid a(n) = a_i]. \tag{2}$$

Thus the environment characteristics are specified by the set of penalty probabilities $\{c_i\}$ (i = 1 to R). On the basis of the response b(n) the action probability vector **p** (n) is updated and a new action chosen at (n+1). We define the reward probabilities as $1-c_i$ for $1 \le i \le R$. The penalty probabilities $\{c_i\}$ are unknown initially and it is desired that as a result of interaction with the environment the automaton arrives at the action which has the minimum expected penalty response. If L is this action, then $p_L(n) = 1$, $p_i(n) = 0$ for $i \neq L$ achieves this result. We seek for updating schemes for **p**(n) with this optimal solution in view.

With no *a priori* information, the automaton chooses the actions with equal probability. The expected penalty is thus initially M_0, the mean of the penalty probabilities. An automaton is said to learn **expediently** if the asymptotic expected penalty, E[M(n)], is less than M_0. The automaton is said to be **optimal** if E[M(n)] asymptotically equals the minimum penalty probability. It is ε-optimal if in the limit, $E[M(n)] < c_L + \varepsilon$, for any arbitrary $\varepsilon > 0$ by suitable choice of some parameter of the automaton. Thus the limiting value of E[M(n)] can be as close to c_L as desired.

III. THE DISCRETIZED LINEAR REWARD-PENALTY (DL_{RP}) AUTOMATON

III.1 The Two-Action DL_{RP} Automaton

The Discretized Linear Reward-Penalty (DL_{RP}) automaton has (N + 1) states where N is an **even** integer. We refer to the set of states as $S = \{ s_0, s_1, \ldots, s_N \}$. Associated with the state s_i is the probability i/N, and this represents the probability of the automaton choosing action a_1. Note that in this state the automaton chooses action a_2 with probability (1-i/N). Since any one of the action probabilities completely defines the vector of action probabilities, we shall, with no loss of generality, consider $p_1(n)$.

The basic idea in the learning process is to make **discrete** changes in the action probabilities. By defining the transition map as a function from S X B to S the changes in the action probabilities are indeed discrete. The transition map of the DL_{RP} automaton is specified by (3) below (and given schematically in [23]) for $s(n) = s_k$, $1 \le k \le N-1$.

$$\begin{aligned} s(n+1) &= s_{k+1} && \text{if } a(n) = a_1 \text{ and } b(n) = 0, \\ & && \text{or } a(n) = a_2 \text{ and } b(n) = 1 \\ &= s_{k-1} && \text{if } a(n) = a_1 \text{ and } b(n) = 1, \\ & && \text{or } a(n) = a_2 \text{ and } b(n) = 0. \end{aligned} \tag{3}$$

Observe that (3) is valid only for the interior states. Otherwise,

$$\begin{aligned} s(n+1) &= s(n) && \text{if } s(n) = s_0 \text{ or } s_N \text{ and } b(n) = 0 \\ &= s_1 && \text{if } s(n) = s_0 \text{ and } b(n) = 1 \\ &= s_{N-1} && \text{if } s(n) = s_N \text{ and } b(n) = 1. \end{aligned}$$

In lieu of the above, if $c_1 < c_2$, the automaton has no absorbing barriers except in the degenerate cases when $c_1 = 0$ or $c_2 = 1$. This implies that the underlying Markov chain is ergodic and that the limiting distribution of being in any state is independent of the corresponding initial distribution. The homogeneous Markov chain is defined by a stochastic matrix M whose arbitrary element $M_{i,j}$ is defined as :

$$\begin{aligned} M_{i,j} &= \Pr[s(n) = s_j \mid s(n-1) = s_i], \quad \text{where,} \\ M_{i,i-1} &= g_i c_1 + g'_i (1 - c_2) \text{ for } 1 \le i \le N, \\ M_{i,i+1} &= g'_i c_2 + g_i (1 - c_1) \text{ for } 0 \le i \le N-1, \\ M_{i,i} &= 0 \text{ for } 1 \le i \le N-1 \end{aligned} \tag{4}$$

where $g_i = i / N$ and $g'_i = 1 - i / N$. All the other elements of M are zero. Furthermore, the boundary conditions for the Markov chain are specified by :

$$M_{0,0} = (1 - c_2) \quad \text{and} \quad M_{N,N} = (1 - c_1). \tag{5}$$

The Markov chain consists of exactly one closed communicating class. Since it is aperiodic the chain is ergodic and the limiting distribution is independent of the initial distribution [2]. Let $\pi(n)$ be the state probability vector, where, for all n, $\pi(n) = [\pi_0(n), \pi_1(n), \ldots, \pi_N(n)]^T$, and, $\pi_i(n) = \Pr[s(n) = s_i]$, with,

$$\sum_{i=0}^{N} \pi_i(n) = 1.$$

Then the limiting value of π is given by the vector which satisfies (6) below and thus the following theorems follow.

$$M^T \pi = \pi. \tag{6}$$

Theorem I.

Let $\Delta = (c_1 + c_2 - 1)$. Then π_i, the ith component of the asymptotic probability vector obeys the following difference equation for $1 \leq i \leq N$.

$$\pi_i = \frac{c_2 - \Delta(\frac{i-1}{N})}{(1-c_2) + \Delta\frac{i}{N}} \pi_{i-1} \qquad i = 1, 2, \ldots, N.$$

Proof : The proof is quite involved and given in [23]. An alternate proof is given in [20]. •••

Theorem II.

The DL_{RP} automaton is ε-optimal whenever the minimum penalty probability is less than 0.5.

Proof : The proof is very involved and given in [23]. •••

The use of the two-action DL_{RP} automaton to achieve the manipulator control will be discussed later. Indeed, this automaton commands the joint that it controls to either go forward or go backward in a discretized joint space. A generalization of this motion requires the joint controller to go forward, go backward or stay at its current location. In order to understand the latter motion, we need to we study the design and the properties of the Multi-Action DL_{RP} automaton. Subsequently, we consider the use of these automata in trajectory planning.

III.2 The Multi-Action DL_{RP} Automaton

The R-Action DL $_{RP}$ automaton operate in a discretized probability space which divides the probability space [0,1] into NR intervals when $N \geq 1$ and does not divide the probability space at all if N= 0. The action-probability vector **p**(n) is defined by a set of probabilities, $[p_1(n), p_2(n), \ldots, p_R(n)]^T$, where $p_i(n)$ is the probability with which the automaton chooses action a_i satisfying (1). Note that apart from (1), we constrain each $p_i(n)$ such that it has to be a value on the discretized space. If, for the ease of notation,we omit the reference to the time instant n, the latter statement means that if $\delta = 1/NR$, then, apart from (1), p_i satisfies : $p_i \in \{i\,\delta \mid 0 \leq i \leq NR\}$.

The definition of the DL_{RP} automaton in the case of a reward (b(n) = 0) is as follows for $j \neq i$:

$$p_j(n+1) = \max(p_j(n) - \delta, 0) \qquad \text{if } a(n)=a_j,\ b(n)=0$$
$$= 1 - \sum_{j \neq i} \max(p_j(n) - \delta, 0) \qquad \text{if } a(n)=a_i,\ b(n)=0$$

The philosophy behind the above equation is quite straightforward. If a_i is chosen and the automaton is rewarded, then, each of the other p_j's is decremented by δ if it is positive. These decrements are then added to $p_i(n)$.

To describe the case of a penalty response, we define a function RandVect, whose input is an integer $k < R-1$, and j the index of an action. The output is a random subset $H_k(j)$ of k indices from the set {1,2,. . . ,R} - {j}. Using this notation, we define the updating rule as below:

$$p_i(n+1) = \max(p_i(n) - (R-1)\delta, 0) \qquad \text{if } a(n)=a_i,\ b(n)=1$$
$$= p_i(n) + \delta \qquad \text{if } a(n)=a_j,\ b(n)=1,\ p_j(n)=k\delta,\ \&\ k \geq (R-1)$$
$$= p_i(n) + \delta \qquad \text{if } a(n)=a_j,\ b(n)=1,\ p_j(n)=k\delta\ \ k < R-1 \text{ and } i \in H_k(j)$$

The philosophy behind the above equation is as follows. If $a(n)=a_i$ and the automaton is penalized, $p_i(n)$ is decremented by $(R-1)\delta$ if $p_i(n) \geq (R-1)\,\delta$, and this decrement is added to other actions, each action probability being incremented by δ. However, if the action probability of the action chosen has only a value of kδ, where $k<R-1$, this probability is decremented to zero, and k out of the (R-1) remaining action probabilities are randomly chosen and these are incremented by δ. The R-state DL_{RP} automaton has the following properties.

Theorem III

The R-state DL_{RP} automaton is expedient.

Proof : The result is proved in [23]. •••

Throughout the rest of the paper we will merely be considering the case when R= 3, primarily because the actions that we require that a joint controller take are those of going forwards one step in the discretized joint space, going backward one step and staying at the same joint angle. The transition map and the transition matrix of the case when R = 3 is explained in detail in [23] for N=1.We conjecture that the R-action DL_{RP} is ε-optimal whenever $c_{min} < 0.5$. The proof of the above conjecture will be quite involved. Indeed, it will require the solution of a stochastic tensor equation. However, simulation results seem to indicate that the conjecture is true.

IV. MANIPULATOR CONTROL USING THE DL_{RP} AUTOMATON

The strategy by which we control the manipulator can now be proposed, since the theoretical framework has been laid. Let us suppose we have a manipulator with K joints. These joints may be revolute or prismatic. The joint angles can be measured to yield the current position of the end-effector $\mathbf{P}_i$, and this is assumed to be done quite accurately. The robot is continuously fed with a noisy version of the Cartesian coordinates of the desired goal position of the end-effector $Q_f(n)$. Unfortunately, $Q_f(n)$ is noisy - and it represents the observable form of the **actual** goal position $\mathbf{P_f}$, the latter itself being unknown. Our intention is to adaptively control the manipulator so that it reaches arbitrarily close to $\mathbf{P_f}$.

The most straightforward method to achieve this "non-adaptively" is to take a large number of observations $\mathbf{Q_f}(n)$ of $\mathbf{P_f}$ and by computing the estimate of $\mathbf{P_f}$ from $\{\mathbf{Q_f}(n)\}$, any straightforward path planning strategy (for example, one using coordinated joint interpolated moves) can be utilized to move the end-effector from $\mathbf{P_i}$ to this estimate of $\mathbf{P_f}$. We do not recommend this strategy for two reasons. First of all, this scheme is non-adaptive. Secondly, in a real environment, the process of obtaining a large number of observations of $\mathbf{P_f}$ can be very time consuming for it could involve processing as many digital images of the workspace. Indeed, the robot will have to wait while all of these images are processed - before it even starts its motion. Also, once it does start its physical motion, the time involved is again a large portion of the time that the user has to spend, because, as is well known, the mechanical motions are often the most time consuming. The ideal scenario would be if the motion was planned piecewise, and as the planned motion is executed mechanically, the computer plans the next segment of the motion. Indeed, this can be achieved, for example, in VAL II by using motion commands which are suffixed by the symbol "!".

The strategy which we propose overcomes both the above drawbacks. Let us assume that the Two-Action DL_{RP} automaton is the learning machine that is used. Every joint of the robot is equipped with such a machine, and each joint independently chooses to either go forward or go backward in the discretized joint space. The way by which this motion of going forward or backward one step is implemented is, of course, robot dependent - and is easily achieved if the motors are stepped motors.

The learning process of the manipulator is described as follows. Let $\psi(n)$ be a criterion function, for example the Euclidian distance between $\mathbf{Q_f}(n)$ and $\mathbf{P_i}$ (n). Based on the actions stochastically chosen by the individual automata, the position of the end-effector of the robot moves to $\mathbf{P_i}(n+1)$. The observation $\mathbf{Q_f}(n+1)$ is now obtained, and the criterion function computed. If the latter function is less than it was in $\psi(n)$, each automaton is rewarded. Otherwise, the automata are penalized. Based on these responses the action probabilities are **locally** updated and the process repeated. When the criterion function is small enough the process is terminated, and a fine motion planning strategy is invoked.

In the case when the Three-Action DL_{RP} automaton is the learning machine, each joint is equipped with **such** a machine, and each joint is commanded stochastically to go forward, stay where it is, or go backward at every time instant based on the action chosen by the automaton. Based on this decision, the position $\mathbf{P_i}(n+1)$ of the end-effector is known, and using the new observation $\mathbf{Q_f}(n+1)$ the criterion function is recomputed to analogously reward or penalize all the automata.

Apart from our technique overcoming the drawbacks discussed above, it has one major advantage. The strategy **does not** require the computation of any **inverse** kinematics - and is thus computationally extremely effective. Indeed, inverse kinematic and inverse dynamic solutions can often have scores of parameters which are position, velocity and acceleration dependent. By permitting the automata to perform stochastically, and by repeatedly computing the straightforward criterion function, we have been able to avoid such tedious computations. Besides this, note that the automata make their decisions and update their probability vectors independently, and hence they can be made to operate **in parallel** - thus reducing the actual physical time that elapses.Furthermore, it must be noted that unlike most VSSA, maintaining the DL_{RP} automata involves just incrementing (or decrementing) **one** integer memory location per action, and furthermore the process of choosing an action involves invoking a random number generator exactly once per joint.

In both the scenarios which we cited above, all the automata were either simultaneously penalized or simultaneously rewarded. If each automaton should get a distinct response, one has to consider how each automaton is performing, as opposed to seeing how the collective performance of the automata is. Thus, in this case, a criterion function $\psi_i(n)$ can be computed for the ith joint, and the performance of this joint (in joint space) must be evaluated. But this requires - at the very least - a linearized model of the inverse kinematics and can be more expensive than the scenarios discussed above.

The technique that has been suggested in this paper has been rigorously tested for a few **simple** two-dimensional robots. The first robot R_1, is the familiar Horn's Robot [4] in which the robot operates in the plane, and the two joints are revolute joints. The second robot is the 3-link generalization of Horn's Robot in which the position and the orientation of the end-effector can be controlled.

Various experiments were conducted in which $\mathbf{P_i}$ was specified and noisy versions of $\mathbf{P_f}$ were generated using noise that had a Gaussian (Normal) distribution. Automata with two actions and three actions were independently used to control the manipulator. In each case a hundred experiments were performed and the expected ensemble path of the robot end-effector was traced. Also the expected decrease in the Euclidian distance from $\mathbf{P_i}(n)$ to final ideal goal position was also obtained. The graphs for a typical scenario are shown in Figures I(a) and (b) respectively for the case when NR=6 and R = 2. The decrease in the Euclidian distance from $\mathbf{P_f}$ as a function of n is shown in Figure I(b). The number of iterations required to converge to a point within the three standard deviation circle of the goal position in this case is approximately 60. A complete survey of some of the experimental results obtained using Two-Action and Three-Action automata and for the cases when the automata receive different responses is given in [23]. Similar results are also available for the three-link generalization of Horn's Robot [23].

V. CONCLUSIONS AND OPEN PROBLEMS

In this paper we have considered the problem of controlling a manipulator arm in which $\mathbf{P_i}$ the initial position of the end-effector is known, and the goal position is noisily sensed. The robot is required to move from $\mathbf{P_i}$ to $\mathbf{P_f}$, but instead of having $\mathbf{P_f}$ specified, a series of noisy observations, $\{Q_f(n)\}$, of $\mathbf{P_f}$ are available.

The solution which we propose involves using learning automata. A new learning automaton, the Two-Action DL_{RP} automaton has been introduced and proven to be ε-optimal wherever the minimum penalty probability is less than 0.5. The general R-action DL_{RP} automaton has been shown to be expedient, but conjectured to be ε-optimal in a similar environment. A DL_{RP} automaton is stationed at each joint of the robot, and these operate in parallel to control the individual joints of the robot.

Experimental results that demonstrate the power of the scheme for two simple two-dimensional robots have been presented. We intend to actually study the power of the scheme for a real life robot which has very primitive sensing operations.Also, preliminary simulation results using various learning automata seem to suggest that this strategy **could** lead to fascinating linear motion (or dog-chase) strategies which **do not involve** extensive inverse kinematic solutions. This avenue remains open. There are also a number of open stability problems such as the convergence of the robot at singularities and the stability analysis of robots which are stochastically receiving incremental motion commands.

ACKNOWLEDGEMENTS

The authors appreciate the help of Prof. Jens Peter Reus Christensen from Denmark for the proof of Theorem II. They are also grateful to Dr. I. Reichstein from Carleton University who proofread the final manuscript. This work was partially supported by the Natural Sciences and Engineering Research Council of Canada and the Oak Ridge National Labs.The work of the last author was supported by the Fondation Yvon Boulanger, Montreal, Canada : H2G 1Z9.

REFERENCES

References Concerning Robot Motion Planning

1. Arimoto, S., Kawamura, S., Miyazaki, F., and Tamaki S. ,"Learning Control Theory for Dynamical Systems", Proc. of the 24th Conf. on Decision and Control,1985, pp. 1375-1379.
2. Arimoto, S., and Takegaki, M.,"An Adaptive Trajectory Control of Manipulators", Int. J. Control, vol34,No. 2, 1981, pp. 219-230.
3. Azadivar, F., " The Effect of Joint Position Errors of Industrial Robots on Their Performance in Manufacturing Operations ", I.E.E.E. Journal of Robotics and Automation, vol. Ra-3, No.2, April 1987, pp. 109-114.
4. Brady, M., Hollerbach J.M., Johnson T.L., Lozano-Perez, T., Mason, M.T, "Robot Motion: Planning and Control".
5. Canny, J.F., " Collision Detection for Moving Polyhedra", A.I. memo 806. Oct. 1984. M.I.T. Artificial Intelligence Laboratory.
6. Craig, J.J., Hsu, P., and Sastry, S.S., " Adaptive Control of Mechanical Manipulators", Proc. of the IEEE Robotics and Automation Conference, April 1986, pp. 190-195.
7. Grossman, D.D., Evans, R.C. and Summers,P.D., "The Value of Multiple Independent Robot Arms", Robotics and Computer-Integrated Manufacturing, Vol.2,1985, pp 135-142.
8. Hopcroft , J.E.,Schwartz , J.T. and Sharir , M.," On the complexity of Motion Planning for Multiple Independent Objects; PSPACE-Hardness of the "Waterhouseman's Problem ", International Journal of Robotics Research,1984, pp76-88.
9. Koivo, A.J., and Guo, T., "Adaptive Linear Controller for Robotic Manipulators", I.E.E.E. Transactions on Automatic Control, vol. AC-28,1983,pp. 162-170.
10. Koivo, A.J., "Force-Position-Velocity Control with Self-Tuning for Robotic Manipulators", Proc. of the I.E.E.E. Robotics and Automation Conference, April 1986, pp. 1563-1568.
11. Lozano-Perez, T., "Spatial Planning: A Configuration Space Approach", IEEE Trans. Computers, Vol. C-32 , Feb. 1983, pp 108-120.
12. Miller III, W. T.," Sensor-Based Control of Robotic Manipulators Using a General Learning Algorithm", I.E.E.E. Journal of Robotics and Automation, Vol. RA 3, No. 2, April 1987, pp. 157-165.
13. Kozlov, A., "Robotics and Model Share Innovations", SIAM News, Vol. 18, No. 5, Sept. 1985, pp. 1,11.
14. Togai, M., and Yamano, O.,"Learning Control and its Optimality", Proc. of the IEEE Robotics and Automation Conference, April 1986, pp. 248-253.
15. Whitesides,S., "Computational Geometry and Motion Planning", Computational Geometry, Ed. by G. Toussaint, North Holland, 1985.

References Concerning Automata Learning

16. Lakshmivarahan, S., "Learning Algorithms Theory and Applications", Springer-Verlag, New York, 1981.
17. Lakshmivarahan, S., and Thathachar, M.A.L., "Absolutely Expedient Algorithms for Stochastic

Automata", IEEE Trans. on Syst. Man and Cybern., Vol. SMC-3, 1973, pp.281-286.

18. Narendra, K.S., and Thathachar, M.A.L., "Learning Automata -- A Survey", IEEE Trans. Syst. Man and Cybern., Vol. SMC-4, 1974, pp.323-334.

19. Oommen, B.J., "Absorbing and Ergodic Discretized Two-Action Learning Automata", IEEE Trans. on Syst. Man and Cybern., Vol. SMC-16, 1986, pp.282-296.

20. Oommen, B.J., and Christensen, J.P.R., "ε-Optimal Discretized Linear Reward-Penalty Automata Learning Automata", Submited for Publication.

21. Tsetlin, M.L., "Automaton Theory and the Modelling of Biological Systems", New York and London, Academic, 1973.

22. Varshavskii, V.I., and Vorontsova, I.P., "On the Behaviour of Stochastic Automata With Variable Structure", Automat. Telemek.(USSR), Vol.24, 1963, pp.327-333.

23. Oommen, B.J., Iyengar S.S. and Andrade, N.," Trajectory Planning of Robot Manipulators in Noisy Workspaces Using Stochastic Automata". (Submitted for Publication).

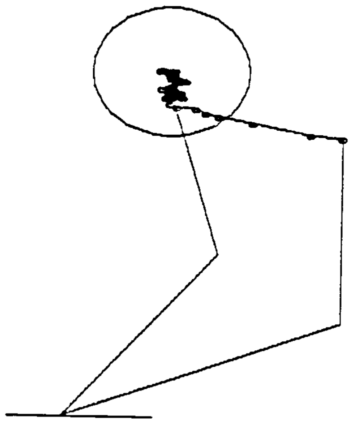

Figure Ia:

Expected Path (given as a sequence of points) of the two-link Horn's Robot with links of length unity. The control is achieved using the DL_{rp} automata with NR = 6 and with a single response controlling both the automata. In this case the standard deviation is 0.1 times the link length. The disk shows the three standard deviation range of noisy points.

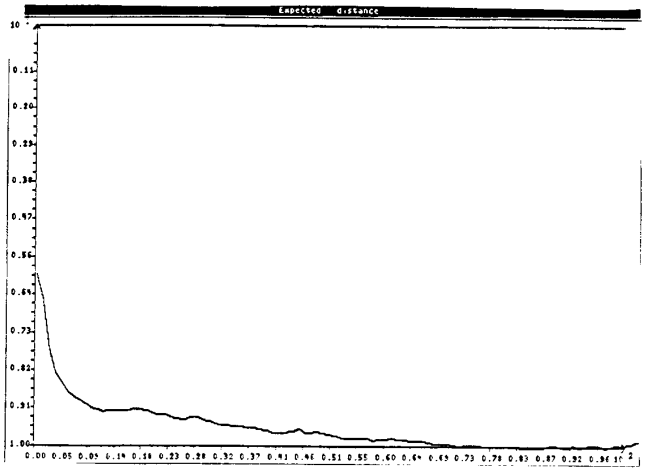

Figure Ib:

Expected change in distance from the initial configuration to the final mean configuration for the set-up described in Figure Ia.