TRAJECTORY PLANNING OF ROBOT MANIPULATORS IN NOISY WORKSPACES USING STOCHASTIC AUTOMATA

By B.J. Oommen[*], S. Sitharam Iyengar[**]
and Nicte Andrade[*]

SCS-TR-125

October 1987

School of Computer Science
Carleton University
Ottawa, Ontario
CANADA K1S 5B6

Partially supported by the Natural Sciences and Engineering Research Council of Canada. The work of the last author was supported by the Fondation Yvon Boulanger, Montreal, Canada H2G 1Z9

[*] School of Computer Science, Carleton University, Ottawa, Canada K1S 5B6
[**] Dept. of Computer Science, Louisiana State University, Baton Rouge, LA 70803

A preliminary version of this paper will be presented at the Fourth IEEE Conference on AI Applications, March 1988 at San Diego, California

TRAJECTORY PLANNING OF ROBOT MANIPULATORS IN NOISY WORKSPACES USING STOCHASTIC AUTOMATA[†]

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

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 usally 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.

[†]Partially supported by the Natural Sciences and Engineering Research Council of Canada.The work of the last author was supported by the Fondation Yvon Boulanger, Montreal, Canada : H2G 1Z9.

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

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

I. INTRODUCTION

Robotics is probably 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.

The initiator for the work in motion planning amidst obstacles was probably Udapa[30]. He introduced the concept of working in configuration space, and later this was studied extensively by many researchers including Brooks et al. [4,5], Lozano Perez et al. [21,22], and many others. Simutaneously, a variety of results concerning the theoretical issues of motion planning were presented by researchers among whom are Schwartz, Sharir [26] and Yap. 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 [31]. We strongly recommend the latter survey [33] for a researcher who is just embarking on working in this rather extensive field.

Whereas the problem of moving a single mobile robot in known terrains was almost completely investigated, the problem of navigating in unknown terrains was almost completely untouched till the last few years. The study of the problem of navigating mobile robots sparked a whole series of very interesting results. Learned paths were suggested by Iyengar et al. [16] who also suggested using learned spatial graphs [15] to compute the path of the robot. Oommen et al. presented a formal approach to tackling the problem using learning visibility graphs [24]. Although their solution was suitable only for a point robot, we believe that it is conceptually an ideal working model, in as much as the fact that the learned visibility graph actually converges to the actual visibility graph. Alternate solution which exist for **practical** robot navigation systems have been proposed, which involve map making [4] environment learning [7], discretization of the terrain [10], and the use of multilevel planning [11]. Other papers involving mobile robot navigation are those of Gouzenes [12] and Chattergy [8].

The subsequent question of much importance has been that of navigating (or moving) multiple objects. Schwartz and Sharir [26] presented an analytic solution for the special case in which the objects to be moved were circular and the obstacles were polygonal. 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 [14] to be PSpace-Hard. From a practical view point Grossaman et al. [12] 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[12] concluded that in both one and two dimensions there is "little merit" in having more than **two** arms.

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[27]: "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.

In this paper we shall concentrate on the problem of controlling a robot manipulator in a noisy workspace. If the robot is a **mobile** robot, and the obstacles are fixed but unknown, a learning strategy can be used to learn the model of the world during (or alternatively prior to) the navigation process [7,15,16,24]. However, even in this case, the problem has been only marginally studied when the observations obtained by the (vision or sonar) sensors are erroneous. However, when the robot is not mobile, but the manipulator joints themselves have to be considered, the problem has a different flavour.

I.1 Problem Statement and Brief Survey of Solutions

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 is unobservable. However, $\{\mathbf{Q}_f(n)\}$ is a sequence of observable points, where,

$$\mathbf{Q}_f(n) = \mathbf{P}_f + \eta$$

and η is an i.i.d. random vector. It is intended that the controller operates by processing $\mathbf{P}_i$ and $\{\mathbf{Q}_f(n)\}$.

Earlier, Azadivar[3] studied the problem of incorporating the positional error associated with the individual joints of the robots in the motion planning. He did this by actually estimating success and failure parameters which fed into his optimization and feedback control loop. Arimoto et al[1] studied mechanical and mechatronics systems with linear and nonlinear dynamics, which may be operated repeatedly at low cost. Given a desired output y_d over a finite time duration $[0,T]$ and an appropriate input u_0, these laws are formed by the following simple iterative processes:

(1) $u_{k+1} = u_k + \Phi(y_d - y_k)$,

(2) $u_{k+1} = u_k + \Gamma d/dt\, (y_d - y_k)$, and

(3) $u_{k+1} = u_k + (\Phi + \Gamma\, d/dt)\, (y_d - y_k)$,

where u_k and u_{k+1} denote the kth and (k+1)st input values, Φ and Γ are positive-definite constant gain matrices, and y_k is the measured output at the kth time instant. Arimoto et al [1] showed that the first law (1) with an appropriate gain matrix Φ is convergent in the sense that $y_k(t)$ approaches $y_{d(t)}$ as $k \rightarrow \infty$ in the meaning of the $L^2[0,T]$ norm if the objective system is linear and strictly positive. The same conclusion was proved when the system was subject to a linear time-invariant or time-varying mechanical system. In addition, a rough sketch of the convergence proof for the second and third learning control laws was presented for a class of linear and nonlinear dynamical systems.

In [9], Craig et al presented an adaptive version of the computed-torque method for the control of manipulators with rigid links. The algorithm estimated parameters on-line which appear in the nonlinear dynamic model of the manipulator, such as load and link mass parameters and friction parameters, and uses the latest estimates in the computed torque servo. The authors proved the global convergence and stability of such a scheme in its nonlinear setting, as well as its asymptotic properties and derived the conditions for parameter convergence.

Koivo and Guo[18] presented a new approach to the position and velocity control of a manipulator by using an adaptive controller of the self-tuning type for each joint. The complicated manipulator system was modeled by a set of time series difference equations. The parameters of the models were determined by on-line recursive algorithms which resulted from minimizing the sum of the squared errors. The adaptive controller of each joint was designed on the basis of the difference equation model and a chosen performance criterion function. The controller gains were then computed on-line using the model with the estimated values of system parameters. Later, in 1986, Koivo studied [19] the force-path control of a robotic manipulator in both the joint and the Cartesian coordinate systems. An autoregressive model with external excitation (ARX-model) was introduced for designing an adaptive controller with self-tuning. The controller minimized the conditional expectation of the sum of a quadratic position (velocity) error and a quadratic force error while satisfying the constraint of the ARX-model in which the estimated parameters were substituted for the unknown parameters. The basic approach was used to obtain an adaptive controller which operated on the variables which were expressed in terms of the joint coordinates. Another adaptive controller was similarly determined for the system variables expressed in terms of the Cartesian coordinates. The adaptive controller for force-path control in [19] has the form similar to that of hybrid force/position controller, but the former has time varying gains.

Going deeper into the traditional control systems concepts, Miller[23] described a practical learning control system which is applicable to the control of complex robotic systems involving multiple feedback sensors and multiple command variables during both repetitive and nonrepetitive operations. In the controller, Miller [23] used a general learning algorithm to learn to reproduce the relationship between the sensor outputs and the system command variables over particular regions of the state space of the system. The learned information was then utilized to predict the command signals required to produce desired changes in the sensor outputs. The learning controller required no *a priori* knowledge of the relationships between the sensor outputs and the command variables, thus facilitating the modification of the control system for specific applications.

The above algorithms are just a few representative examples of solutions that are applicable to the problem we are studying. Indeed, any of the solutions found in traditional textbooks on optimal control and adaptive control can also be tailored to solve this particular problem.

In this paper, 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.

In the next section we shall discuss the elementary concepts of learning automata and discuss in detail the automaton which we use in this problem. We then discuss the use of the automaton in the field robotics.

For the sake of convenience, we have listed the references in two distinct sections. Whereas references [1] through [31] are a brief collection of paper in robotics and motion planning, references [32] through [52] survey the field of learning automata. We believe that such a distinct survey will make the paper more complete.

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 [39] and an excellent book by Lakshmivarahan [33] 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 which include game playing [35], pattern recognition and hypothesis testing [39], priority assignment in a queueing system [37] and telephone routing [40,41]. Applications not found in [33] include the solution of stochastic geometric problems using learning automata [45] and the partitioning of objects using various types of automata [46,47].

Broadly speaking, learning automata can be classified into two categories : Fixed Structure Stochastic Automata (FSSA), and Variable Structure Stochastic Automata (VSSA). A Fixed Structure Stochastic Automaton (FSSA) is one whose transition and output functions are time invariant. Examples of such automata are the Tsetlin, Krylov and Krinsky automata [49,50]. By far, most of the research in this area has involved the second category, namely, Variable Structure Stochastic Automata (VSSA). Automata in this category possess transition and output functions which evolve as the learning process proceeds. It can be shown that a VSSA is completely defined by a set of action probability updating functions [38,39,52].

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]. Hence the RNG must theoretically possess infinite accuracy. In practice, however, the probabilities are rounded off to a certain number of decimal places depending on the architecture of the machine that is used to implement the automaton.

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 [42,46]. 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.

Learning automata can also be broadly classified in terms of their Markovian representations. Generally speaking, learning automata are either ergodic [40,43,44-47,49] or possess absorbing barriers [36,39,42]. Automata in the former class converge with a distribution which is independent of the initial distribution of the action probabilities. This feature is desirable when interacting with a non-stationary environment - for the automaton does not "lock itself" into choosing any one action. However, if the environment is stationary an automaton with an absorbing barrier is preferred. Various absolutely expedient schemes which ideally interact with such environments have been proposed in the literature [33,36,38,39].

In this paper we shall present a new discretized automaton which is the Two-Action Discretized Linear Reward-Penalty (DL_{RP}) automaton. We shall prove that the 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 shall also consider the three-action Discretized Linear Reward-Penalty (DL_{RP}) automaton and show that it is expedient. We shall then state the advantages of these automata over the traditional learning automata and proceed to propose their application to the particular robotics problem.

I.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, **p** (n) = $[p_1(n), p_2(n), \ldots, p_R(n)]^T$ with,

$$p_i(n) = \Pr[a(n) = a_i],$$
$$\sum_{i=1}^{R} p_i(n) = 1 \qquad \text{for all n.} \qquad (1)$$

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,

$$c_i = \Pr[b(n) = 1 \mid a(n) = a_i] \qquad (i = 1 \text{ to } R). \qquad (2)$$

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 $\{ c_i \}$ are unknown initially and it is desired that as a result of interaction with the environment the automaton arrives at the action which evokes the minimum penalty response in an expected sense. It may be noted that if L is the action which obeys,

$$c_L = \min_i (c_i) \qquad (3)$$

then $p_L(n) = 1$, $p_i(n) = 0$ for $i \neq L$ achieves this result. Updating schemes for **p** (n) are to be chosen 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, as time tends towards infinity, the expected penalty is less than M_0. We denote the expected penalty at time 'n' as E[M(n)]. The automaton is said to be **optimal** if E[M(n)] equals the minimum penalty probability in the limit as time goes towards infinity.

It is ε-optimal if in the limit $E[\, M(n)\,] < c_{min} + \varepsilon$ where $c_{min} = \min \{ c_i \}$, 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_{min} 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 (4) below for $s(n) = s_k$, $1 \le k \le N-1$.

$$\begin{aligned} s(n+1) &= s_{k+1} && \text{if } a(n) = a_1 \text{ and } b(n) = 0, \\ & && \text{or } a(n) = a_2 \text{ and } b(n) = 1 \\ &= s_{k-1} && \text{if } a(n) = a_1 \text{ and } b(n) = 1, \\ & && \text{or } a(n) = a_2 \text{ and } b(n) = 0. \end{aligned} \tag{4}$$

Observe that (4) is valid only for the interior states. For the end states :

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

Figure I shows the transition map of the automaton schematically.

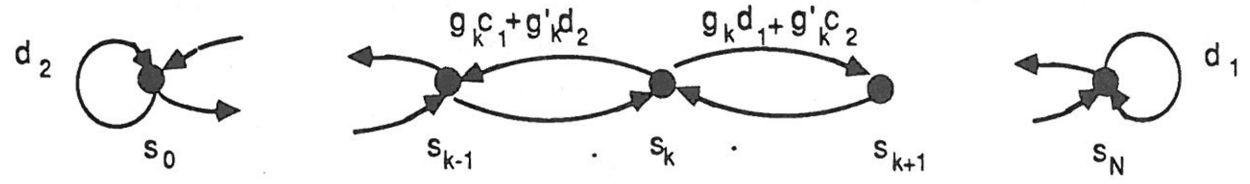

Figure I : Transition map of the DL_{RP} automaton

Observe that if the machine is in state s_0 it has to choose a_2 and similarly if it is in s_N it has to choose a_1. Thus the change in action probabilities can be written for $0<p_1(n)<1$ as :

$p_1(n+1) = p_1(n) + 1/N$	if a_1 is chosen and $b(n) = 0$ or a_2 is chosen and $b(n) = 1$
$= p_1(n) - 1/N$	if a_1 is chosen and $b(n) = 1$ or a_2 is chosen and $b(n) = 0$.

(5)

At the end states the following equality holds :

$p_1(n+1) = p_1(n)$	if $p_1(n) = 0$ or 1	and $b(n) = 0$
$= 1/N$	if $p_1(n) = 0$	and $b(n) = 1$
$= 1 - 1/N$	if $p_1(n) = 1$	and $b(n) = 1$.

The way in which the action probabilities are updated warrants the name of the automaton.

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 Markov chain is ergodic and that the limiting distribution of being in any state is independent of the corresponding initial distribution [2]. The homogeneous Markov chain is defined by a stochastic matrix M whose arbitrary element $M_{i,j}$ is defined as :

$$M_{i,j} = \Pr[\, s(n) = s_j \mid s(n-1) = s_i \,], \quad \text{where,}$$

$$
\begin{aligned}
M_{i,i-1} &= g_i c_1 + g'_i (1 - c_2) && \text{for } 1 \le i \le N, \\
M_{i,i+1} &= g'_i c_2 + g_i (1 - c_1) && \text{for } 0 \le i \le N-1, \\
M_{i,i} &= 0 \quad \text{for } 1 \le i \le N-1
\end{aligned} \tag{6}
$$

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 :

$$M_{0,0} = (1 - c_2) \quad \text{and} \quad M_{N,N} = (1 - c_1). \tag{7}$$

The Markov chain consists of exactly one closed communicating class. Further, 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,

$$\pi(n) = [\pi_0(n), \pi_1(n), \ldots, \pi_N(n)]^T, \qquad \pi_i(n) = \Pr[s(n) = s_i], \text{ and,} \tag{8}$$

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

Then the limiting value of π is given by the vector which satisfies,

$$M^T \pi = \pi \tag{9}$$

Using (9) we now derive the asymptotic properties of the DL_{RP} automaton [32].

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 :

By definition, the limiting equilibrium probability vector π satisfies

$$M^T \pi = \pi,$$

where, π is defined by (8) above [32].

Consider the stationary ergodic Markov chain defined by M. Suppose that starting at time n, we consider the sequence of states s(n), s(n-1),s(n-2),...., going backward in time. It turns out that this sequence of states is itself a Markov chain with transition probabilities $Q_{i,j}$ defined as below for state indices i and j :

$$\begin{aligned} Q_{i,j} &= \Pr[s(m) = j \mid s(m+1) = i] \\ &= \frac{\Pr[s(m) = j,\ s(m+1) = i]}{\Pr[s(m+1) = i]} \\ &= \frac{\Pr[s(m) = j] \Pr[s(m+1) = i \mid s(m) = j]}{\Pr[s(m+1) = i]} \\ &= \pi_j M_{j,i} / \pi_i \end{aligned}$$

Furthermore, consider $\Pr[s(m)=j \mid s(m+1)=i, s(m+2)=i_2, s(m+3)=i_3, \ldots, s(m+k)=i_k]$, where $j, i, i_2, i_3, \ldots,$ and i_k are state indices. Indeed, this is equal to $Q_{i,j}$ since,

$$\Pr[s(m) = j \mid s(m+1) = i, s(m+2) = i_2, \ldots, s(m+k) = i_k]$$

$$= \frac{\Pr[s(m) = j, s(m+1) = i, s(m+2) = i_2, \ldots, s(m+k) = i_k]}{\Pr[s(m+1) = i, s(m+2) = i_2, \ldots, s(m+k) = i_k]}$$

$$= \frac{\pi_j M_{j,i} \Pr[\ s(m+2) = i_2, \ldots, s(m+k) = i_k \mid s(m+1) = i\ ]}{\pi_j \Pr[\ s(m+2) = i_2, \ldots, s(m+k) = i_k \mid s(m+1) = i\ ]}$$

$$= Q_{i,j}.$$

If $Q_{i,j} = M_{i,j}$ for all i, j, then the Markov chain is said to be time reversible[32], and in this case, the stationary probabilities obey :

$$\pi_i M_{i,j} = \pi_j M_{j,i} \qquad \text{for all } i,j. \tag{10}$$

The above implies that for all states s_i and s_j, the rate at which the process goes from s_i to s_j, which is $\pi_i M_{i,j}$, is equal to the rate at which it goes from s_j to s_i, which is $\pi_j M_{j,i}$.

In the case of the DL_{RP} automaton, since the chain can only make transitions from any state to one of its two nearest neighbors, without the need of any extensive evaluations, it is possible to argue, that the Markov chain is time reversible. The argument is as follows : Between any two transitions from s_i to s_{i+1} there must be one from s_{i+1} to s_i (and vice versa) and since the only way to re-enter s_i from a higher state is via state s_{i+1}, one can argue that for the Markov chain, the number of transitions from s_i to s_{i+1} must at all times be within 1 of the number of transitions from s_{i+1} to s_i. Thus, the rate of transitions from s_i to s_{i+1} equals the rate from s_{i+1} to s_i, rendering the process time reversible.

Thus we can easily obtain the limiting probabilities by solving for (10) for each state index i=0,1,. . ., N-1. Explicitly, for all i=0,1,. . ., N-1,

$$\pi_i(1 - M_{i,i+1}) = \pi_{i-1}(M_{i-1,i}). \tag{13}$$

But, $M_{i,i+1} = (g'_i\ c_2 + g_i(1-c_1))$, and $M_{i-1,i} = (g'_i\ c_2 + g_{i-1}(1-c1))$, where $g_i = i / N$ and $g'_i = 1-i / N$. Equation (13) yields the desired result after substituting the latter values and performing some simplifying algebraic manipulations. ●●●

Theorem II.

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

Proof :

With no loss of generality let a_1 be the optimal action (i.e., let $c_1 < c_2$). It is remains to be proved that $E[p_1(\infty)]$ tends to unity as $N \to \infty$ if and only if $c_1 < 0.5$, where,

$$E[p_1(\infty)] = \sum_{i=0}^{N} \left(\frac{i}{N}\right) \pi_i$$

We consider three mutually exclusive and exhaustive cases.

Case I : $c_2 > 0.5 > c_1$.

From (13) we can see that if $c_2 > 0.5 > c_1$, then,

$$\pi_i \geq q\, \pi_{i-1}$$

where $q > 1$ for all i.

This easily implies that as $N \to \infty$ the major part of the probability measure on π is contained in an arbitrarily small neighbourhood of unity. Thus,

$$\lim_{N \to \infty} E[p_1(\infty)] = \lim_{N \to \infty} \sum_{i=0}^{N} \frac{i}{N} \pi_i \quad \to \quad 1.$$

Case II : $0.5 \geq c_2 > c_1$.

$$\text{Let } i_0 = N\left[\frac{1}{2\ N} + \frac{1 - 2c_2}{2(1 - (c_1 + c_2))}\right].$$

Note that i_0 need not be an integer. For the sake of notation, let the ratio of π_i to π_{i-1} in (13) be q_i. Then, a simple algebraic computation shows that :

$$q_i = \frac{c_2 - \Delta\left(\frac{i-1}{N}\right)}{(1 - c_2) + \Delta\left(\frac{i}{N}\right)} \quad < 1 \qquad \text{for } 1 \leq i < i_0$$

$$> 1 \qquad \text{for } i_0 < i \leq N.$$

It is important to observe that for large N, $i_0 < qN$, where q is strictly less than 0.5.

The ε-optimality of the scheme when $c_2 = 0.5$ is disposed of by remarking that q_i increases to $(1-c_1+ \Delta/N) / c_1$ and is strictly greater than unity for all i.

Consider now the case when $c_2 < 0.5$. In this case, $q_i < 1$ and increases for $1 \le i < i_0$, and continues to increase beyond i_0. We compare π_i for $0 \le i < i_0$ and $i_0 < i < 2i_0 + 2$ to π_i for i in the interval $2i_0 + 2 < i \le N$. In the latter interval,

$$q_i > \frac{1 - c_2}{c_2} > 1.$$

Let i_1 be the first integer in the interval $(2i_0 + 2, N]$. Then the probability measure in the first two intervals sum to a quantity **less** than $2qN\pi_1$, where q is chosen strictly less than 0.5 such that $i < qN$ for sufficiently large N and q independent of N. Similarly, the probability measure in the last interval sums to a quantity **more** than S', where,

$$S' = \frac{1 - (\frac{1 - c_2}{c_2})^{(1-2q)N}}{1 - \frac{1 - c_2}{c_2}} \cdot \pi_i \qquad (14)$$

This shows that for $N \rightarrow \infty$ most of the probability mass sits in the last interval, and an argument as in case (i) finishes the proof. Between cases I and II we see that the scheme is ε-optimal whenever $c_1 < 0.5$.

Case III : $c_2 > c_1 \ge 0.5$.

Let α be defined as $\alpha = (2c_2 - 1)/2(c_2 + c_1 - 1)$. Of course, α is strictly greater than 0.5 in the case we are considering. Let $d > 0$ be an arbitrarily small positive number. For $i = 0,\ldots, N$, let i_1 be the first of the numbers i/N which belong to the interval from α-d to α, and let $\pi(i_1)$ be the corresponding associated probability measure. Since the probabilities are increasing in the interval from 0 to i_1 the probability of the whole interval from 0 to α-d is bounded **above** by $i_1 . \pi(i_1)$. We shall show that the probability of the interval from α-d to α-d/2 is bounded **below** by $\pi(i_1)$ times the sum of a quotient series where the quotient is bounded **below** by $d(c_1+c_2-1)+1$ independent of N (if d is sufficiently small and N is large enough). But the number of terms in that quotient series is asymptotic to (1/2d) N since each of the numbers in the intersection of the progression i/N with the interval from α-d to α - d/2 contributes one term. Hence, as N tends to infinity, most of the probability mass in the interval from 0 to α sits in the interval from α-d to α. A very similar argument gives that most of the probability mass in the interval from α to 1 sits in the interval from α to α+d, where d is arbitrarily small. This concludes the proof.

Hence the automaton is not ε-optimal whenever the minimum penalty probability is greater than 0.5. Interestingly enough the value of $E[p_1(\infty)] = 1$ when $c_1 = 0.5$.

Hence the theorem ! ●●●

Remarks.

1. The question of whether the DL_{RP} automaton was ε-optimal was left open in [44], but Oommen conjectured that the machine was ε-optimal in all environments. As is obvious from the above, the latter conjecture is false.

2. When Tsetlin first designed the Tsetlin automaton, $L_{2N,2}$ (or linear tactic), his automaton was the first (deterministic or stochastic) automaton that could be proven to possess learning properties. The automaton was shown to be ε-optimal in environments whenever the minimum penalty probability is less than 0.5. It is not inappropriate to mention that the DL_{RP} automaton is **not** a generalized version of the linear tactic, but is distinct in both design and operation for the following reasons :

(a) Whereas the $L_{2N,2}$ automaton is a FSSA, the DL_{RP} scheme is a VSSA.

(b) In the case of the $L_{2N,2}$ automaton, the action probability vector is a **deterministic** vector. In the case of the DL_{RP} scheme, **p**(n) is a random vector. Thus, whenever $c_1 < 0.5$, whereas in the former case the **probability** $p_1(\infty) \rightarrow 1$ as $N \rightarrow \infty$, in the latter case the **expected** probability $E\ [p_1(\infty)] \rightarrow 1$ as $N \rightarrow \infty$. Thus all the advantages of VSSA over FSSA (such as that of possessing the capability of choosing different actions at almost all consecutive time instances) are found in the DL_{RP} scheme. Additionally, the expected penalty tends to the value of the minimum penalty probability whenever the latter quantity is less than 0.5.

(c) When interacting with non-stationary environments, we can show that the DL_{RP} scheme is superior to the $L_{2N,2}$ automaton.

Elsewhere [46], modified absorbing and ergodic versions of the DL_{RP} automaton which are ε-optimal in **all** random environments have also been presented, but they are not of direct relevance to our particular problem, and so are not described here.

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, the use of these automata in motion planning will be presented.

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. Clearly, for all n,

$$\sum_{i=1}^{R} p_i(n) = 1 \qquad (15)$$

For the ease of notation, as before, we omit the reference to the time instant n.

Apart from (15), we constrain each p_i such that it has to be a value on the discretized space. Let $\delta = 1/NR$. Then, apart from (15), p_i satisfies :

$$p_i \in \{0, \delta, 2\delta, \ldots, (NR-1)\delta, 1\}$$

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

$$\begin{aligned} p_j(n+1) &= \max(p_j(n) - \delta, 0) && \text{if } a(n) = a_j,\ b(n) = 0 \\ &= 1 - \textstyle\sum_{j\neq i} \max(p_j(n) - \delta, 0) && \text{if } a(n) = a_i,\ b(n) = 0 \end{aligned} \qquad (16)$$

The philosophy behind (16) is quite straightforward. If a_j 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 random subset $H_k(j)$ of k indices from the set $\{1,2,\ldots,R\} - \{j\}$. Using this notation, we define the updating rule as below:

$$
\begin{aligned}
p_j(n+1) &= \max(p_j(n) - (R-1)\delta, 0) && \text{if } a(n) = a_j,\ b(n) = 1 \\
&= p_i(n) + \delta && \text{if } a(n) = a_j,\ b(n) = 1,\ p_j(n) = k\delta, \\
& && \text{where } k \geq (R-1) \\
&= p_i(n) + \delta && \text{if } a(n) = a_j,\ b(n) = 1,\ p_j(n) = k\delta, \\
& && \text{where } k < R-1 \text{ and } i \in H_k(j) \qquad (17)
\end{aligned}
$$

The philosophy motivating (17) is also quite straightforward. If a_i is chosen 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, distributing the probability mass of δ for each action probability. However, if the action probability of the action chosen has only a value of $k\delta$, where $k<R-1$, this probability is decremented to zero, and k out of the $(R-1)$ remaining action probabilities are randomly chosen and each incremented by δ.

We now prove that the asymptotic properties of the R-state DL_{RP} automaton.

Theorem III

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

Proof:

Consider the R-state DL_{RP} automaton with $N = 0$. In this case, the vector $\mathbf{p}(n)$ is a unit vector $\mathbf{e}_i$, where $\mathbf{e}_i$ is the vector with zeros and a unity in the ith position. By definition if $\mathbf{p}(n) = \mathbf{e}_i$, the action chosen must be a_i. Thus, using (16) and (17), the probability updating rule can be written in the vector form as (18) below for $j \neq i$.

$$
\begin{aligned}
\mathbf{p}(n+1) &= \mathbf{e}_i && \text{if } \mathbf{p}(n) = \mathbf{e}_i,\ b(n) = 0 \\
&= \mathbf{e}_j && \text{with prob. } (1/R-1) \text{ if } \mathbf{p}(n) = \mathbf{e}_i,\ b(n) = 1, \qquad (18)
\end{aligned}
$$

Consider the Markov chain $\aleph$ defined by the states $\{i|\ 1 \leq i|R\}$, where $\aleph = i$ if a_i is the **action** chosen by the automaton. If A is the stochastic matrix defining the chain it can be seen that by virtue of (18), that A matrix has the form:

$$A = \begin{bmatrix} 1-c_1 & c_1/R-1 & c_1/R-1 & \cdots & c_1/R-1 \\ c_2/R-1 & 1-c_2 & c_2/R-1 & & c_2/R-1 \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ c_R/R-1 & c_R/R-1 & c_R/R-1 & \cdots & 1-c_R \end{bmatrix}$$

(19)

Let $\pi = [\pi_1, \pi_2, \ldots, \pi_R]^T$ be the limiting aymptotic probability vector of the ergodic Markov chain described by (19). Clearly, π is obtained by solving

$$\pi = A^T \pi$$

π is thus the eigenvector of the eigenvalue of A which is unity.

Solving $[I - A]^T \pi = 0$ yields for the first row,

$$\{ c_1 \pi_1 - (1/(R-1)) \sum_{i=2}^{R} c_i \pi_i \} = 0$$

whence $\pi_1 = J / Rc_1$, where J is a constant independent of 'i' and is equal to

$$J = \sum_{j=1}^{R} c_j \pi_j \ .$$

Similarly, $\pi_i = J / R\, c_i$, where J is defined above.

Since, $\sum_{i=1}^{R} \pi_i$ is unity, J can be solved for to have the value :

$$J = \sum_{j=1}^{R} 1/c_j$$

whence, $\pi_i = (1/c_i) / \sum_{j=1}^{R} (1/c_j)$.

Since π_i is the limiting probability of being in state i, which is indeed the limiting probability of the automaton choosing a_i, the limiting expected penalty is $M(\infty)$, where,

$$M(\infty) = \sum_{j=1}^{R} \pi_i c_i = R / \sum_{j=1}^{R} 1/c_j$$

Indeed, $M(\infty)$ is the harmonic mean of $\{c_i\}$.The expedience is proved from the above by observing that the harmonic mean of a sequence of numbers is never greater than that arithmetic mean.

●●●

Remark

1. It can be seen that the above **special case** of the R-Action DL_{RP} automaton obtained by setting N=0, is itself a very general (stochastic) version of Tsetlin's $L_{R,R}$ automaton [49,50]. Of course, when $N \geq 1$, the resemblence between the two families of automata disappears because Tsetlin's automaton is a FSSA, and the DL_{RP} automaton is, by definition, a VSSA.

2. 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. In the case when R = 3, the probability vector can be drawn graphically as a point in the plane inside of an equilateral triangle. The height of the triangle is unity, and the vertices represent the unit vectors (1,0,0), (0,1,0) and (0,0,1). A point interior to the triangle represents the vector $[p_1, p_2, p_3]^T$, where the quantity p_i is the length of the perpendicular from the point to the ith side of the triangle.

When N=1, (i.e., when the probability space is divided into three equal intervals), there are 10 permissible probability vectors. These vectors are represented by the set of tuples (i,j,k) given below:

$$\{(i,j,k) \mid i+j+k = 3 \; ; i,j,k \geq 0\}$$

Note that the action probability vector reprented by the tuple (i,j,k) is the vector $[i/3, j/3, k/3]^T$. The physical representation of these tuples on the equilateral triangle described above and the transition matrix describing the Markov chain defined by the Three-Action DL_{RP} automaton are given below in Figure II and Table I respectively.

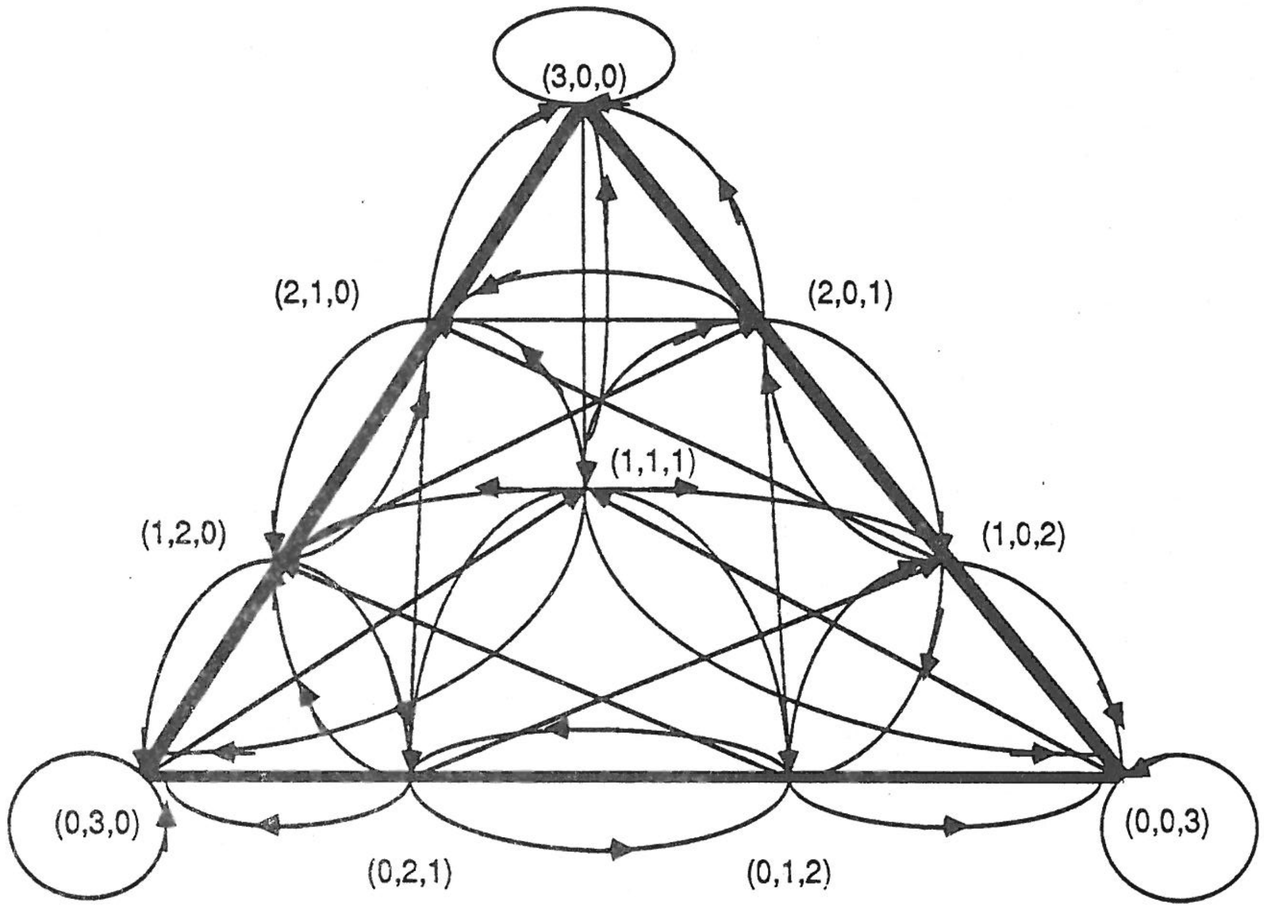

Figure II : The transition map of the DL_{RP} automaton for the case when R=3 and N=1.

$$
\begin{array}{c|cccccccccc}
 & (300) & (030) & (003) & (210) & (120) & (201) & (102) & (021) & (012) & (111) \\
\hline
(300) & d1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c1 \\
(030) & 0 & d2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & c2 \\
(003) & 0 & 0 & d3 & 0 & 0 & 0 & 0 & 0 & 0 & c3 \\
(210) & c2/6+2d1/3 & 0 & 0 & 0 & d2/3 & c2/6 & 0 & 2c1/3 & 0 & 0 \\
(120) & 0 & c1/6+2d2/3 & 0 & d1/3 & 0 & 2c2/3 & 0 & c1/6 & 0 & 0 \\
(201) & c3/6+2d1/3 & 0 & 0 & c3/6 & 0 & 0 & d3/3 & 0 & 2c1/3 & 0 \\
(102) & 0 & 0 & c1/6+2d3/3 & 2c3/3 & 0 & d1/3 & 0 & 0 & c1/6 & 0 \\
(021) & 0 & c3/6+2d2/3 & 0 & 0 & c3/6 & 0 & 2c2/3 & 0 & d3/3 & 0 \\
(012) & 0 & 0 & c2/6+2d3/3 & 0 & 2c3/3 & 0 & c2/6 & d2/3 & 0 & 0 \\
(111) & d1/3 & d2/3 & d3/3 & c3/6 & c3/6 & c2/6 & c2/6 & c1/6 & c1/6 & 0
\end{array}
$$

Table I : The transition matrix of the DL_{RP} automaton for the case when R=3 and N=1. The transition map of the automaton is given in Figure II.

The asympotic efficiency of the automaton can be studied by computing the stationary probabilities of A and computing the net average penalty. This has been done for various penalty probabilities $[c_1, c_2, c_3]$. In all cases the performance of the machine is superior to that of the corresponding machine obtained for N=0.

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 $\mathbf{Q}_f(n)$. Unfortunately, $\mathbf{Q}_f(n)$ 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 control the manipulator so that it reaches arbitrarily close to $\mathbf{P}_f$. Furthermore, we intend to achieve the task adaptively.

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 updated 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.

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. This is quite appealing.

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. We shall discuss the actual experimental significances of these issues in the next section.

V EXPERIMENTAL RESULTS

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 R_2 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. In the figures that are included (Figures III to VIII) circles which represent perturbations of up to three standard deviations of the noise are presented to show the degree of noise introduced.

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 III(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 III (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. The decrease of the number of iterations with the increasing number of states is shown in Figure IV. Notice that in this case, the path planning is acheived in a few as 45 iterations for the case when NR = 8. Similar graphs are available for the case when R=3. On Figure V (a) we have shown the expected trajectory taken when NR = 9, and the decrease in the Euclidian distance is shown in Figure V(b). Again, the rate of convergence of the scheme as a function of NR, the number of states is shown in Figure VI. The power of the scheme is obvious - since the expected trajectory enters the three standard deviation disk in only 75 iterations for the case when NR = 12.

In the case when each automaton had three actions (R=3) and the automata received distinct responses, the responses to the automata were obtained by computing the linearized versions of the inverse kinematics. These were obtained at any time instant by solving linear equations involving the "recent history" (i.e., last few Cartesian and joint space coordinates) of the end effector. The performance of the automaton is shown in Figure VII (a) for NR = 9. The corresponding decrease in the Euclidian distance from $\mathbf{P}_f$ as a function of n is shown in Figure VII (b). Amazingly enough, the different responses do not improve the collective learning capability of the automata, but actually degrade it. (cf. Figure V(b) and Figure VII (b)).

Finally to study the effect of the scheme for a more general robot, we have investigated its performance for the case of the three-link generalization of Horn's Robot. A variety of results have been obtained. Typically, the convergence is shown for the case when NR= 9 and R= 3 in Figure VIII(a), and the corresponding decrease in the Euclidian distance is shown in Figure VIII (b).

A question may be asked regarding the need for using the DL_{RP} automaton in this application. It has been well-known that the updating function of a learning automaton must be dependent on the response it receives from the environment. For example, consider a continuous VSSA which **completely ignores** the penalty responses of the environment. Such an automaton is of the Reward-Inaction type, and it is well known that there are linear and nonlinear Reward-Inaction schemes which are both absolutely expedient and ε-optimal. Apart from the continuous schemes, indeed as shown in Section IV of [44], even discretized ε-optimal schemes of the Reward-Inaction flavour do exist. However, although the linear symmetric Reward-Penalty schemes are at their best expedient (and definitely **not** absolutely expedient[33]) Reward-Penalty schemes need not be entirely rejected. In this paper, we have shown that by discretizing the probability space the resulting symmetric automaton is indeed ε-optimal wherever $c_{min} < 0.5$. Furthermore, apart from being ε-optimal, the automaton utilizes **all** the information provided by the environment, and thus does not ignore any response as a Reward-Inaction automaton would.

The power of using the scheme suggested in this paper is that the sensing mechanism used need not be all too sophisticated. A primitive sensor with such a learning strategy could indeed provide excellent results.

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 pernalty 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.

The following are the future research possibilities that we envisage:

(i) We intend to actually study the power of the scheme for a real life robot which has very primitive sensing operations.

(ii) 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.

(iii) We have also recently discovered two new families of discretized Reward-Penalty automata which are ε-optimal in **all** random environments. We hope to study the use of these automata to achieve the path planning more effectively.

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.

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. Brooks, R.A., and Lozano-Perez, T., "A Subdivision Algorithm in Configuration Space for Findpath with Rotation", IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-15, March/April 1985, pp.224-233.
6. Canny, J.F., " Collision Detection for Moving Polyhedra", A.I. memo 806. Oct. 1984. M.I.T. Artificial Intelligence Laboratory.
7. Chatilla, R., "Path Planning and Environment Learning in a Mobile Robot System", Proc. European Conf. Artificial Intelligence, Torsay, France, 1982.
8. Chattergy, R.,"Some Heuristics for the Navigation of a Robot", The Int. Journal of Robotics Research, Vol. 4, Spring 1985, pp.59-66.
9. 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.
10. Crowley, J.L.,"Navigation of an Intelligent Mobile Robot", IEEE J. Robotics and Automation, Vol. RA-1, March 1985, pp.31-41.
11. Giralt, G., Sobek, R. and Chatila, R., "A Multilevel Planning and Navigation System for a mobile Robot", Proc. 6th Int. Joint Conf. Artificial Intelligence, 1979,Tokyo, pp335-338.
12. Gouzenes, L.," Strategies for Solving Collision-Free Trajectoriy Problems for Mobile and Manipulator Robots ",The Int. Journal of Robotics Research, Vol. 3, Winter 1984, pp.51 - 65.
13. 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.
14. 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.
15. Iyengar, S.S., Jorgensen, C.C. , Rao, S.V.N. , and Weisbin , C.R., " Robot Navigation Algorithms Using Learned Spatial Graphs", ORNL Technical Report TM-9782,Oak Ridge Nat. Lab.,Oak Ridge, August 1985.
16. Iyengar, S.S., Jorgensen, C.C., Rao, S.V.N. and Weisbin, C.R., " Learned Navigation Paths for a Robot in Unexplored Terrain ", Proc. 2nd Conf. Artificial

Intelligence Applications and Engineering of Knowledge Based Systems, Miami Beach , Florida, 1985, pp.148-155.

17. Kant, K. and Zucker, S.W. , " Trajectory Planning in Time-Varying Environments,1: TPP = PPP + VPP", TR-84-7R, McGill Univ.,Montreal,Computer Vision and Robotics Lab.,1984.

18. 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.

19. 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.

20. Laumond, J., " Model Structuring and Concept Recognition : Two Aspects of Learning for a Mobile Robot ", Proc. 8th Conf. Artificial Intelligence, 1983, Karlsruhe, West Germany, pp. 839.

21. Lozano-Perez, T., "Spatial Planning: A Configuration Space Approach", IEEE Trans. Computers, Vol. C-32 , Feb. 1983, pp 108-120.

22. Lozano -Perez, T. and Wesley, M.A. , "An Algorithm for Planning Collision-free Paths among Polyhedral Obstacles ", Communications ACM , Vol. 22 , No. 10 , Oct. 1979, pp. 560-570.

23. 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.

24. Oommen, B.J., Iyengar, S.S., Rao, N.S.V. and Kashyap, R.L.,"Robot Navigation in Unknown Terrains Using Learned Visibility Graphs. Part I : The Disjoint Convex Obstacle Case". To appear in the IEEE Trans. on Robotics and Automation. Also available as SCS-TR-86, School of Computer Science, Carleton University, Ottawa, Canada.

25. Rao, S.V.N. ,Iyengar, S.S.,Jorgensen, C.C. and Weisbin, C.R., " Concurrent Algorithms for Autonomous Robot Navigation in an Unexplored Terrain ", Tech. Rep. 85-048, Dept. Computer Science,Louisiana State University , 1985.

26. Schwartz, J.T., and Sharir, M., "On the Piano Movers' Problem : III. Coordinating the Motion of Several Independent Bodies : The Special Case of Circular Bodies Amidst Polygonal Barriers", Int. Journal of Robotics Research, Vol. 2, 1983, pp.46-75.

27. Kozlov, A., "Robotics and Model Share Innovations", SIAM News, Vol. 18, No. 5, Sept. 1985, pp. 1,11.

28. Togai, M., and Yamano, O.,"learning Control and its Optimality", Proc. of the IEEE Robotics and Automation Conference, April 1986, pp. 248-253.

29. Turchen, M.P. and Wong, A.K.C. , "Low Level Learning for a Mobile Robot : Environmental Model Acquisition ", Proc. 2nd Int. Conf. Artificial Intelligence Applications, 1985, pp.156-161.

30. Udupa, S.M., "Collision Detection and Avoidance in Computer Controlled Manipulators ", Proc. 5th Int. Conf. Artificial Intelligence,M.I.T.,Cambridge,Mass.,Aug. 1977, pp. 737-748.

31. Whitesides,S., "Computational Geometry and Motion Planning", Computational Geometry, Ed. by G. Toussaint, North Holland, 1985.

32. Ross, S.M., "Introduction to Probability Models", Academic Pres, New York, 1980.
33. Lakshmivarahan, S., "Learning Algorithms Theory and Applications", Springer-Verlag, New York, 1981.
34. Lakshmivarahan, S., " ε-Optimal Learning Algorithms - Non-Absorbing Barrier Type", Technical Report EECS 7901, February 1979, School of Electrical Engineering and Computing Sciences, University of Oklahoma, Norman, Oklahoma.
35. Lakshmivarahan, S., "Two Person Decentralized Team With Incomplete Information", Applied Mathematics and Computation, Vol. 8, pp.51-78, 1981.
36. 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.
37. Meybodi, M.R., "Learning Automata and Its Application to Priority Assignment in a Queueing System With Unknown Characteristics", Ph.D. Thesis, School of Electrical Engineering and Computing Sciences, University of Oklahoma, Norman, Oklahoma.
38. Narendra, K.S., and Thathachar, M.A.L., Forthcoming book on Learning Automata.
39. 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.
40. Narendra, K.S., and Thathachar, M.A.L., "On the Behaviour of a Learning Automaton in a Changing Environment With Routing Applications", IEEE Trans. on Syst. Man and Cybern., Vol. SMC-10, 1980, pp.262-269.
41. Narendra, K.S., Wright, E., and Mason, L.G., "Applications of Learning Automata to Telephone Traffic Routing", IEEE Trans. on Syst. Man and Cybern., Vol. SMC-7, 1977, pp.785-792.
42. Oommen, B.J., and Hansen, E.R., "The Asymptotic Optimality of Discretized Linear Reward-Inaction Learning Automata", IEEE Trans. on Syst. Man and Cybern., May/June 1984, pp.542-545.
43. Oommen, B.J., and Thathachar, M.A.L., "Multi-Action Learning Automata Possessing Ergodicity of the Mean", Proc. of the 1983 IASTED Symposium on Measurement and Control, MECO 83, pp.61-64.
44. 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.
45. Oommen, B.J., "A Learning Automaton Solution to the Stochastic Minimum Spanning Circle Problem", IEEE Trans. on Syst. Man and Cybern., July/Aug. 1986, pp.598-603.
46. Oommen, B.J., and Christensen, J.P.R., "ε-Optimal Discretized Linear Reward-Penalty Automata Learning Automata", Submited for Publication.
47. Oommen, B.J., and Ma, D.C.Y., "Fast Object Partitioning Using Stochastic Learning Automata". Proceedings of the 1987 International Conference on Research and Development in Information Retrieval, New Orleans, June 1987.
48. Thathachar, M.A.L., and Oommen, B.J., "Discretized Reward-Inaction Learning Automata", Journal of Cybernetics and Information Sciences, Spring 1979, pp.24-29.
49. Tsetlin, M.L., "On the Behaviour of Finite Automata in Random Media", Automat. Telemekh.(USSR), Vol.22, 1961, pp.1345-1354.
50. Tsetlin, M.L., "Automaton Theory and the Modelling of Biological Systems",

50. Tsetlin, M.L., "Automaton Theory and the Modelling of Biological Systems", New York and London, Academic, 1973.
51. Tsypkin, Y.Z. and Poznyak, A.S., "Finite Learning Automata", Engineering Cybernetics, Vol.10, 1972, pp.478-490.
52. 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.

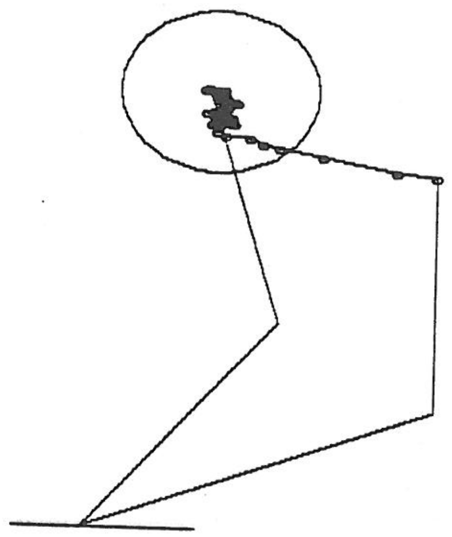

FIGURE : IIIa

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 DLrp 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.

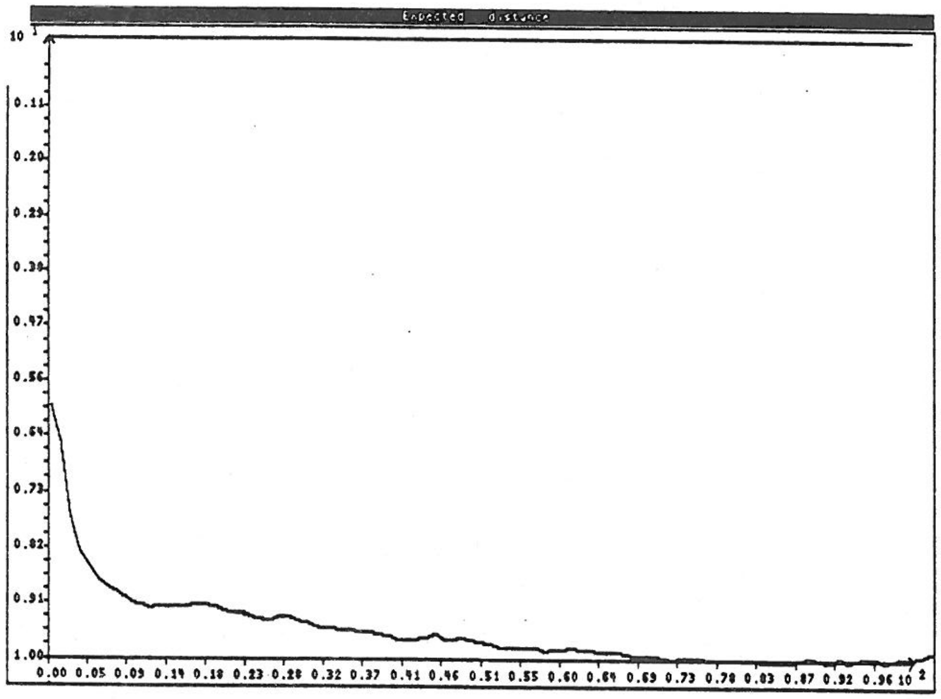

FIGURE : IIIb
Expected change in distance from the initial configuration to the final mean configuration for the set-up described in Figure : IIIa

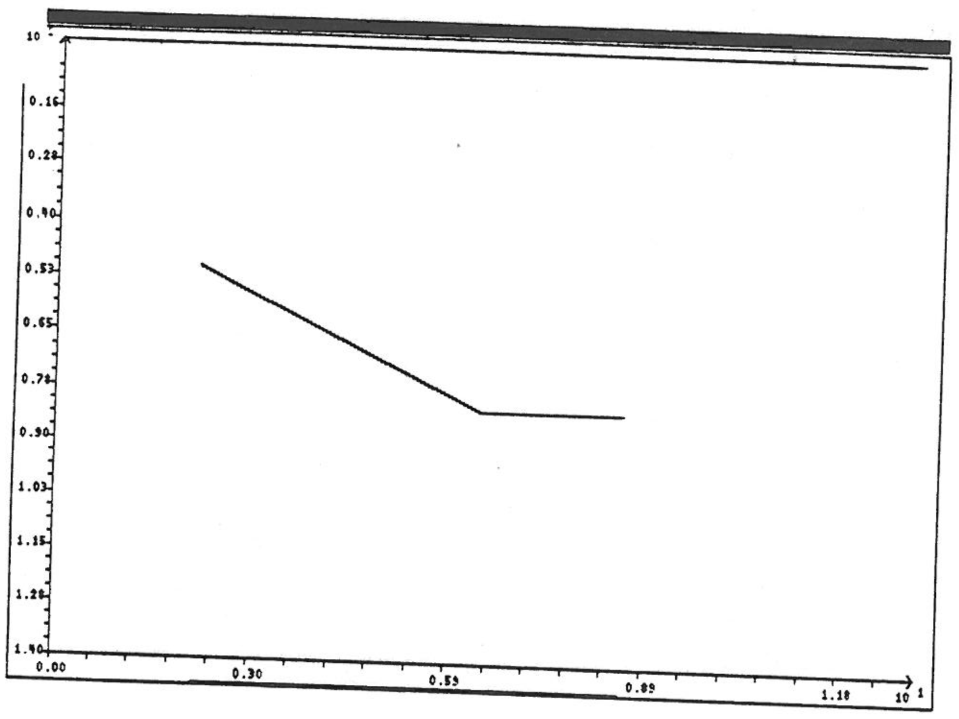

FIGURE : IV
Expected number of iteractions for the robots to enter the three standard deviation disk of the final point for cases when the automata use a single response.

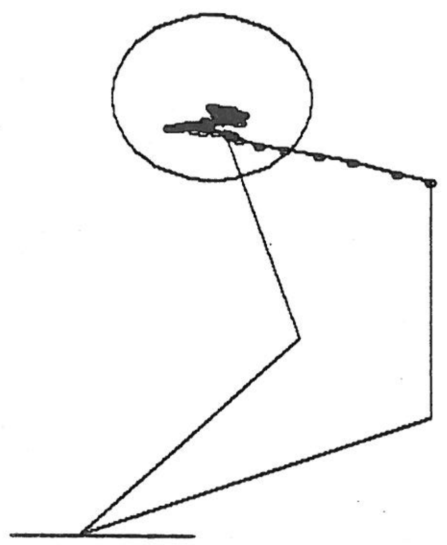

FIGURE : Va

Expected Path (given as a sequence of points) of the two-link Horn's Rebot with links of length unity. The control is achieved using the DLrp automata with NR = 9 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.

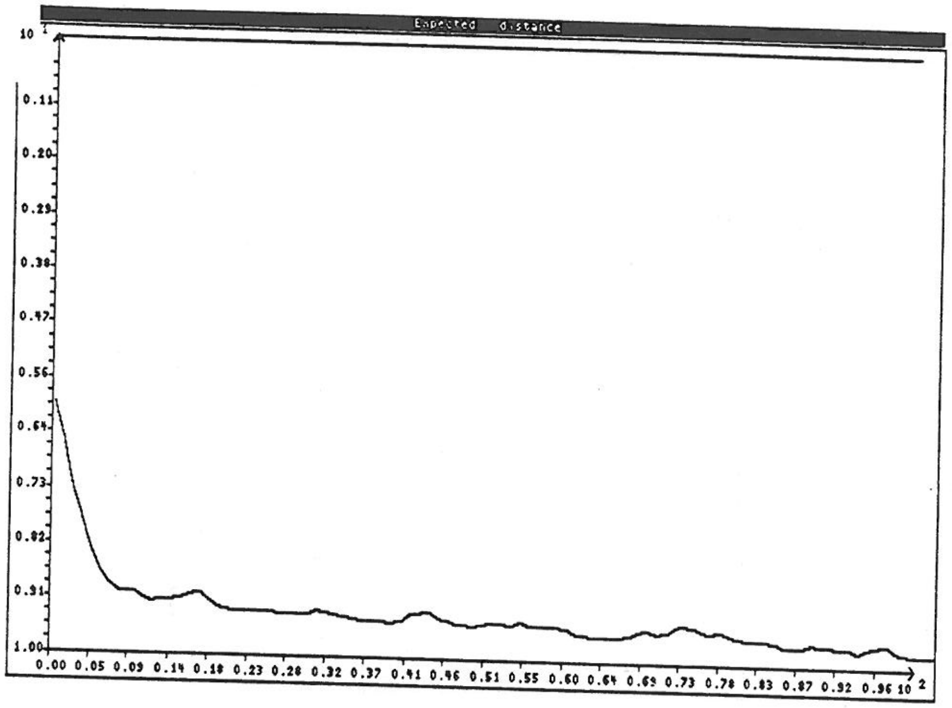

FIGURE : Vb
Expected change in distance from the initial configuration to the final mean configuration for the set-up described in Figure : Va

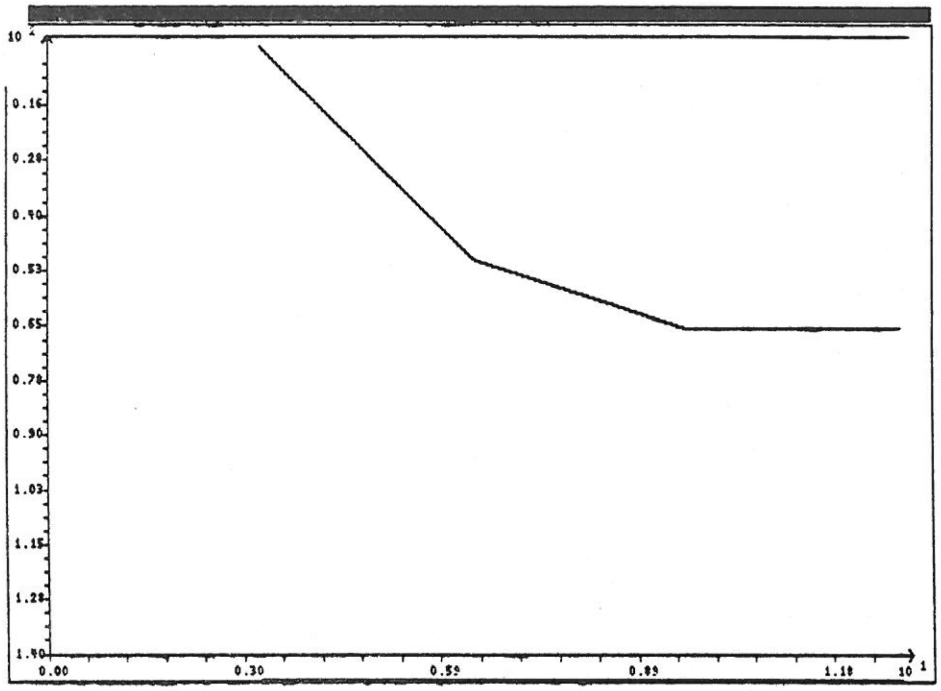

FIGURE : VI
Expected number of iteractions for the robots to enter the three standard deviation disk of the final point for cases when the automata use a single response.

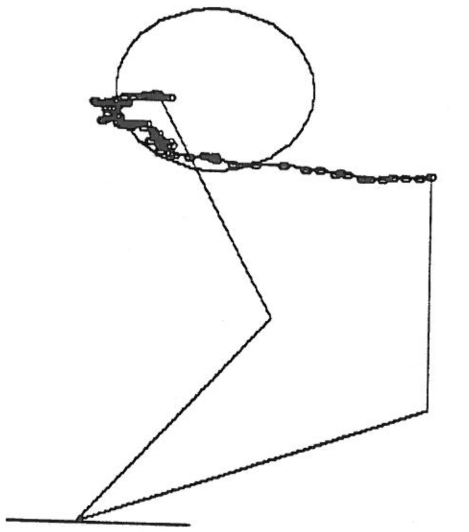

FIGURE : VIIa

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 DLrp automata with NR = 9 and with a separate response for each automata, where these responses are obtained using a first order approximation of inverse kinematics. In this case the standard deviation is 0.1 times the link length. The disk shows the three standard deviation range of noisy points.

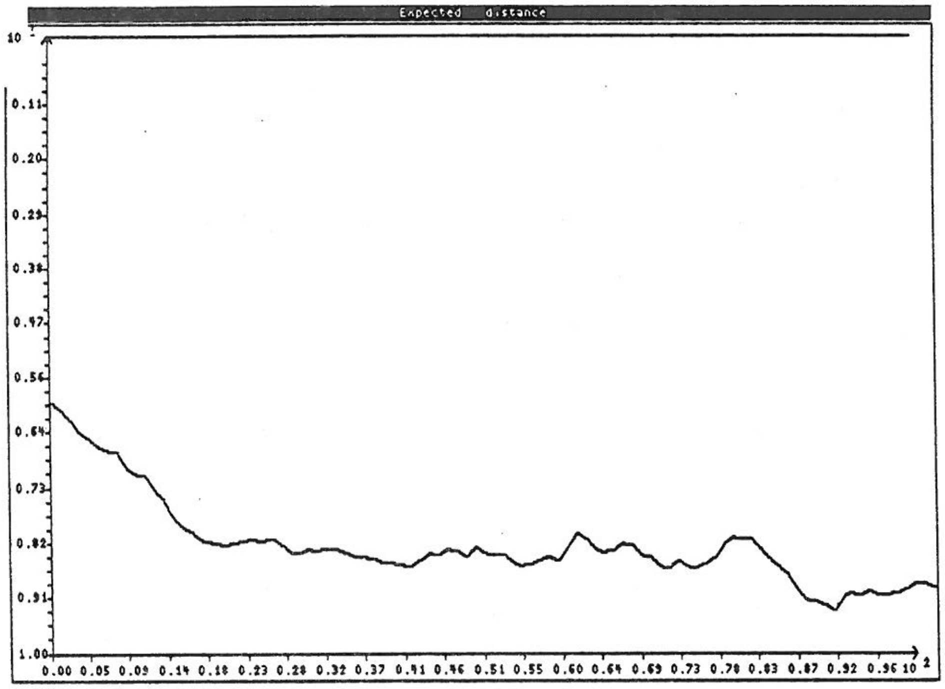

FIGURE : VIIb
Expected change in distance from the initial configuration to the final mean configuration for the set-up described in Figure : VIIa

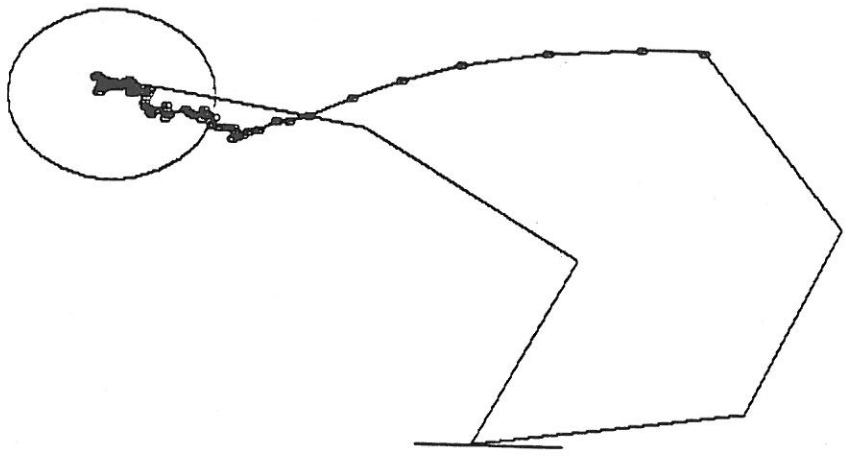

FIGURE : VIIIa

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 DLrp automata with NR = 9 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.

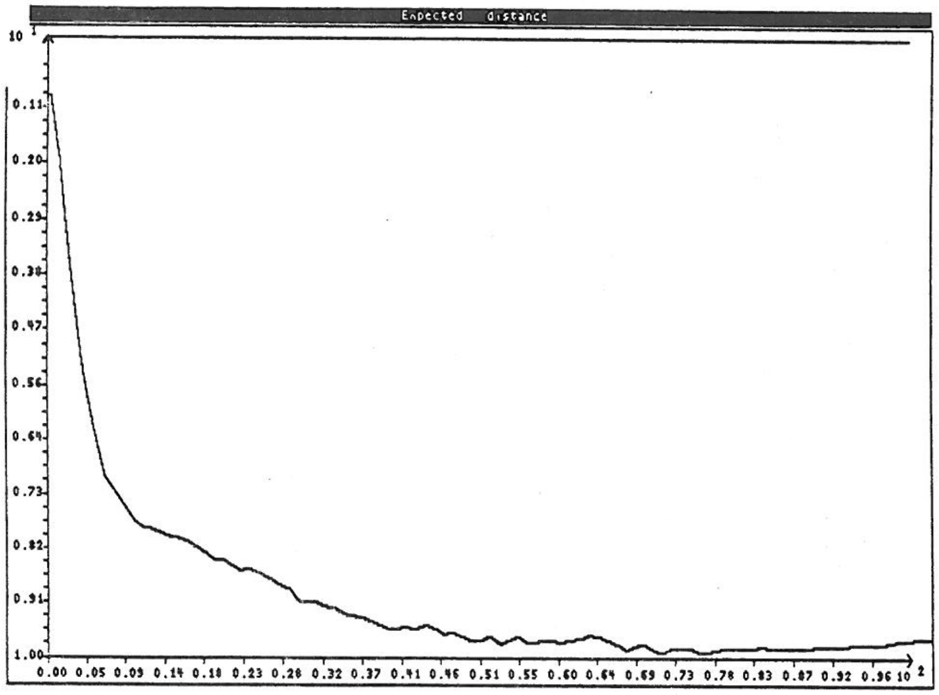

FIGURE : VIIIb
Expected change in distance from the initial configuration to the final mean configuration for the set-up described in Figure : VIIIa

Carleton University, School of Computer Science
Bibliography of Technical Reports
Publications List (1985 -->)

School of Computer Science
Carleton University
Ottawa, Ontario, Canada
KIS 5B6

SCS-TR-66	**On the Futility of Arbitrarily Increasing Memory Capabilities of Stochastic Learning Automata**
————	B.J. Oommen, October 1984. Revised May 1985.
SCS-TR-67	**Heaps in Heaps**
————	T. Strothotte, J.-R. Sack, November 1984. Revised April 1985.
SCS-TR-68	**Partial Orders and Comparison Problems**
out-of-print	M.D. Atkinson, November 1984. See Congressus Numerantium 47 ('86), 77-88
SCS-TR-69	**On the Expected Communication Complexity of Distributed Selection**
————	N. Santoro, J.B. Sidney, S.J. Sidney, February 1985.
SCS-TR-70	**Features of Fifth Generation Languages: A Panoramic View**
————	Wilf R. LaLonde, John R. Pugh, March 1985.
SCS-TR-71	**Actra: The Design of an Industrial Fifth Generation Smalltalk System**
————	David A. Thomas, Wilf R. LaLonde, April 1985.
SCS-TR-72	**Minmaxheaps, Orderstatisticstrees and their Application to the Coursemarks Problem**
————	M.D. Atkinson, J.-R. Sack, N. Santoro, T. Strothotte, March 1985.
SCS-TR-73	**Designing Communities of Data Types**
	Wilf R. LaLonde, May 1985.
Replaced by SCS-TR-108	
SCS-TR-74	**Absorbing and Ergodic Discretized Two Action Learning Automata**
out-of-print	B. John Oommen, May 1985. See IEEE Trans. on Systems, Man and Cybernetics, March/April 1986, pp. 282-293.
SCS-TR-75	**Optimal Parallel Merging Without Memory Conflicts**
————	Selim Akl and Nicola Santoro, May 1985
SCS-TR-76	**List Organizing Strategies Using Stochastic Move-to-Front and Stochastic Move-to-Rear Operations**
————	B. John Oommen, May 1985.
SCS-TR-77	**Linearizing the Directory Growth in Order Preserving Extendible Hashing**
————	E.J. Otoo, July 1985.
SCS-TR-78	**Improving Semijoin Evaluation in Distributed Query Processing**
————	E.J. Otoo, N. Santoro, D. Rotem, July 1985.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-79	**On the Problem of Translating an Elliptic Object Through a Workspace of Elliptic Obstacles** B.J. Oommen, I. Reichstein, July 1985.
SCS-TR-80	**Smalltalk - Discovering the System** W. LaLonde, J. Pugh, D. Thomas, October l985.
SCS-TR-81	**A Learning Automation Solution to the Stochastic Minimum Spanning Circle Problem** B.J. Oommen, October 1985.
SCS-TR-82	**Separability of Sets of Polygons** Frank Dehne, Jörg-R. Sack, October 1985.
SCS-TR-83 out-of-print	**Extensions of Partial Orders of Bounded Width** M.D. Atkinson and H.W. Chang, November 1985. See Congressus Numerantium, Vol. 52 (May 1986), pp. 21-35.
SCS-TR-84	**Deterministic Learning Automata Solutions to the Object Partitioning Problem** B. John Oommen, D.C.Y. Ma, November 1985
SCS-TR-85 out-of-print	**Selecting Subsets of the Correct Density** M.D. Atkinson, December 1985. To appear in Congressus Numerantium, Proceedings of the 1986 South-Eastern conference on Graph theory, combinatorics and Computing.
SCS-TR-86	**Robot Navigation in Unknown Terrains Using Learned Visiblity Graphs. Part I: The Disjoint Convex Obstacles Case** B. J. Oommen, S.S. Iyengar, S.V.N. Rao, R.L. Kashyap, February 1986
SCS-TR-87	**Breaking Symmetry in Synchronous Networks** Greg N. Frederickson, Nicola Santoro, April 1986
SCS-TR-88	**Data Structures and Data Types: An Object-Oriented Approach** John R. Pugh, Wilf R. LaLonde and David A. Thomas, April 1986
SCS-TR-89	**Ergodic Learning Automata Capable of Incorporating Apriori Information** B. J. Oommen, May 1986
SCS-TR-90	**Iterative Decomposition of Digital Systems and Its Applications** Vaclav Dvorak, May 1986.
SCS-TR-91	**Actors in a Smalltalk Multiprocessor: A Case for Limited Parallelism** Wilf R. LaLonde, Dave A. Thomas and John R. Pugh, May 1986
SCS-TR-92	**ACTRA - A Multitasking/Multiprocessing Smalltalk** David A. Thomas, Wilf R. LaLonde, and John R. Pugh, May 1986
SCS-TR-93	**Why Exemplars are Better Than Classes** Wilf R. LaLonde, May 1986
SCS-TR-94	**An Exemplar Based Smalltalk** Wilf R. LaLonde, Dave A. Thomas and John R. Pugh, May 1986
SCS-TR-95	**Recognition of Noisy Subsequences Using Constrained Edit Distances** B. John Oommen, June 1986

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-96	**Guessing Games and Distributed Computations in Synchronous Networks** J. van Leeuwen, N. Santoro, J. Urrutia and S. Zaks, June 1986.
SCS-TR-97	**Bit vs. Time Tradeoffs for Distributed Elections in Synchronous Rings** M. Overmars and N. Santoro, June 1986.
SCS-TR-98	**Reduction Techniques for Distributed Selection** N. Santoro and E. Suen, June 1986.
SCS-TR-99	**A Note on Lower Bounds for Min-Max Heaps** A. Hasham and J.-R. Sack, June 1986.
SCS-TR-100	**Sums of Lexicographically Ordered Sets** M.D. Atkinson, A. Negro, and N. Santoro, May 1987.
SCS-TR-102	**Computing on a Systolic Screen: Hulls, Contours, and Applications** F. Dehne, J.-R. Sack and N. Santoro, October 1986.
SCS-TR-103	**Stochastic Automata Solutions to the Object Partitioning Problem** B.J. Oommen and D.C.Y. Ma, November 1986.
SCS-TR-104	**Parallel Computational Geometry and Clustering Methods** F. Dehne, December 1986.
SCS-TR-105	**On Adding *Constraint Accumulation* to Prolog** Wilf R. LaLonde, January 1987.
SCS-TR-107	**On the Problem of Multiple Mobile Robots Cluttering a Workspace** B. J. Oommen and I. Reichstein, January 1987.
SCS-TR-108	**Designing Families of Data Types Using Exemplars** Wilf R. LaLonde, February 1987.
SCS-TR-109	**From Rings to Complete Graphs - Θ(n logn) to Θ(n) Distributed Leader Election** Hagit Attiya, Nicola Santoro and Shmuel Zaks, March 1987.
SCS-TR-110	**A Transputer Based Adaptable Pipeline** Anirban Basu, March 1987.
SCS-TR-111	**Impact of Prediction Accuracy on the Performance of a Pipeline Computer** Anirban Basu, March 1987.
SCS-TR-112	**ε-Optimal Discretized Linear Reward-Penalty Learning Automata** B.J. Oommen and J.P.R. Christensen, May 1987.
SCS-TR-113	**Angle Orders, Regular n-gon Orders and the Crossing Number of a Partial Order** N. Santoro and J. Urrutia, June 1987.
SCS-TR-115	**Time Is Not a Healer: Impossibility of Distributed Agreement in Synchronous Systems with Random Omissions** N. Santoro, June 1987.

Carleton University, School of Computer Science
Bibliography of Technical Reports

SCS-TR-116	**A Practical Algorithm for Boolean Matrix Multiplication** M.D. Atkinson and N. Santoro, June 1987.
SCS-TR-117	**Recognizing Polygons, or How to Spy** James A. Dean, Andrzej Lingas and Jörg-R. Sack, August 1987.
SCS-TR-118	**Stochastic Rendezvous Network Performance - Fast, First-Order Approximations** J.E. Neilson, C.M. Woodside, J.W. Miernik, D.C. Petriu, August 1987.
SCS-TR-120	**Searching on Alphanumeric Keys Using Local Balanced Trie Hashing** E.J. Otoo, August 1987.
SCS-TR-121	**An O(√n) Algorithm for the ECDF Searching Problem for Arbitrary Dimensions on a Mesh-of-Processors** Frank Dehne and Ivan Stojmenovic, October 1987.
SCS-TR-122	**An Optimal Algorithm for Computing the Voronoi Diagram on a Cone** Frank Dehne and Rolf Klein, November 1987.
SCS-TR-123	**Solving Visibility and Separability Problems on a Mesh-of-Processors** Frank Dehne, November 1987.
SCS-TR-124	**Deterministic Optimal and Expedient Move-to-Rear List Organizing Strategies** B.J. Oommen, E.R. Hansen and J.I. Munro, October 1987.
SCS-TR-125	**Trajectory Planning of Robot Manipulators in Noisy Workspaces Using Stochastic Automata** B.J. Oommen, S. Sitharam Iyengar and Nicte Andrade, October 1987.
SCS-TR-126	**Adaptive Structuring of Binary Search Trees Using Conditional Rotations** R.P. Cheetham, B.J. Oommen and D.T.H. Ng, October 1987.