

Neuro-Resistive Grid Approach to Trainable Controllers: A Pole Balancing Example

Raju S. Bapi, Brendan D'Cruz, and Guido Bugmann

Neurodynamics Research Group
School of Computing
University of Plymouth
Plymouth PL4 8AA
United Kingdom.

Phone: (+44) (0)1752 23 26 19

Fax: (+44) (0)1752 23 25 40

{rajubapi,brendan,gbugmann}@soc.plym.ac.uk

Acknowledgments: Grant (GR/J42151) support from the Engineering and Physical Sciences Research Council (EPSRC), UK to RSB (post doctoral research fellowship) and BDC (research studentship) is gratefully acknowledged.

Note: Accepted for publication in **Neural Computing & Applications**, 1996.

Abstract

A new neural network approach is described for the task of pole-balancing, considered a benchmark learning control problem. This approach combines Barto, Sutton, and Anderson's [1] Associative Search Element (ASE) with a Neuro-Resistive Grid (NRG) [2] acting as Adaptive Critic Element (ACE). The novel feature in NRG is that it provides evaluation of a state based on propagation of the failure information to the neighbours in the grid. NRG is updated only on a failure and provides ASE with a continuous internal reinforcement signal by comparing the value of the present state to the previous state. The resulting system learns more rapidly and with fewer computations than that of Barto et al [1]. To establish a uniform basis of comparison of algorithms for pole balancing, both the systems are simulated using benchmark parameters and tests specified in [3].

Keywords

Inverted pendulum problem, reinforcement learning, learning control, nonlinear control, neural networks, neuro-resistive grid method, value map

1. Introduction

Pole-balancing or balancing-an-inverted-pendulum has been identified as a benchmark problem for trainable controllers [3]. The task consists of balancing a pole, attached vertically to a movable cart, by applying one dimensional forces of constant magnitude to the base of the cart. The controller does not have access to the equations of motion of the system. The general problem here is to discover a sequence of binary (right or left) control forces that can keep the system balanced for long periods of time. To enable this discovery, the only information available to the learning control system is a negative reinforcement signal given when the system collapses. Hence the controller faces the problem of evaluating its

intermediate actions in the absence of any continuous external information. The strategy in these delayed reinforcement problems is to select the actions at every step that would reduce the possibility of eventual failure.

The system state space is determined by four variables, namely, the position of the cart on the rail (measured with reference to the centre of the railing), linear velocity of the cart, the angle of inclination of the pole (measured with reference to the vertical line), and the angular velocity of the pole. In order to discover a sequence of correct control actions over this four dimensional space, it is easier if the space is quantised so that the problem remains tractable. The quantisation can be fixed *a priori* or learnt from the system behaviour [4]. In the solution proposed in this paper, as was the case in [1] also, we will assume that the quantisation is determined *a priori*. The effects of such a fixed quantisation scheme on the performance of both the algorithms are discussed at the end of Section 4. The system equations, parameters, and the state space quantisation scheme are shown in Table 1. The controller is deemed to have failed if the pole angle exceeds the specified limit or the cart reaches either end point of the one dimensional railing. Based on the failure signal the system needs to adjust its decisions and internal mechanisms that give rise to these decisions such that the future performance is improved.

Table 1 about here

Several algorithms have been proposed for solving this problem (See [3] for a good review) but ASE/ACE method appears the most general method of all these. The adaptive critic approach belongs to the family of temporal difference (TD) learning algorithms wherein system states are evaluated based on a delayed reinforcement signal [5]. The work presented here reports simulation results of the ASE/ACE system of Barto et al [1] on benchmark parameters and compares this approach with the new method using the neuro-resistive grid.

In the adaptive critic method, Barto et al [1] introduced a way of coping with delayed reinforcement information. Each system state is associated with an action and its evaluation. The action associated with a state is stored in the value of the weight connecting the decoder and the associative search element (ASE).

The evaluation of a state is stored in the value of the weight connecting the decoder and the adaptive critic element (ACE). Value attached to a state indicates how often that state was part of a sequence of actions that led to system collapse. The system has to enter a state before its evaluation can be initiated. So it takes long time to build the value map over the state space and thereby leads to long learning times with this algorithm. In the work reported here, the basic architecture of Barto et al [1] has been retained except for replacing the adaptive critic element (ACE) with the neuro-resistive grid (NRG).

The design of the neuro-resistive grid technique was inspired by the shape of the evaluation maps produced by the Temporal Difference (TD) learning methods in a maze solving task [6]. The potential distribution in the resistive grid results from the flow of current away from the goal state. It does not exactly have the same shape as the evaluation map and hence it does not produce exactly the same sequence of actions as in the TD learning method. However, qualitatively the resistive grid technique captures the essence of TD methods, assigning higher values to points close to the goal and proposing realisable paths that avoid obstacles. Due to the lateral connections in the resistive grid, evaluations spread rapidly which improves generalisation, which is a weak point of TD learning methods [6]. TD learning methods are part of Dynamic Programming methods such as Q-learning which evaluate state-action pairs instead of states alone [7]. For Q-learning also it was observed that lateral spread of evaluations leads to better generalisation [8]. Lateral communication also reduces exploration time and accelerates learning as will be outlined in later sections.

A detailed description of the adaptive critic approach is given in the next section. Neuro-resistive grid method is outlined in Section 3 and its application to the pole-balancing problem is given in Section 4. Simulation results of these two systems (adaptive critic approach and the neuro-resistive grid method) are presented with a comparative discussion in Section 5. We will conclude with a summary and an outline of future directions in Section 6.

2. Adaptive Critic Approach

Since the new method proposed here is a variation of the adaptive critic approach of Barto et al [1], the latter will be described in detail below. The control system configuration of Barto et al [1] is shown in Table 2(a) and the equations for the system components are shown in Table 2(b). In the controller proposed by them, input to the controller is a vector of four variables specifying current values of the cart position (x), its velocity (\dot{x}), angle of the pole (θ), and its angular velocity ($\dot{\theta}$). This four dimensional space is then quantised (the quantisation parameters are given in Table 1) into 162 regions by the decoder. The decoder converts the input state vector into a 162-bit binary number (d_i) that has a unit value for the bit corresponding to the region that the input vector belongs to and a zero value for the remaining bits. Each of the regions (boxes) of the decoder is connected to both the associative search element (ASE) and the adaptive critic element (ACE) by a weight, u_i and v_i respectively. In other words, each of the 162 system states has its own ASE and ACE weights, u_i and v_i , respectively. Based on the sign of the weight (u_i), ASE generates a binary control action (y), which in turn is transformed into a control force (F) to be applied to the right or left of the base of the cart. The weights (v_i) attached to the adaptive critic element (ACE) reflect the evaluation of states of the system. The ACE receives an external reinforcement signal (r) when the pole cart system fails. The temporal traces, e_i and \bar{a}_i , in ASE and ACE respectively, help keep track of the time elapsed from the last visit to the state. This trace mechanism helps apportioning the blame for failure of the system to various states. As there is no intermediate reinforcement signal available to evaluate the actions of the ASE, an internal reinforcement signal (\hat{r}) is generated using the ACE weights (v_i). The \hat{r} signal is used to train the ASE weights.

Table 2 about here

The weights to ASE and ACE are set to zero initially. The pole cart system always starts from the centre of the railing with zero initial condition on all the variables. Initial control actions from the ASE are random because of the noise term ($noise(t)$) in the ASE output equation in Table 2(b). This is a Gaussian

noise term with zero mean and a standard deviation of 0.01. This small random term enables the controller to explore the space in the absence of a known control action. The learning in ASE results in enhancing the weights so that eventually the weight value can overcome the random noise term, thereby generating a known (non-random) control action. The control action amounts merely to applying a constant control force (F) on either the left or the right of the base of the cart. The system equations (in Table 1(c)) are updated using this control force. The equations are integrated numerically using Euler’s method with a time step of 0.02 s. The decoder determines the next box that the system enters by decoding the new system state vector. The above process is continued until the system fails. The time period that the system keeps the pole balanced is called a “trial”. In our simulations, one hundred such trials constitute a “run”. A run is terminated before 100 trials if the total time of balance exceeds 500,000 time steps as in Barto et al [1]. This leads to unequal number of trials in different runs. In order to compare the performance across “runs”, the data has to be adapted suitably. These details are discussed in Section 5.

On the very first trial in each run, there is no evaluation ($p(t)$) from ACE as the initial ACE weights (v_i) are set to zero. The external failure signal (r) is always set to zero and made equal to -1 only when the system fails. So when a failure occurs, the ACE weights of the boxes that the system visited will be updated as per the equation in Table 2(b). It is clear that, since the external reinforcement signal (r) is -1 , the ACE weights will always be negative. A strong negative value for ACE weight (v_i) indicates that after visiting this state, the pole cart system often failed. Whereas a value close to 0 for ACE weight, indicates that the state is associated with prolonged balancing episodes. Barto et al [1] termed the weights as reflecting prediction of failure. Hence they call $p(t)$, the *prediction* signal. Alternatively, it can also be seen as the *evaluation* of a state and the resulting set of values over the state space, as a *value map* (for a general discussion of the idea of value map in neuro-resistive grid, refer to [2]). According to Barto et al [1], the failure signal leads to punishment of all the recent control actions of ASE that were preceding the failure and results in increasing the prediction of failure in all the recent boxes in ACE. In order to keep track of the visitations, a trace variable is turned on in each box whenever the system enters that box. The trace in

ASE keeps track of both the nature of the action (right/left) and the length of time since that action took place. The trace in the ACE, however, does not have a sign component. ASE traces are reset on every trial but ACE traces are reset only at the beginning of a run. The effect of resetting the ACE traces every trial on the learning speed is discussed in Section 5. The learning of the ASE weights is accomplished by an internal reinforcement signal provided by the adaptive critic element (ACE). ACE computes the internal reinforcement signal (\hat{r}) by comparing the *value* of the current state and that of the previous state. Using this internal reinforcement signal evaluations of all the visited states (i.e. the weights between decoder boxes and ACE) and actions performed (i.e. the weights between decoder boxes and ASE) are adjusted in proportion to the recency information given by the ASE and ACE traces, e_i and \bar{a}_i , respectively. As shown in the equations for weights in Table 2(b), the updated weights are used for calculating prediction, $p(t)$ and control action y in the next time step and are updated continuously throughout the learning period. When the learning is complete an ASE weight reflects appropriate control action for that box such that the pole remains balanced for long time periods.

The internal reinforcement signal (\hat{r}) is positive if the system moves from an “unsafe” box to a “safe” box” and is negative if it is the other way round. With this signal the controller can modify both its actions and its state-evaluations continuously and does not have to wait until the actual failure to occur before any modification can take place. Another notable feature in the evaluations made by ACE is that every move by the system will have consequences on the predictions of all the previously visited boxes in that run with the help of traces in ACE. For example, if the system moves from a “safe” to an “unsafe” box all the “live” (whose traces are turned on) ACE boxes are punished, that is, the prediction of failure is increased as a result of this move. Thus traces in ACE enable a form of generalisation across boxes. It needs to be emphasised here that each box (state) must have been visited at least once before its evaluation can be assigned. This sole fact implies a lengthy training procedure for the ASE/ACE system.

In the neuro-resistive grid approach, generalisation is achieved through lateral connections to the neighbouring states in the grid. Thus the structure of the grid itself makes the propagation of the value-

information across boxes without resorting to traces in ACE. This mechanism reduces the number of computations dramatically as will be demonstrated in Section 5.

3. Neuro-Resistive Grid (NRG) Approach

3.1 Laplacian Methods

In this section, Laplacian methods which are the precursors to the neuro-resistive grid approach, are discussed. Laplacian function methods were introduced for robot path-planning problems (see [2] for review). Connolly et al [9] proposed the use of harmonic solutions of Laplace's equation as the path lines for a robot moving from a start point to a goal point. They considered obstacles as current sources and the goal as a sink (potential fixed at zero). These conditions amount to defining *Dirichlet* boundary conditions for solving Laplace's equation. Using this solution a potential field distribution can be calculated. Given a starting point the path to the goal can be easily constructed by following current lines, which amounts to performing a steepest descent on this potential field (that is, finding a succession of points with lower potentials that lead to the lowest potential in the domain which happens to be at the goal point). It has been demonstrated by Connolly et al [9] that the path constructed by using gradient descent guarantees a path to the goal without encountering local minima and successfully avoiding any obstacles. If the positions of the goals and obstacles are fixed then the potential field has to be calculated just once and can be used for any starting point.

Tarassenko & Blake [10] modelled obstacles as non-conducting solids in a conducting medium, the starting point as a current source and the goal as an equal and opposite current sink. These conditions amount to specifying *Neumann* boundary conditions for solving Laplace's equation. As the normal derivative is fixed at the boundaries (*Neumann* conditions) of the domain, the range of values of the potential gradients will be bounded and the gradients will not decay with distance as is the case with *Dirichlet* conditions. Once solutions of Laplace's equation are found under these boundary conditions, a

potential field can be computed. A gradient descent on this potential field determines the path from starting point to the goal. Tarassenko & Blake [10] asserted that a resistive grid implementation of this method overcomes the computational problems inherent in this method.

3.2 Neural Network Implementation of Laplacian Methods

Bugmann et al [2] proposed a neural implementation of the resistive grid method, the Neuro-Resistive Grid (NRG) method. The domain is discretised and mapped into the nodes of a grid of neurons connected by weights. The information of the positions of the starting point, goal and obstacles are kept in a “memory” layer which has the same number of nodes as the neuron-grid. The neuron-grid and the memory layer are connected in a one-to-one fashion. Thus the activation of some of the nodes in the neuron-grid can be held fixed from the memory layer. Thus each neuron i calculates its activation or potential z_i as follows:

$$z_i = f\left(\sum_{j=1}^n W_{ij} \cdot z_j + I_i\right); \text{ where } W_{ij} \text{ is the weight of the input from neuron } j \text{ to neuron } i, I_i \text{ is the input}$$

from the memory and $f(\bullet)$ is the activation transfer function, which is a linear saturating function such as:

$$f(\zeta) = \begin{cases} 0 & \text{if } \zeta < 0 \\ \zeta & \text{if } 0 < \zeta < 1 \\ 1 & \text{if } \zeta > 1 \end{cases}$$

By using $W_{ij} = 1/n$ where n is the number of neighbours ($= 2m$ where m is the dimension of a square grid), the neurons set their potentials to the average potential of all the neighbours. Bugmann et al [2] have shown that this formulation leads to a Poisson equation and the solution of this Poisson equation determines the activation (potential) distribution of the neurons in the NRG. The shape of the potential distribution in NRG depends on the encoding scheme of the forbidden states (obstacles), whether the corresponding nodes are current sinks (Dirichlet condition) or are disconnected from the grid (Neumann condition) [2].

4. Application of NRG to the pole-balancing problem

In the work reported here the ACE is replaced by an NRG. As described in Section 2, ACE is developing a value map of the system states. The rationale for replacement is that by propagating the failure information across the grid (especially to the immediate neighbours), in the future if the system enters this neighbourhood, the NRG will send a failure prediction signal that enables the ASE to adjust the recent control actions performed. This in turn prevents the system from performing a similar sequence of actions in the future. Thus even if some states are not visited in the previous trials, the NRG system deems them as safe if they are away from “bad” boxes and unsafe if they are near a “bad” box. Whereas in ACE, if a particular box is not visited so far in a run, there is no prediction available for that box. Thus in the NRG system it is possible to build up a value surface relatively quickly.

As shown in Figure 1, a four-dimensional resistive grid is constructed for the four variables, position (x), angle (θ), velocity (\dot{x}) and angular velocity ($\dot{\theta}$). In Figure 1, the neighbourhood for one cell is shown schematically. The control system set-up that includes the NRG, is shown in Figure 2. Apart from the external input (I_i) from the memory layer, a small bias voltage (0.01) is fed to all the nodes in the grid. Incorporation of the bias voltage is a novel modification of the original NRG technique [2] to adapt it to pole balancing application as discussed below.

In the application of NRG to path planning, there is a goal state, which is a node with the highest potential, that the system is asked to attain. However, the formulation of the pole balancing problem does not define a goal state for the pole. It is only specified that falling should be avoided. Therefore we have assigned an initial goal value to each state of the pole-cart system by providing each node in the grid with a small current source produced by the bias voltage in the memory layer. States where the pole falls during training are transformed into current sinks. This progressively leads to a potential distribution with highest values for states where the pole is close to a vertical and centred position, as in the ACE evaluation function.

The weights between neighbours are set to $W_{ij} = (1 - \text{bias})/n$ where n is the number of neighbours and bias is set to 0.01. This bias term in the weights prevents saturation of node voltages. At the beginning of every run the grid is updated (i.e. every node activation is replaced by the average activation over its neighbourhood) until the potential field stabilises. In our simulations we observed that 30 cycles of updating achieves a stable potential field. The grid is not updated again until a failure occurs. On failure, the node corresponding to the failure is set to a potential of -1 in the memory (shown as ‘failure memory’ in Figure 2) layer and the grid is cycled for 30 times. A run is terminated if the system reaches a cumulative time count of 500,000 time steps or if there are 100 failures (trials), whichever happens first (the same scheme was used in Barto et al [1]). This amounts to a total simulation time of balancing approximately equal to 2.8 hours. The actual simulation, performed on Cortex-Pro neural network simulation package running on a PC-486 DX2-50, took approximately 8 hours for the NRG algorithm. It is interesting to note that the adaptive critic algorithm took on an average about 18 hours to complete a run.

At each time step, internal reinforcement (\hat{r}) is computed by taking the difference between activation (in the resistive grid) of the present and previous nodes (boxes) that the system entered. This signal in turn adjusts all those ASE weights whose eligibility traces are active. In the adaptive critic approach the ACE predictions are also adjusted every time step whereas the potential field in NRG is updated again only after a failure. Thus the NRG approach is computationally more economical than the adaptive critic approach. The number of updates of NRG is a constant proportional to the number of failures whereas the number of updates in ACE is proportional to the number of time steps of balance (which can grow exponentially once the system stays balanced). When the system fails in the NRG approach, the node that the system entered immediately before the failure is noted. This information is made available to the failure memory layer, where this node is clamped to -1 for the rest of the simulation. During the grid cycling period, this information gets propagated to the neighbouring nodes. Hence a potential surface reflecting the relative goodness of a state (value map) evolves in the NRG. As in the system of Barto et al [1], the NRG system also produces a reinforcement signal with the same interpretation. If the system transition is from a “good”

to a “bad” state the reinforcement signal is negative indicating a punishment signal to all the recent actions of ASE. Whereas, if the transition is from a “bad” to a “good” state, the reinforcement signal is positive indicating a reward signal to all the recent actions of the ASE. The NRG system sends a neutral signal to other transitions (see the table in Figure 2b).

Figures 1 and 2 about here.

The quantisation of the state space is prefixed in both the adaptive critic and the neuro-resistive grid algorithms. Performance improvements have been reported with adaptive quantisation schemes (for example see [4]). In these schemes the boundaries of regions that make up system states are made elastic so that they contract or expand based on the behaviour of the system. At the end of training and stabilisation, important regions will have been quantised with a finer resolution and others with a coarse resolution. Thus performance of the adaptive critic algorithm can be enhanced by using an optimal partitioning scheme of the state space. Since the basic mechanism of the NRG algorithm, namely the propagation of information to neighbouring nodes to enable faster generalisation, does not depend on the particular quantisation scheme used, all the advantages of using an optimal quantisation scheme discussed above are transferable to the neuro-resistive grid algorithm. Hence the relative improvements over the adaptive critic algorithm reported in the next section still hold true with a different partitioning scheme.

In the next section simulation results are presented. We are aware of many other approaches to pole balancing (reviewed in [3]). However, since the efforts in this work are aimed at introducing new ways of evaluating states as described above, all the comparisons of performance in the next section, will be made with reference to the system of Barto et al [1].

5. Results and Discussion

Both the systems are run with the benchmark parameters (note especially the increased angle of failure) and the system equations as advised in [3] (indicated in Table 1). Geva & Sitte [3] felt that the previous failure angle of 12° is too restrictive and advised for a 90° failure angle for the benchmark experiment. Both the systems are simulated for 10 runs. All the trials start from the zero initial conditions on the variables. In all of the runs, the system learnt successfully before completing 100 trials (failures). Hence all the runs were terminated before 100 trials. In order to avoid having runs of different lengths, all the remaining trials in a run are assigned a value equal to the higher of the two immediately preceding times of balance [1]. For each run a data point is plotted by averaging the number of time steps to failure in 5 successive trials (see figures 3a & b). Then these results are averaged over all the 10 runs (see figure 4).

Figures 3(a) and 3(b) depict the results for the adaptive critic and NRG respectively for each of the 10 runs. Figure 4 shows the rate of learning for the two systems. The graphs clearly show that NRG is able to learn a successful controller quickly. For example, by about 15 trials an average adaptive critic controller is capable of balancing for about 120,000 time steps whereas an average NRG controller can balance for about 250,000 time steps. Also the rate of learning is faster for NRG compared to that of the adaptive critic. NRG starts learning very rapidly due to the availability of the failure information through lateral connectivity in the grid. In the original Barto et al [1] algorithm, the temporal traces in ACE are reset only at the end of each run. To investigate the role of the traces in the ACE, we ran the simulations of the adaptive critic algorithm wherein the ACE traces are reset at the end of every trial. From Figure 4 it is clear that the learning speed is the lowest if the ACE traces are reset at the end of every trial. By waiting to reset the traces only at the end of a run, the states that were visited in the previous trials will remain active in the subsequent trials. It appears that this facility enables a continuous adjustment of the evaluations and thus leads to faster learning in the adaptive critic algorithm, as demonstrated by the learning curves in Figure 4. In the NRG algorithm, this process is improved further in that the states do not have to be necessarily visited before their evaluation is ascertained. Due to the lateral connectivity, every state will have an

evaluation based on the neighbourhood. We suspect that this lateral diffusion of information is critical to NRG's success.

Figures 3-4 about here.

In order to characterise the performance of both the controllers, two benchmark tests [3] are conducted. The first test determines the capability of the controller to balance the pole from a different starting position other than that of the training period and also characterises the amount of deviation of the angle of the pole from the vertical and that of the cart position from the centre of the railing. The new initial conditions are: $x = -1$, $\theta = -0.1$, $\dot{x} = -1$, $\dot{\theta} = -0.2$. The values of the cart position and the pole angle over the first 1,000 time steps are shown in Figures 5(a) & 6(a) and the values for the rest of the 50,000 time steps are shown in Figures 5(b)-(c) for the adaptive critic system and in Figures 6(b)-(c) for the NRG system. The controller trained by NRG has smaller deviation in both the position and the angle throughout the test period. However the controller trained by the adaptive critic method has larger deviation in these values. This indicates that the AC-controller allows big oscillations both in position and angle whereas the NRG-controller does not. Both the systems have residual oscillations because of the bang-bang nature of the control force.

Figures 5-6 about here.

The second test characterises the dynamic range of the controller. The pole cart system is released from various initial angles and angular velocities, with the cart at the centre of the track. If the system remains balanced for 15,000 time steps, a data point corresponding to the pole angle and the angular velocity is plotted on the graph. Thus a plot of all such points reflects the range of initial conditions from which the pole can be balanced successfully for long time. This range is termed as the dynamic range and it reflects a form of generalisation over the set of initial conditions. We have noticed that not all the trained controllers were successful on the first benchmark test. So we chose the successful controller on the first test that possesses the largest dynamic range as the best controller for each of the adaptive critic and the NRG

systems, for comparison in Figure 7. It is evident from the graph that the NRG-controller has less dynamic range compared to the adaptive critic-controller.

In summary, the first test demonstrates that the NRG-controller allows less deviation in position and angle. The second test reveals that the range of initial conditions over which an NRG-controller can successfully balance is smaller than that of the adaptive critic-controller. Although the NRG system learns a controller faster than the adaptive critic system, the dynamic range is more limited. A comparative summary of performance characteristics of the two algorithms is compiled in Table 3.

Figure 7 & Table 3 about here.

These results suggest a possible trade-off between the learning speed (computational expense) versus the dynamic range (generalisation over the set of initial conditions). We observed in our simulations of NRG that some of the ASE weights are either not modified at all (because the system never visited these states) or they are very small, of value comparable to the noise term, the later being due to successive small positive and negative weight changes which cancel each other. These small or unset weights are a probable cause of lack of dynamic range which we discuss below as due to a “lack of experience” and due to “rigidity of value map”. Another possible reason for limited dynamic range is a non-optimal encoding of sequences of visited states in the weights. This is discussed below as the “relearning problem”.

(i) Lack of experience: The NRG converges very rapidly to an almost correct evaluation of states and thus provides correct feedback very early to the ASE. Thereby, for each visited state the ASE learns a correct action and as soon as a sequence of actions leads to a cyclic pattern of visited states, learning stops as no new states are visited. Thus, although ASE is being successful quickly, it has less knowledge of control actions over the state space.

While NRG learns faster than ACE due to lateral propagation of values, the ASE still relies on the technique used by Barto et al [1], wherein learning takes place only for the states that are actually visited. This suggests the need for matching convergence speeds of ACE/NRG and ASE systems. We have observed in our simulations that the weights of ASE have similar values in neighbouring regions of the

state space. Hence it may be possible to increase the learning speed in ASE also by virtue of lateral propagation of information. Thus achieving generalisation over actions may probably call for a new way of representing actions in the ASE system.

In this context we wish to alert the reader that incorporation of prior knowledge must be considered with caution. Prior knowledge can be incorporated in ACE/NRG by prefixing the evaluations of the edge states where we know failures occur. However, by doing this, ACE/NRG learning may be further accelerated and as a result the dynamic range may be reduced even more because by learning faster the system may not have had opportunity to visit many states.

(ii) Rigidity of value map: In the NRG the values of states are not modified within a trial, i.e., until failure occurs. On the one hand this reduces computational expense but on the other hand it also leads to a rigid value map during a trial which may contribute to cancellation of positive and negative weight changes. In contrast, in the ACE valuations of states are modified at each step so that returning to a previous state will not, in general, result in an internal reinforcement signal (\hat{r}) of the same value but opposite sign leading to similar cancellations as in the NRG. It remains to be verified how important this effect is.

(iii) Relearning problem (of evaluations for failed states): In the NRG method, a failed state remains clamped as failed for the rest of the run. It must be noted that the quantised states used here cover a large portion of the system state space. Depending on how a state is entered, a given action may lead to a successful sequence of movements, while another action may lead to a failure. Thereby a too early definitive evaluation of a state may prevent a number of potentially successful sequences of states to become part of the dynamics encoded in the ASE weights. In contrast the adaptive critic approach allows a failed state to alter its value if it becomes part of a viable sequence of balancing forces in the future. This appears to be a subtle problem which deserves further analysis.

6. Conclusions and Future Directions

In summary, an NRG algorithm is presented for the pole-balancing problem and compared to the ACE algorithm. Both the systems are simulated using the same benchmark parameters. Results from the training phase show that the NRG algorithm learns faster and with fewer updates than the adaptive critic algorithm. This is due to the propagation of failure information to the neighbouring nodes in the NRG. Benchmark tests are conducted to compare the performance of the controllers trained with both the algorithms. Firstly, it is observed that the controller discovered by NRG keeps the angle and position of the pole in a narrow range whereas that learnt by the adaptive critic method allows more variation in both these variables. Secondly, the dynamic range (range of initial conditions for which the controller can balance the pole for long periods of time) of the NRG-controller is relatively less than that of the adaptive critic-controller.

Directions for future investigation have been outlined that lead toward a better understanding of the reasons for the differences and improvements of the NRG controller. There are two potential causes for the reduced dynamic range of the NRG controller. Firstly, it may be that the NRG-controller lacks sufficient experience as the system learns successfully before the occurrence of many failures. This may be compensated by training the system with a wide range of initial conditions rather than training from a fixed initial position. Alternatively, it may be possible to reduce the mismatch between convergence speeds of the evaluations in the NRG and the action policy in the ASE by using a lateral propagation scheme in the ASE as well. Secondly, it may be due to the absence of flexibility in the evaluation of the states. In the current system when once failure occurs in a state, that state is fixed as “failed” throughout the run. One way to improve on this is to allow flexibility in the memory layer so that a failed state can become “good” in future trials if there is sufficient evidence from the behaviour of the system.

In recent times, temporal difference methods have been successfully applied to many interesting control problems such as the mountain car problem [12], backgammon [13], automatic aircraft landing [14], etc. Exploring the application of NRG to these problems may help gain insight into both the NRG and TD methods.

On the theory side, unified view of dynamic programming, reinforcement learning and heuristic search has been proposed [7]. Most of these advances are in the area of estimation of optimal value function. The formal link between TD learning and resistive grid method remains to be investigated.

References

- [1] Barto AG, Sutton RS, Anderson CW. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans. Syst., Man., Cybern.* 1983; SMC-13:834-846.
- [2] Bugmann G, Taylor JG, Denham MJ. Route finding by neural nets. In: Taylor JG (ed.) *Neural Networks*, Unicom & Alfred Waller Ltd. Publ., UK, 1995, pp 217-231.
- [3] Geva S, Sitte J. A Cartpole experiment benchmark for trainable controllers. *IEEE Control Systems Magazine*, 1993; 13:40-51.
- [4] Rosen BE, Goodwin JM, Vidal JJ. Process control with adaptive range coding. *Biol. Cybern.* 1992; 66, 419-428.
- [5] Sutton RS. Learning to predict by the method of temporal differences. *Machine Learning*, 1988; 3: 9-44.
- [6] Barto AG, Sutton RS, Watkins CJCH. Learning and sequential decision making. In: Gabriel M., Moore J. (ed.) *Learning and Computational Neuroscience: Foundation of Adaptive Networks*, MIT Press, Cambridge, MA, 1990, pp 539-602.
- [7] Barto AG, Bradtke SJ, Singh SP. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 1995, 72:81-138.
- [8] Ribeiro CHC. Attentional Mechanism as a Strategy for Generalisation in the Q-Learning Algorithm. In: Fogelman-Soulié F., Gallinari P. (ed.), *Proc. of ICANN'95, Paris, Vol. 1, 1995*, pp. 455-460.
- [9] Connolly CI, Burns JB, Weiss R. Path planning using Laplace's equation. *Proc. IEEE Int. Conf. on Robotics & Automation*, 1990, pp. 2102-2106.

- [10] Tarassenko L, Blake A. Analogue computation of collision-free paths. Proc. IEEE Int. Conf. on Robotics & Automation, Sacramento, CA, USA, 1991, pp. 540-545.
- [11] Sutton RS, Pinette B. The learning of world models by connectionist networks. Proc. Seventh Ann. Conf. of the Cog. Sci. Soc., Irvine, CA: Lawrence Erlbaum, 1985, pp. 54-64.
- [12] Moore AW. Efficient memory-based learning for robot control. Ph.D. Thesis, University of Cambridge, Cambridge, UK, 1990.
- [13] Tesauro G. Temporal difference learning and TD-Gammon. Communications of the ACM, 1995, 38(3), 58-68.
- [14] Prokhorov DV, Santiago RA, Wunsch II DC. Adaptive critic designs: A case study for neurocontrol. Neural Networks, 1995, 8(9), 1367-1372.

Figure Legends

1. A 3x3 grid showing the arrangement of the 162 boxes that constitute the pole cart system state space. The convention for the variables is shown on the sides of the grid and the numbers represent the indices of the quantised regions for each variable. An example of connectivity for one node is illustrated. The node here has 8 neighbours. The nodes on the edges have less number of neighbours. The connection strength is fixed so that the edge affects are balanced.
2. (a) The NRG/ASE system configuration showing the replacement of the adaptive critic element (ACE) by the neuro-resistive grid (NRG) (compare from the adaptive critic system in Table 2a). The internal reinforcement signal is fed to the adaptive search element (ASE) which in turn generates a control action.
2 (b) Table indicates the interpretation of the internal reinforcement signal. The reinforcement signal is set equal to the difference between the *values* of the new and the old boxes that the system entered and η is a constant equal to 0.95.
3. Results of the training phase of the adaptive critic and the NRG algorithms. Time steps to failure for each of the 10 runs with the (a) adaptive critic approach and (b) the neuro-resistive grid (NRG) method are shown. Each data point is obtained by taking an average of time steps of balance over 5 trials.
4. Learning curves comparing the speed of learning in the adaptive critic method versus the NRG method. Each data point is obtained by taking average over the 10 runs shown in Figure 3. The graphs clearly show that NRG is able to learn a successful controller quickly. Two graphs are shown for the adaptive critic algorithm, one with the ACE traces reset at the end of every run and the other with these traces reset at the end of every trial. The former learns faster than the latter and both of these are slower than the NRG algorithm.
5. Results of the first benchmark test on the adaptive critic-controller. (a) Graph shows the cart position and the pole angle over the first 1,000 time steps. The root mean squared (RMS) value and the standard deviation (SD) for the cart position are 0.76m and 0.49, respectively. The RMS and SD values for the pole angle are 0.27 radians and 0.27, respectively. (b) Graph shows the deviation in the cart position for the

remaining time steps till 50,000. The RMS and SD values for the cart position are 0.76m and 0.49, respectively. (c) Graph shows the deviation in the pole angle for the remaining time steps till 50,000. The RMS and SD values for the pole angle are 0.28 radians and 0.28, respectively.

6. Results of the first benchmark test on the NRG-controller.(a) Graph shows the cart position and the pole angle over the first 1,000 time steps. The root mean squared (RMS) value and the standard deviation (SD) for the cart position are 0.81m and 0.18, respectively. The RMS and SD values for the pole angle are 0.17 radians and 0.17, respectively. (b) Graph shows the deviation in the cart position for the remaining time steps till 50,000. The RMS and SD values for the cart position are 0.8m and 0.16, respectively. (c) Graph shows the deviation in the pole angle for the remaining time steps till 50,000. The RMS and SD values for the pole angle are 0.17 radians and 0.17, respectively.

7. Results of the second benchmark test on both the controllers. The pole cart system is released from the centre of the track each time and the initial angle and angular velocity of the pole are varied. For each angle of the pole, the minimum and maximum angular velocities at which the controller can successfully balance for 15,000 time steps are plotted. Thus the dynamic range for the adaptive critic versus NRG controllers is characterised. The NRG controller has lower dynamic range than the adaptive critic controller (See text for discussion).

Polecart parameter	Value
Length of the track	2.4m
Failure angles (θ)	$\pm \pi/2$ rad
Gravity (g)	-9.81m/s^2
Length of the pole ($2l$)	1m
Mass of the cart (m_c)	1kg
Mass of the pole (m_p)	0.1kg
Control force (F)	10N
Integration time step	0.02s

(a)

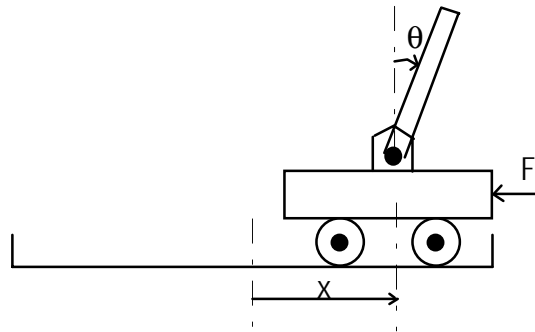
Variable	Range	Region
x (m)	$[-2.4,-0.8]$	1
	$[-0.8,0.8]$	2
	$(0.8,2.4]$	3
θ (rad)	$[-1.57,-0.21]$	1
	$[-0.21,-0.02]$	2
	$[-0.02,0.00]$	3
	$[0.00,0.02]$	4
	$(0.02,0.21]$	5
	$(0.21,1.57]$	6
\dot{x} (m/s)	$(-\infty,-0.5)$	1
	$[-0.5,0.5]$	2
	$(0.5,+\infty)$	3
$\dot{\theta}$ (rad/s)	$(-\infty,-0.87)$	1
	$[-0.87,0.87]$	2
	$(0.87,+\infty)$	3

(b)

$$\frac{d^2\theta}{dt^2} = \frac{g \sin\theta - a \cos\theta - \mu_p \dot{\theta}^2 l \cos\theta \sin\theta}{l \left(\frac{4}{3} - \mu_p \cos^2\theta \right)}; \text{ where } a = \frac{F}{m_p + m_c} \text{ and } \mu_p = \frac{m_p}{m_p + m_c}$$

$$\frac{d^2x}{dt^2} = \frac{\frac{4}{3}a + \left(\frac{4}{3}\dot{\theta}^2 l - g \cos\theta \right) \mu_p \sin\theta}{\left(\frac{4}{3} - \mu_p \cos^2\theta \right)}$$

(c)



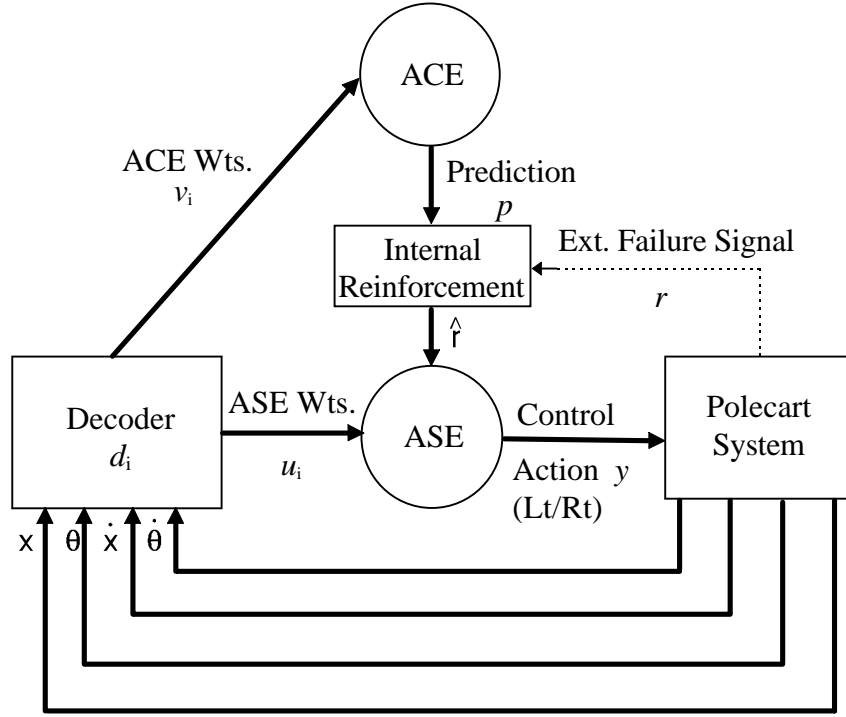
(d)

Table 1 (a) System parameters

(b) State space quantisation scheme

(c) Polecart system equations (A minor error in the system equations given in Geva & Sitte [3] has been corrected)

(d) Diagram of the polecart system



(a)

ACE EQUATIONS:

Prediction from ACE:

$$p(t) = v_i(t)d_i(t); \quad d_i(t) \text{ is the decoded system state}$$

ACE weights:

$$v_i(t+1) = v_i(t) + \beta \hat{r}(t)\bar{a}_i(t); \quad \beta \text{ is a positive learning constant}$$

ACE eligibility traces:

$$\bar{a}_i(t+1) = \lambda \bar{a}_i(t) + (1-\lambda)d_i(t); \quad \lambda \text{ is trace decay rate constant}$$

Internal reinforcement signal:

$$\hat{r}(t) = r(t) + \gamma p(t) - p(t-1); \quad \gamma \text{ is discount factor, } r \text{ is external failure signal}$$

ASE EQUATIONS:

ASE output (control action fed to the polecart system):

$$y(t) = g[u_i(t)d_i(t) + noise(t)]; \quad g[w] = \begin{cases} +1, & \text{if } w \geq 0 \quad (\text{right control force}) \\ -1, & \text{if } w < 0 \quad (\text{left control force}) \end{cases}$$

ASE weights:

$$u_i(t+1) = u_i(t) + \alpha \hat{r}(t)e_i(t); \quad \alpha \text{ is learning rate}$$

ASE eligibility traces:

$$e_i(t+1) = \delta e_i(t) + (1-\delta)y(t)d_i(t); \quad \delta \text{ is trace decay rate constant}$$

(b)

Table 2 (a) ASE/ACE system configuration. Based on the state parameters from the polecart system, decoder determines the box (d_i) that the system entered. This information is used to determine the control action (y) and update all the weights (u_i and v_i). See text for more details. (b) Equations for the Adaptive Critic system.

Property	NRG	AC
Learning Speed	Fast	Slow
Number of Computations	Less	More
Quality of Control	Less angular / position deviation	More angular / position deviation
Dynamic Range	Medium	Large

Table 3 Comparison of the NRG and the adaptive critic algorithms for control.

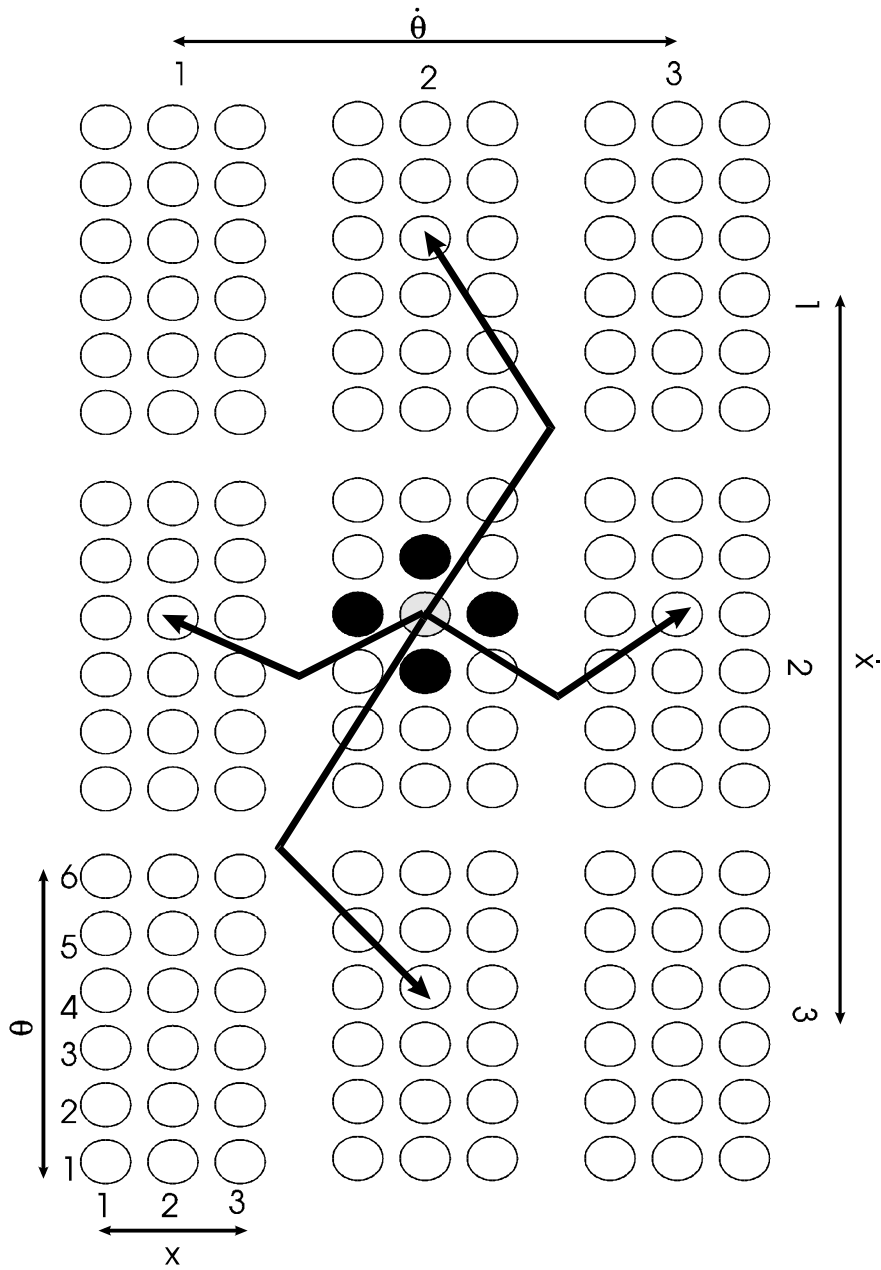
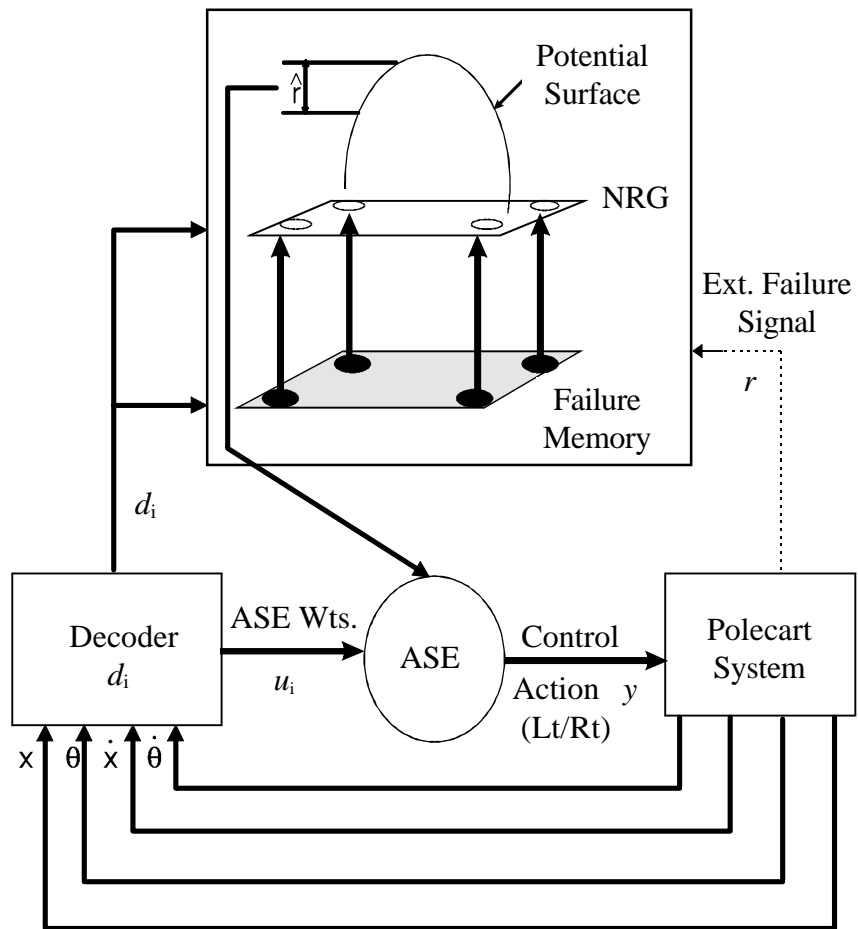


Figure 1



(a)

$\hat{r} = \eta(\text{New Value} - \text{Old Value})$	
\hat{r} Value	State Transition
> 0	'Bad' to 'Good'
< 0	'Good' to 'Bad'
0	'Bad' to 'Bad' 'Good' to 'Good'

(b)

Figure 2

Figure 3(a)

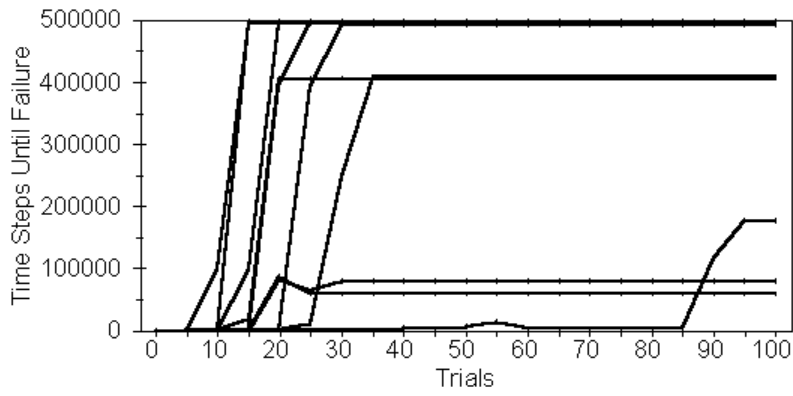


Figure 3(b)

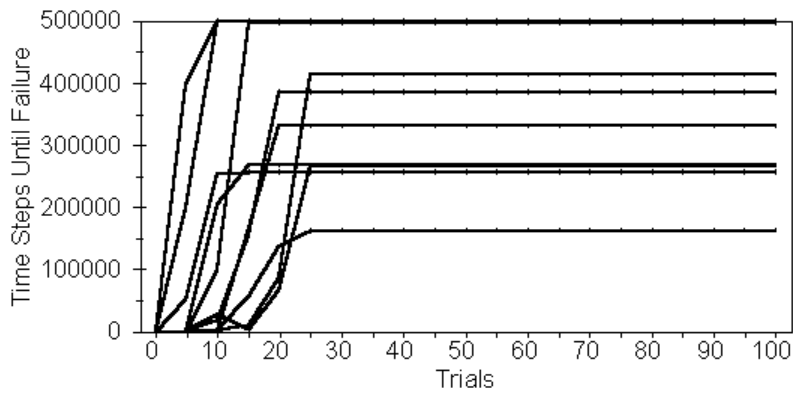


Figure 4

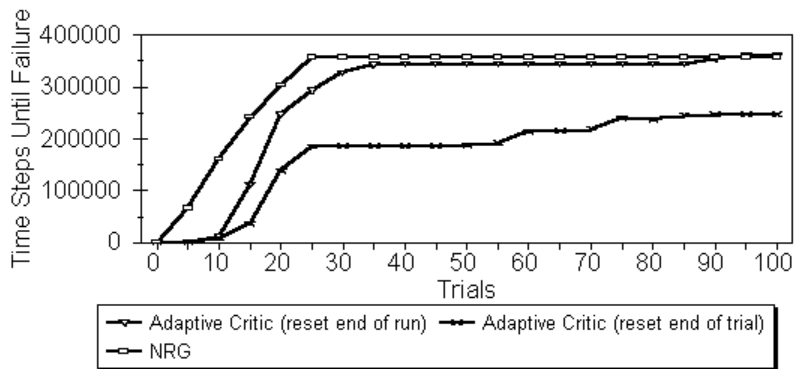


Figure 5(a)

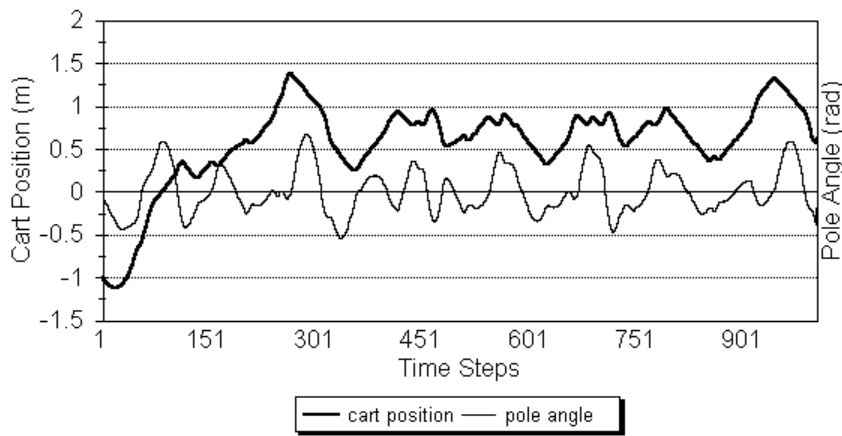


Figure 5(b)

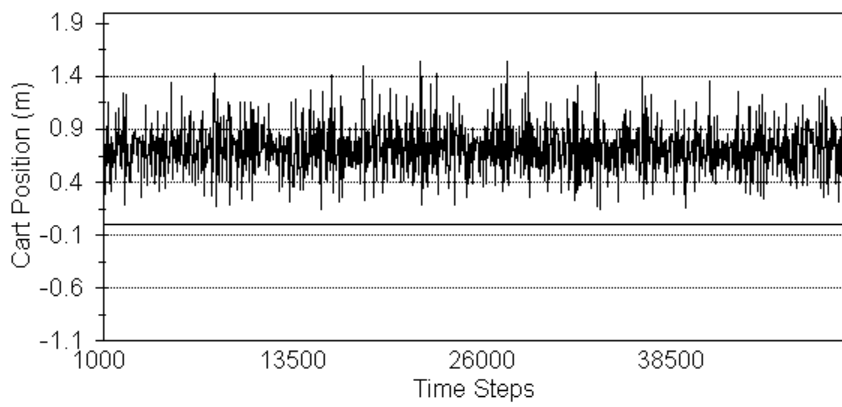


Figure 5(c)

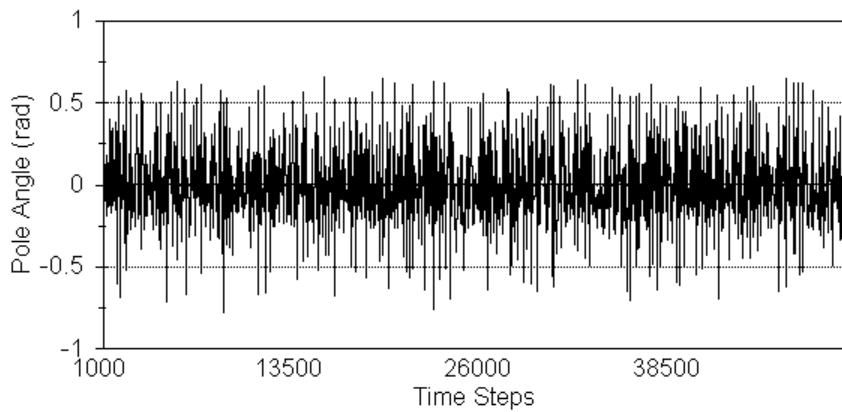


Figure 6(a)

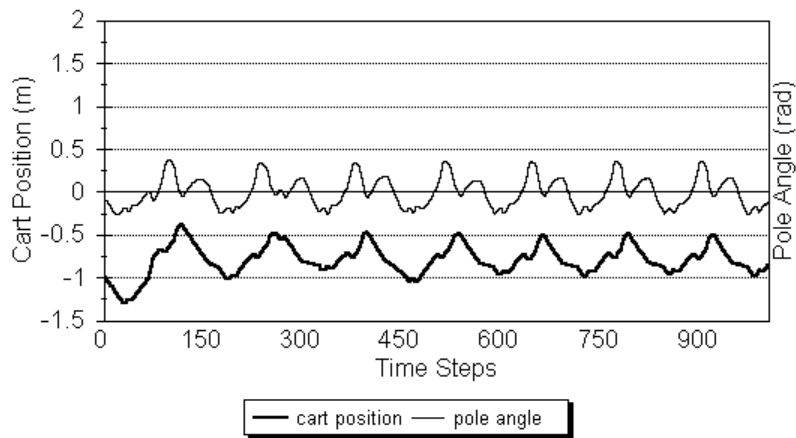


Figure 6(b)

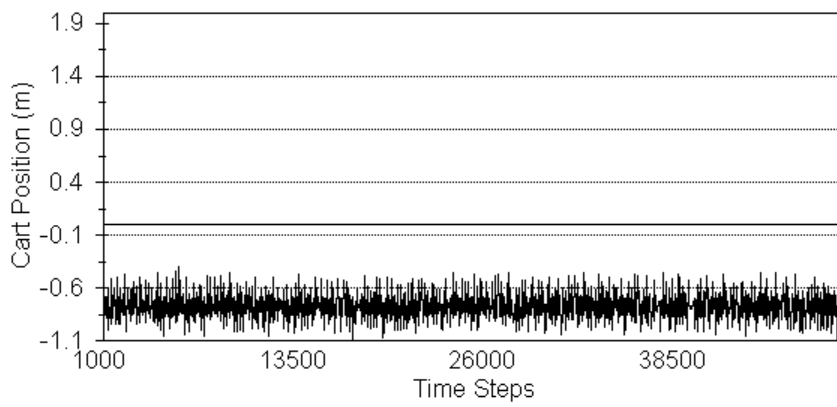


Figure 6(c)

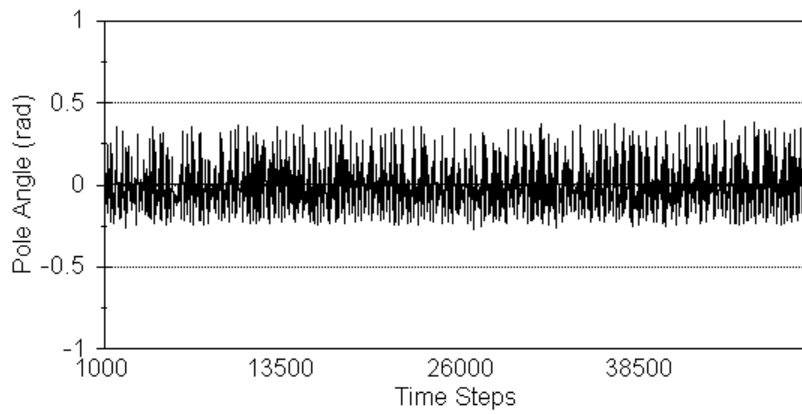


Figure 7

