Neural Networks, Vol.17, pp.925-933, 2004

Learning to Generate Combinatorial Action Sequences Utilizing the Initial Sensitivity of Deterministic Dynamical Systems

Ryu Nishimoto and Jun Tani

Brain Science Institute, RIKEN

2-1 Hirosawa, Wako-shi, Saitama, 351-0198 Japan

Tel +81-48-467-6467, FAX +81-48-467-7248

E-mail ryu@brain.riken.go.jp, tani@brain.riken.go.jp

Original Contribution

Contact Information

Requests for reprints should be sent to Ryunosuke Nishimoto, Brain Science Institute, RIKEN. 2-1 Hirosawa, Wako-shi, Saitama, 351-0198 Japan

Running Title

Learning to Generate Combinatorial Action Sequences

Key Words

Learning, Actions, Initial sensitivity, Deterministic chaos

Learning to Generate Combinatorial Action Sequences Utilizing the Initial Sensitivity of Deterministic Dynamical Systems

Abstract

This study shows how sensory-action sequences of imitating finite state machines (FSMs) can be learned by utilizing the deterministic dynamics of recurrent neural networks (RNNs). Our experiments indicated that each possible combinatorial sequence can be recalled by specifying its respective initial state value and also that fractal structures appear in this initial state mapping after the learning converges. We also observed that the sequences of mimicking FSMs are encoded utilizing the transient regions rather than the invariant sets of the evolved dynamical systems of the RNNs.

1 Introduction

The ideas of forward models (Uno, Kawato, & Suzuki, 1989; Wolpert & Kawato, 1998; Jordan & Rumelhart, 1992) have been utilized widely in the context of planning and executing optimal action sequences in a complex environment. The forward model outputs predictions of the next time step's sensory state based on the current sensory state and action. Cascades of this forward computation in sequences can generate lookahead prediction of sensory sequences for given combinations of action sequences. The action plans for future desired goal states can be generated by conducting lookahead prediction for various action combinations. However, one drawback of this scheme is that the forward model itself cannot regenerate action sequences since action sequences are just given from the outside. When adequate action sequences for achieving certain goals are obtained, they cannot be stored for later re-use without using external memory systems.

In the current paper the authors construct a neuro-dynamical system in which sensory and action sequences for each plan are internally represented as a unified trajectory of the system dynamics that reaches its corresponding goal state. The scheme utilizes the initial sensitivity characteristics of nonlinear dynamics in order to generate various sensory-action plans.

The forward model in the proposed scheme learns to generate different sensory-action sequences by means of its forward dynamics depending on the initial state given to it. Given that there exists a forward dynamics: $f()$ which outputs actions A_{t+1}, sensory anticipation S_{t+1} and internal state X_{t+1} of the next step, which are represented as $(A_{t+1}, S_{t+1}, X_{t+1}) = f(A_t, S_t, X_t)$, the function $f()$ can generate different sequences, e.g. SEQUENCE-1 and SEQUENCE-2, from a distinct initial value of the internal state, e.g. $X_0 1$,$X_0 2$. In our proposed scheme, the sensory-action sequences for future behavior plans are internally represented in terms of the corresponding initial internal states.

A question now arises: "Could this model account for the mechanism of combinatorial action selections in sequences that are required in the general action planning paradigm?" For example, assume that a robot is navigating through a maze environment that is represented by a graph structure consisting of multiple branching points. In this situation, there could exist infinite number of paths that the robot can go through by means of arbitrary combinations of branching. How can the forward model generate this sort of different combinations of actions as dependent on the initial state values? One could speculate that the initial sensitivity characteristics of deterministic chaos can account for the combinatorial mechanism in generating action selections from the view of symbolic dynamics (Hao, 1989; Crutchfield, 1989). Our simulation results, however, will show that complex transient dynamics rather than attractor of chaos plays an essential role in generating valid combinatorial sensory-action sequences. We will discuss generalization and robustness characteristics of the model in learning and generating combinatorial sequences as related to the observed dynamical system mechanism.

2 Model and experiment settings

In our numerical experiments, a maze-like environment is represented simply by a finite state machine (FSM) in which an agent explores, as shown in Fig.1. Each node in the target FSM represents a corresponding landmark in which sensory inputs of (A,B,C,D,E,F,G) are perceived with given binary branching actions of (0,1) at every node. Note that not all possible combinations of 1 and 0 actions are valid since only 1 or 0 is a valid action in some nodes while both of them are valid in other nodes. It should be noted that this FSM is closed; –i.e., starting from the start node "A", the agent goes through one of the goal nodes of "D", "F" and "G" and returns to the starting node within a finite number of steps. If the agent determines the actions of

0 or 1 randomly at each node, the possibility of 1 cycle of returning from the starting node without halting is calculated as 12.5 %. Note also that the number of the agent's possible paths increases exponentially when the agent cycles around this environment more times. We will attempt to show that this sort of combinatorial characteristics in the path generation by FSMs can be imitated by our proposed neural network model. In the following, the employed neural network model is first introduced. Second, the specific experimental setting is explained.

2.1 Neural network model

By utilizing the recurrent-type neural network (RNN) model (Jordan, 1986; Elman, 1990; Pollack, 1991; Kolen, 1994; Giles, Sun, Chen, Lee, & Chen, 1990; Tani & Fukumura, 1995) as shown in Fig2, sensory-action sequences are learned in terms of the forward model. More specifically, the RNN learns to predict the incoming sensory state S_{t+1} and next branching action A_{t+1} by perceiving the current sensory inputs S_t and the branching action A_t through the agent's iterative search. The internal state X_t represented by context units in the input layer is mapped to X_{t+1} in the output layer. As will be discussed in the experimental results section, the mechanism of generating combinatorial branching can be achieved by adequately self-organizing the internal structure utilizing the context units. The seven sensory states of "A", "B", "C", "D", "E", "F" and "G" are encoded locally using seven sensor nodes and the branching actions of 0 or 1 are encoded by one action node. In the experiment, 12 context nodes and 30 hidden nodes are employed.

In the learning of the sample sequences, the connective weights of the RNN can be obtained through the iterative calculation utilizing the back-propagation through time (BPTT) algorithm (Rumelhart, Hinton, & Williams, 1986). In this computation, the RNN is transformed into a cascaded feed-forward network without loops by duplication of the original three-layer network in the time direction. The generalized delta rule (Rumelhart et al., 1986) is applied to the cascaded network to find the weight update vector at each step l as $\Delta w_{ij,l}$, which is:

$$\Delta w_{ij,l} = \delta_{i,l} a_{j,l} \tag{1}$$

where $a_{j,l}$ is the output of the jth unit of the cascaded network. For the output unit, $\delta_{i,l}$ is calculated as

$$\delta_{i,l} = (y_{i,l} - o_{i,l}) f_i'(net_{o,l}) \tag{2}$$

where $o_{i,l}$ and $y_{i,l}$ are the ith output and its target values, respectively and $f_i'(net_{o,l})$ is the derivative of the sigmoid function with its internal value (input summation) in this output unit. For hidden units for which there are no teaching signals, $\delta_{i,l}$ is calculated recursively as the error signals back-propagate through the connections to the unit:

$$\delta_{i,l} = f'_i(net_{i,l}) \sum_k \delta_{k,l} w_{i,k} \tag{3}$$

The update weight vector for all sequences Δw_{ij} is obtained as a summation of $\Delta w_{ij,l}$. In addition to determining the connective weights common to all sequences, the training process of the network should determine the optimal initial state for each forward sequence corresponding to the specific training sequence which minimizes the learning error. The value of the i th unit of context units $x_{0,i}$ in the initial step for each training sequence is iteratively computed by the steepest descent method utilizing δ information in the initial step. This iterative search of the initial state is conducted for each sequence, simultaneously with the one for the connective weights. To accelerate the learning speed, a momentum term is included in the update of both weights and initial states

$$\Delta w_{ij}(n+1) = \eta \delta_{i,l} a_{j,l} + \alpha \Delta w_{ij}(n) \tag{4}$$

$$\Delta x_{0,i}(n+1) = \eta \delta_{0,i} + \alpha \Delta x_{0,i}(n) \tag{5}$$

where n indexes the iteration time in the learning, and η and α are the coefficients of the learning rate and the momentum, respectively. In the actual simulation (detailed later), only two context unit values are updated in the initial step. The values of the remaining context units are fixed at 0.5, in the initial step. The training procedure employs a parameter control for the learning rate. Our tests revealed that learning proceeds through a precise structure of bifurcation and that, if the learning speed is too high, the structure becomes corrupted, showing a sudden increase in the error. Thus we introduced a heuristic control scheme in which the learning rate η is adjusted by using the following annealing scheme,

$$\eta = \eta_0 exp(\frac{-n}{n_{max}\beta}) \tag{6}$$

where n_{max} is the maximum iteration time of the learning, η_0 is the initial learning rate, and β is the coefficient of annealing. In this manner, the learning speed is polynomially reduced as the learning error E becomes smaller. It is expected that the learned RNN can regenerate each sequence by means of its forward dynamics computation , starting from each initial state, as determined from when the learning converges.

To generate the action sequences, the FSM environment and the learned RNN are coupled with the sensory-action loop. The forward computation (Jordan & Rumelhart, 1992) starts in the RNN with the setting of the initial sensory and action values and two initial context unit values in the input layer. The action outputs as well as all the context unit outputs are fed into their corresponding units in the input layer for the next step of the forward computation. Simultaneously, the current state in the FSM environment is updated based on the action generated by the RNN. The resultant sensory values given in the FSM environment are fed into the sensory input units of the RNN. This iteration of the forward computation is continued as long as the sensory prediction in the output layer accords with its actual sensation in the FSM environment. Note that this open-loop computation of the RNN coupled with the FSM is completely deterministic.

2.2 Experiment setting

In the learning experiment, a set of the sensory-action training sequences were prepared by utilizing the FSM environment. More specifically, the agent was guided in such a way as to loop around the FSM environment for three cycles, starting from the start node "A". (Cycle 1 starts at node "A", and the agent must pass through one of the goal nodes "D", "F", or "G" before it can return to "A"). Because of multiplicative combinations of branching at each node, there exist $3^3 = 27$ different sequences for the agent traveling through three cycles. In the learning experiment, 21 sample sequences out of the 27 were used for the purpose of testing the generalization capabilities in the learning process. The learning parameters were set as $\eta_0 = 0.013$, $\alpha = 0.65$ and $\beta = 0.3$. The initial values of connective weights and biases were randomly set between -1.0 and 1.0.

The questions that could be studied with the experiment are as follows:(1) How is the initial sensitivity utilized in generating complex sequences? (2) To what extent can the learning be generalized by using only partial training data? (3) What type of dynamics evolves for embedded complex sequences? (4) How robust is the dynamics in terms of generating sequences?

3 Results and Analysis

The learning process converged with a mean square error of 4.16×10^{-6}. Figure 5 shows the time development of the mean square error during the learning process. The details of the convergence process will be explained later. Firstly, we observed how the sensory-action sequences could be generated depending on the two variables of the initial internal state by plotting their phase diagram using the RNN obtained in the end of learning after 49900 learning steps. For this purpose, multiple steps of the forward activation are computed while varying the setting of the initial values of the two context units used for adaptation in the learning phase.

Figure 3 shows a mapping between the initial state values and the nodes (“G”, “F” or “D”) reached before returning to the start node during one cycle. The colored regions (differently graded brightness) represent the regions of the initial state that lead to the goal nodes “G”, “F” or “D”. Interestingly, it can be seen that the initial state space is first divided into two regions of “D” and (“G”+”F”) and further divided into sub-regions of “G” and “F”. It is considered that the similarity of the sequences that reach “G”, “F” or “D” results in this sort of self-organization of the initial state sensitivity mapping. (Note that “G” and “F” are on the same branch of the FSM, whereas “D” is on a different branch.)

For multiple cycles, it is assumed that the initial state space has to be divided into a number of sub-regions in order to generate all possible action sequences. For comparison, figure 4 shows the same mapping for up to four cycles. The color indicates the goal nodes “G”, “F” or “D” reached by the end of the cycle. In the map of the second cycle, each “G”, “F” or “D” region of the first cycle has been further sub-divided into “G”, “F”, “D” and halting sub-regions.

This means that all possible combinations of action sequences within two cycles are mapped into these separate sub-regions in the initial state space. The black regions represent a halting region for which the initial state sensory prediction is incorrect and the forward computation has halted. Similar recursive division structures in the initial state space were observed in cycle 3 and cycle 4. This suggests that the combinatorial action generation properties of the FSM are imitated by the RNN recursive functions by means of self-organizing fractal structures in the mapping from the initial state to the goals reached. It is noted that the initial sensitivity map obtained here does not represent fractal basin of multiple attractors since this map shows the initial sensitivity of trajectories of only finite steps. The initial sensitivity map can be plotted independent of global or multiple attractor cases. In fact the RNN dynamics obtained in the end of learning turns out to be a global attractor as will be shown later.

We examined how many valid action sequences were generated, as well as how many halting sequences were generated using the learned RNN by varying the two initial context unit values for each cycle, up to six cycles. Table 1 shows the ratio of the number of generated sequences versus the number of all possible sequences for each cycle and the percentage of halting sequences generated. All possible sequences were generated in each of the first three cycles, and nearly all sequences (96.3 %) were generated in the fourth cycle. The halting percentage gradually increases as the cycle is iterated. The ratio of halting sequences generated is at most 35.45% in the six-cycle case. This means that 65 % of the sequences are valid with generating various possible combinations up to six cycles. (Remember that if the agent determines actions of 0 or 1 randomly at each node, the possibility in one cycle of returning without halting is only 12.5 %.) Although the network is neither trained to become exactly identical to the target FSM nor perfectly generalized, it can be said that the network mostly mimics the target patterns especially early steps in the sequences, by capturing certain underlying structures from the limited number of training patterns.

To examine in detail the dynamic structures of the RNN developed during the learning process, the phase plot and the initial state sensitivity mapping were drawn for specific time points in the learning process. Figure 6 shows plots at 5000, 35700, 36000, 49900 learning steps, respectively. The phase plot was drawn by conducting 5000 steps of the forward computation of the RNN while fixing the synaptic weights obtained at each learning step. An average over 6 context outputs and that for the remaining 6 context outputs is computed for the last 1000 steps and plotted in 2-dimensions. In these phase plots, a strange attractor suggestive of chaos appears in the case of 36000 learning steps. Other plots at 5000, 35700 and 49900 learning steps indicate generation of limit cycling dynamics, since only a finite number of points are observed in those plots. Those attractors are identified as global attractors since the same shapes of plots were regenerated by starting from various different initial states. Multiple attractors are also generated during the learning process. The phase plots in Figure 6 (e) show two different attractors generated depending on the initial states at 46500 learning steps. The initial state mappings have similar structures, except in the 5000 learning step case. It seems that the initial sensitivity map stops changing after the learning process has converged. Our further analysis showed that the attractor shape changes frequently from a limit cycle to another one and to chaos that could be global or from multiple attractors, even after only a few learning steps.

The apparent contradiction regarding these observations is that the dynamical structure of the RNN seems to continue to change dramatically in terms of the attractor shape even after the 35700th step where the learning error is minimized and the initial sensitivity map does not change anymore. The mathematical theory of the symbolic dynamics, as well as that of Markov partitions (Hao, 1989), indicates that deterministic dynamical systems can imitate stochastic ones in the branching of FSMs only if they can generate chaotic dynamics. Thus, how can an RNN, exhibiting a limit cycle with fixed periodicity, imitate the FSM characteristics of combinatorial branching? One possible reason explaining this contradiction might be that the transient dynamics, before converging to the limit cycle, is relatively long and complex where the training sequences of limited step length are embedded.

In order to clarify how the sequences are encoded utilizing the transient dynamics, the sequences are regenerated by using each of 21 initial states obtained in the learning process. Figure 7 shows those 21 sequences generated by utilizing the synaptic weights and the initial states obtained after 49900 learning steps. Each sequence is plotted horizontally with its sensory state transitions until the sequence is halted for generating either an illegal sensation or an illegal action prediction. "x" denotes the halting state. The plot of each sequence is followed by a number indicating a valid step length before halting, and a number within parenthesis indicating the step length of the transient sequence before converging to a cycling sequence (Note that all sequences converge to the same cycling pattern since the attractor obtained at step 49900 is a global limit cycle attractor.) It is observed that the valid sequences continue mostly more than six cycles over 30 steps (18 sequences out of 21) and that the transient sequences continue from hundreds to thousands of steps. These results indicate that relatively long steps of valid sequences (compared to the step length for training) are embedded in the transient dynamics by utilizing its initial sensitive complexity. However, this encoding utilizing the transient dynamics becomes stable after 35700 learning steps, as indicated by the fact that the initial sensitivity map stops changing, even though the shape of the attractor still continues to change. The above mentioned analysis indicates that the ideas of symbolic dynamics may not be applicable to the current RNN learning model. Grammatical structures underlying chaos attractors tend to be quite sensitive to the parameters of dynamical systems, as shown in the epsilon-machine reconstructions of logistic maps by Crutchfield (1989). It seems that the RNN in our simulations found ways to utilize transient regions rather than structurally unstable invariant sets of the evolved dynamical systems for encoding target sequences.

4 Discussion

The current study investigated how a specific sensory-action sequence can be recalled from among other learned sequences by setting its respective initial state condition in a forward model by using an RNN. In the case of learning to imitate target FSMs, it was shown that fractal structures are self-organized in the initial state sensitivity map by which most of the possible combinations in branching sequences can be generated. It was also found that action sequences of the target FSMs can be imitated by organizing complex transient dynamics in the network. Even after, learning has almost converged the invariant set of the RNN dynamics dramatically changes between chaos and limit cycles of global, as well as multiple attractors. Nevertheless, our analysis showed that transient trajectories yielded enough valid long-step sequences mimicking the target FSM.

One obvious advantage of the proposed scheme is that sensory-action sequences of arbitrary length can be encoded by their respective initial state values. Additionally, the scheme can be extended to perform goal-directed planning. In the case of our FSM environment, a goal node can be specified in terms of its sensory state and possible action sequences for reaching this node can be searched by generating simulated action sequences by modulating the initial state value for the forward dynamics. If the generated sequence is found to be worth memorizing for future re-use, the system only has to memorize its corresponding initial state value in an additional memory system.

Although there are many prior studies investigating the RNN capabilities for learning FSMs or grammatical structures, no studies have focused on utilizing the initial sensitivity of the deterministic RNN dynamics for generating structured sequences. The Elman net (Elman, 1990) is considered to be a stochastic system equivalent to hidden Markov models in which output activations represent the probability of the next words to be generated. The dynamical recognizer studied by Pollack (1991), Kolen (1994), Giles et al. (1990) is equivalent to an iterated function system where the stochasticity of branching at each FSM node comes from input bit sequences which are randomly chosen. Previously Tani (Tani, 1996) proposed a model of chaos driven planning utilizing an RNN that is applied for navigation tasks of mobile robots. That model, however, is quite different from the one proposed in the current paper. It utilized the conventional forward model, where only the sensation not the action of the next step is predicted upon receiving the current sensation and action. (The model was equivalent to an iterative function system (IFS) with action inputs.) In planning to achieve a specified goal, an action sequence which was optimal in terms of the shortest path to the goal was computed by the inverse dynamics scheme utilizing the back-propagation error through time algorithm (Werbos, 1990) implemented in the RNN. However the steepest descent method to search for the optimal action sequence in the inverse dynamics scheme is easily trapped by local minima. For the purpose of avoiding this local minima problem, an additional nonequilibrium term was added in the steepest descent dynamics which turned out to generate a chaotic search of action sequences. In this case, chaotic dynamics originated from this externally added dynamic system. The action sequence generator and its coupling with the RNN forward dynamics realized complex search processes. On the other hand, in the current model, the RNN forward dynamics, as an autonomous dynamical system, evolves to generate the diverse sensory-action sequences of mimicking the target FSM.

The most essential benefit in the current model might be that the plan search space could be reduced substantially compared to the conventional one. Consider utilizing the conventional forward model for learning the same FSM employed in the current paper. For the purpose of searching an optimal action sequence to a goal state, the conventional forward model ought to be tested with all bit combinations of action 0 or 1 at each node. However, they would include invalid action sequences since some nodes take only a single action. (The forward model could generate illusory predictions for the inputs of such invalid actions.) In the current model, on the other hand, the search space for the action sequences is constrained to valid action sequences since only valid ones are learned to be generated. Although the generated action sequences could be valid only for some initial cycles in our simulations, it might be allowable since humans also may have difficulty in generating such long mental plans. The issue of constraining the search space of action plans is related to the frame problem (McCarthy, 1963) that discusses how to stop inferring the outcomes of infinite action possibilities. The current model would not suffer from this problem since it generates actions plans along only limited paths of actually experienced and learned ones rather than all possibilities. The advantage is that such paths can be represented simply by the initial states and trajectories of an autonomous dynamical system.

The encoding of combinatory action sequences into fractal structures of the initial state is beneficial for mimicking various heuristic plan search scheme. If the initial state is changed subtlely, only the action sequences of long after steps are modulated. On the other hand, if the initial state is largely changed, all steps of the sequences immediately after the initial step are modulated. Either of the depth-first search (Winston, 1992) or the width-first search (Winston, 1992), or their mixtures can be naturally achieved by tuning fineness of searches in the initial state space. Tsuda (2001) showed a different model of encoding sequences into fractal structures. In this model, CA1 network in hippocampus receives inputs from memory dynamics processes considered in CA3 network. As the CA1 network plays the roles of an IFS with receiving input sequences from CA3 of which dynamics is characterized by the chaotic itinerary (CI) (Kaneko & Tsuda, 2003), fractal structures are observed in the state trajectories in this IFS. However, what are encoded in the fractal structures in this model and in ours are different in a complementary manner. Tsuda (2001) encodes the past input sequences fed into the IFS while that in ours does for the future output sequences predicted from the autonomous dynamical system. It is, however, reminded that the fractal structures can encode only limited length of sequences in reality in both cases since their fine structures can be easily perturbed by real world noises.

Although the most of nonlinear dynamical systems studies focus on structures of attractors, a unexpected finding in the current study is that the learning processes utilize transient trajectories rather than invariant sets of attractors for encoding the FSM structure. The invariant sets become observable only after systems run for infinite time periods. However, the invariant sets may not be accessible by any cognitive systems since their processes are always bounded in finite time periods. Consequently, cognitive systems are forced to utilize transient trajectories. Or, it should be said that they adapt to utilize the transient ones. Interesting observations in the current study are that the transient trajectories could be very complex and could continue for long periods even though they end up in simple attractors. This suggests that the transient regions maintains diverse and rich dynamic characteristics which cannot be inferred solely from the attractor characteristics. Future studies should investigate such characteristics of the transient dynamics more intensively in order to explore essential mechanisms of dynamics cognitions.

Another interesting future study might be to examine if animals or humans are using this initial state sensitivity scheme for generating action sequences or plans. In fact, Tanji and Shima (1994) found that specific pre-SMA activities in motor preparatory periods generate corresponding combinatorial action sequences. It could be interpreted that these pre-SMA activities in motor preparatory periods might play the roles of the initial states in our scheme. This initial state hypothesis could be examined by human psychological experiments using transcranial magnetic stimulation (TMS)(Rossi, 2000). After subjects are forced to learn target FSMs, if they show more instability in the later steps of regenerated sequences than in the earlier steps with TMS stimulations in the motor preparatory period, it could support the idea of the initial sensitivity for action generations. Such studies should be conducted jointly by physiology, psychology and cognitive modeling researchers in future.

5 Conclusions

The current study investigated how a specific sensory-action sequence can be recalled from among other learned sequences by setting its respective initial state condition in a forward model by using an RNN. In the case of learning to imitate target FSMs, it was shown that fractal structures are self-organized in the initial state sensitivity map by which most of the possible combinations in branching sequences can be generated. It was also found that action sequences of the target FSMs can be imitated by organizing complex transient dynamics in the network. From these results we can derive the argument that fractal structures as well as transient dynamics might play crucial roles in achieving the higher cognitive functions such as planning in compositional situations.

References

Crutchfield, J. (1989). Inferring statistical complexity. *Phys Rev Lett*, *63*, 105–108.

Elman, J. (1990). Finding structure in time. *Cognitive Science*, *14*, 179–211.

Giles, C., Sun, G., Chen, H., Lee, Y., & Chen, D. (1990). Higher order recurrent networks and grammatical inference. In D. Touretzky (Ed.), *Advances in neural information processing systems 2* (pp. 380–387). San Mateo, CA: Morgan Kaufmann.

Hao, B. l. (1989). *Elementary Symbolic Dynamic and Chaos in Dissipative System.* Singapore: World Scientific.

Jordan, M. (1986). Attractor dynamics and parallelism in a connectionist sequential machine. In *Proc. of eighth annual conference of cognitive science society* (pp. 531–546). Hillsdale, NJ: Erlbaum.

Jordan, M., & Rumelhart, D. (1992). Forward models: supervised learning with a distal teacher. *Cognitive Science*, *16*, 307–354.

Kaneko, K., & Tsuda, I. (2003). Chaotic Itinerany. *Chaos 13 (3)*, *13:3*, 926–936.

Kolen, J. (1994). *Exploring the computational capabilities of recurrent neural networks.* Unpublished doctoral dissertation, The Ohio State University.

McCarthy, J. (1963). *Situations, actions and causal laws.* (Stanford Artificial Intelligence Project, Memo2)

Pollack, J. (1991). The induction of dynamical recognizers. *Machine Learning*, *7*, 227–252.

Rossi, S. (2000). Effects of repetitive transcranial magnetic stimulation on movement-related cortical activity in human. *Cereb. Cortex*, *10*, 802–808.

Rumelhart, D., Hinton, G., & Williams, R. (1986). Learning internal representations by error propagation. In D. Rumelhart & J. Mclelland (Eds.), *Parallel distributed processing.* Cambridge, MA: MIT Press.

Tani, J. (1996). Model-Based Learning for Mobile Robot Navigation from the Dynamical Systems Perspective. *IEEE Trans. on SMC (B)*, *26*(3), 421–436.

Tani, J., & Fukumura, N. (1995). Embedding a Grammatical Description in Deterministic Chaos: an Experiment in Recurrent Neural Learning. *Biological Cybernetics*, *72*, 365–370.

Tanji, J., & Shima, K. (1994). Role for supplementary motor area cells in planning several movements ahead. *Nature*, *371*, 413–416.

Tsuda, I. (2001). Toward an interpretation of dynamic neural activity in terms of chaotic dynamical systems. *Behavioral and Brain Sciences*, *24:5*, 793–848.

Uno, Y., Kawato, M., & Suzuki, R. (1989). Formation and control of optimal trajectory in human multijoint arm movement. *Biological Cybernetics*, *61*, 73–85.

Werbos, P. (1990). A menu of designs for reinforcement learning over time. In W. Miller, R. Sutton, & P. Werbos (Eds.), *Neural networks for control* (pp. 67–95). Boston, MA: MIT Press.

Winston, P. (1992). *Artificial Intelligence, 3rd Edition.* Addison-Wesley, Reading, MA.

Wolpert, D., & Kawato, M. (1998). Multiple paired forward and inverse models for motor control. *Neural Networks*, *11*, 1317–1329.

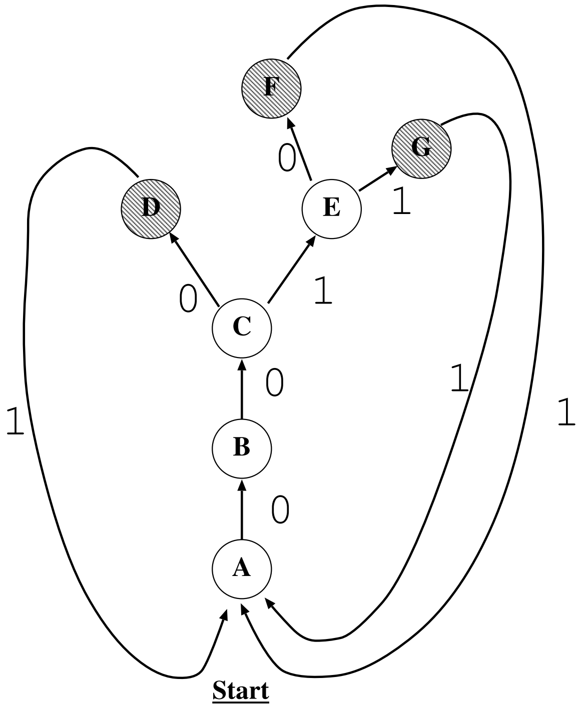


Figure 1: In this finite state machine environment, the agent starts at the start node “A”, and travels to one of the goal nodes of “D”, “F” or “G”, before it can return to the start node in one cycle. The agent perceives sensory inputs of “A”, “B”, “C”, “D”, “E”, “F” and “G” and transits from one node to another depending on 0 or 1 branching action. If the agent takes an invalid action, the process is halted.

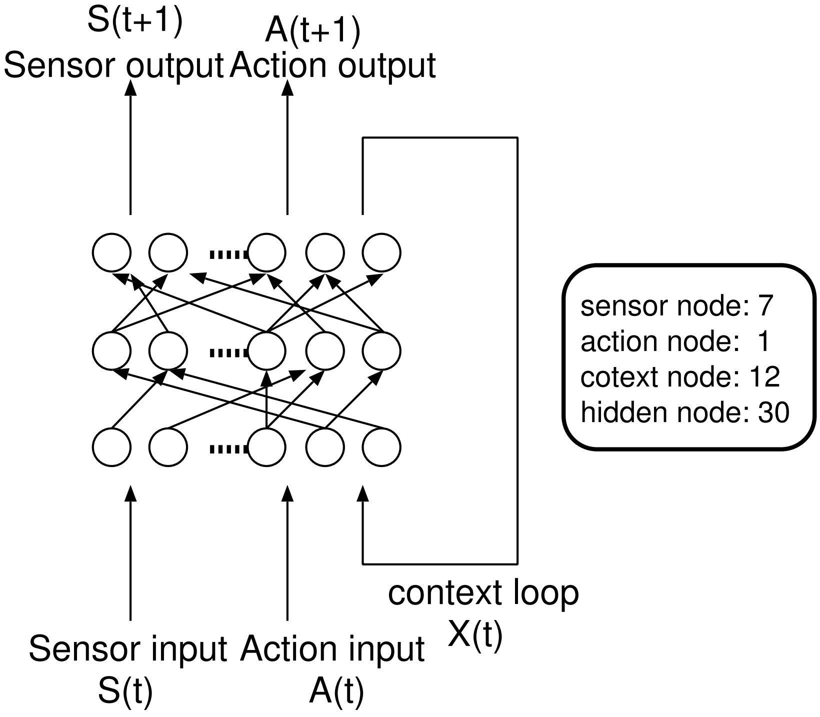


Figure 2: The RNN employed in this paper.

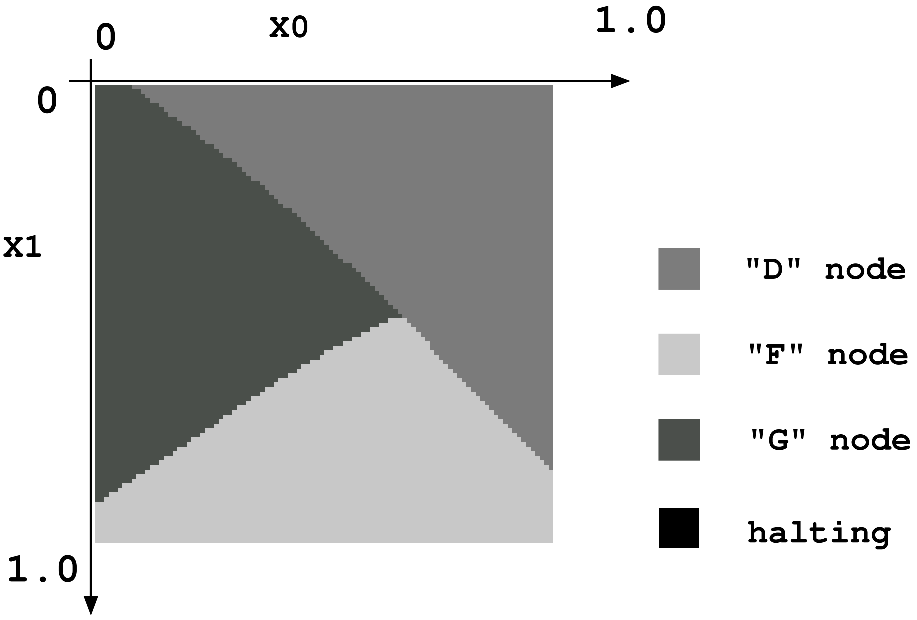


Figure 3: The mapping between the initial state represented by two context unit values and the goal reached among three in one cycle, as shown using different colors.

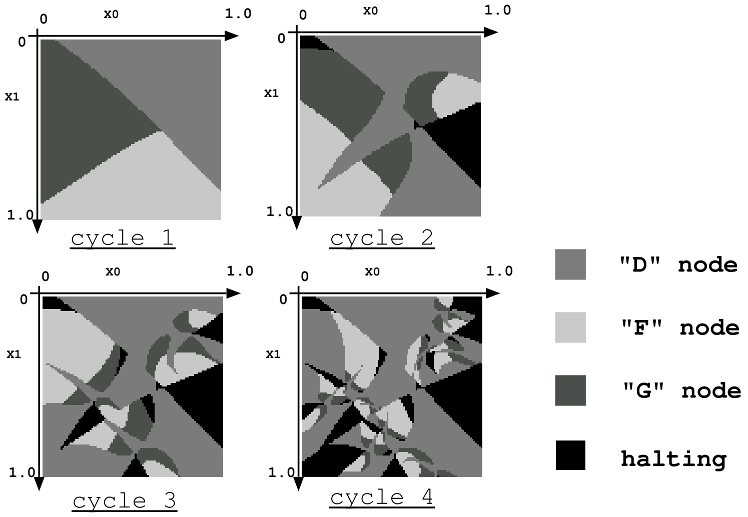


Figure 4: The mapping between the initial state represented by two context unit values and the goal reached at the end of each cycle.

Table 1: Ratio of number of generated sequences versus number of all possible sequences and the percentage of halting sequences generated for each cycle.

Cycle	generated patterns /possible combinations	halting percentage
Cycle 1	3/3 (100.0%)	0.00%
Cycle 2	9/9 (100.0%)	9.09%
Cycle 3	27/27 (100.0%)	14.47%
Cycle 4	78/81 (96.3%)	22.64%
Cycle 5	210/243 (86.42%)	30.51%
Cycle 6	556/729 (76.27%)	35.45%

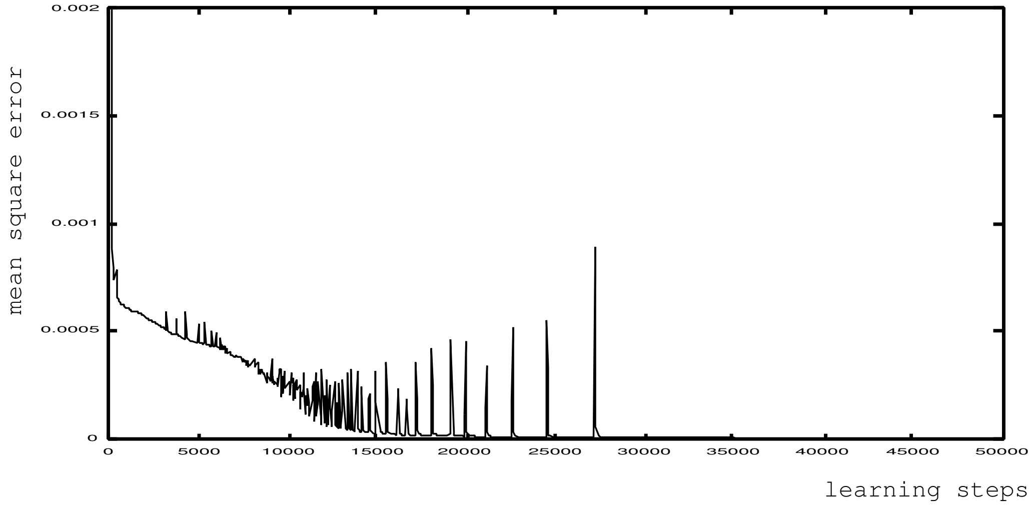
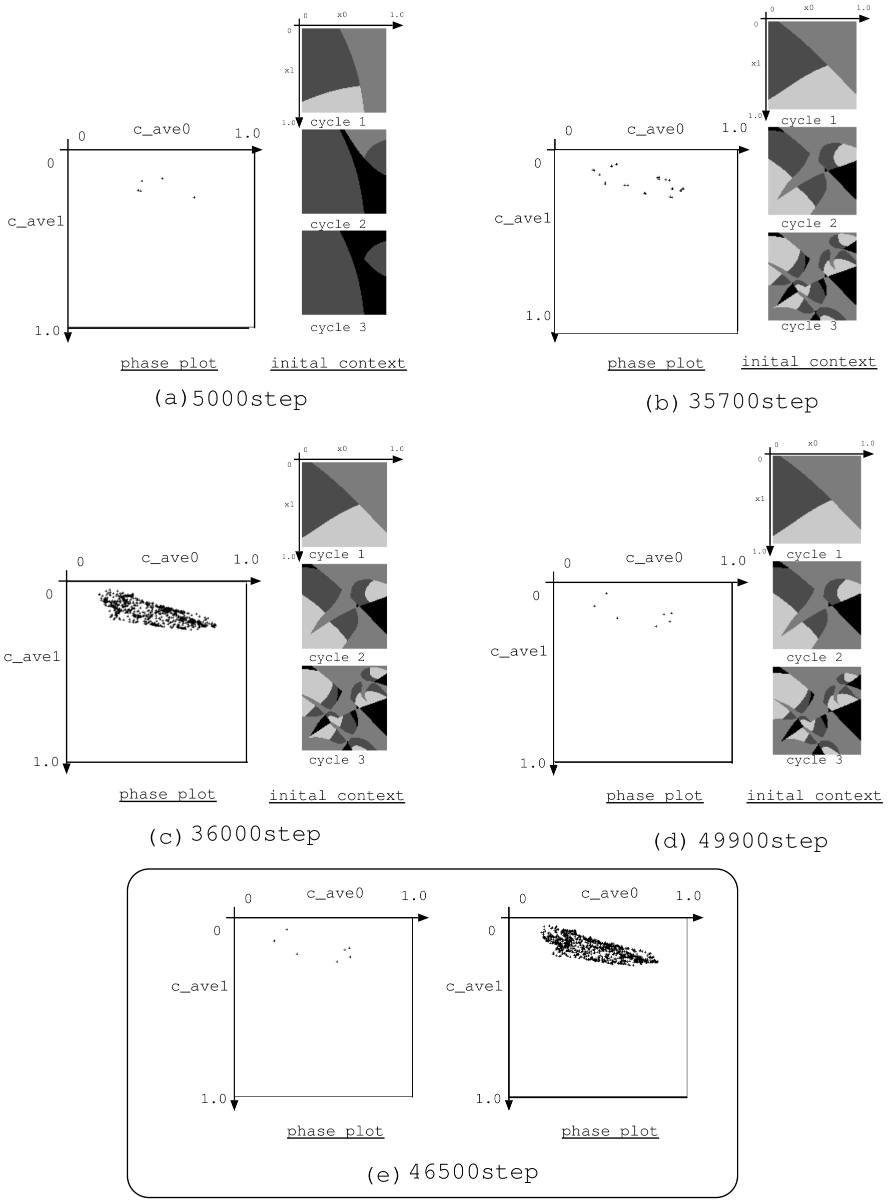


Figure 5: The mean square learning error of the RNN dynamics during the learning process.



(a)5000step

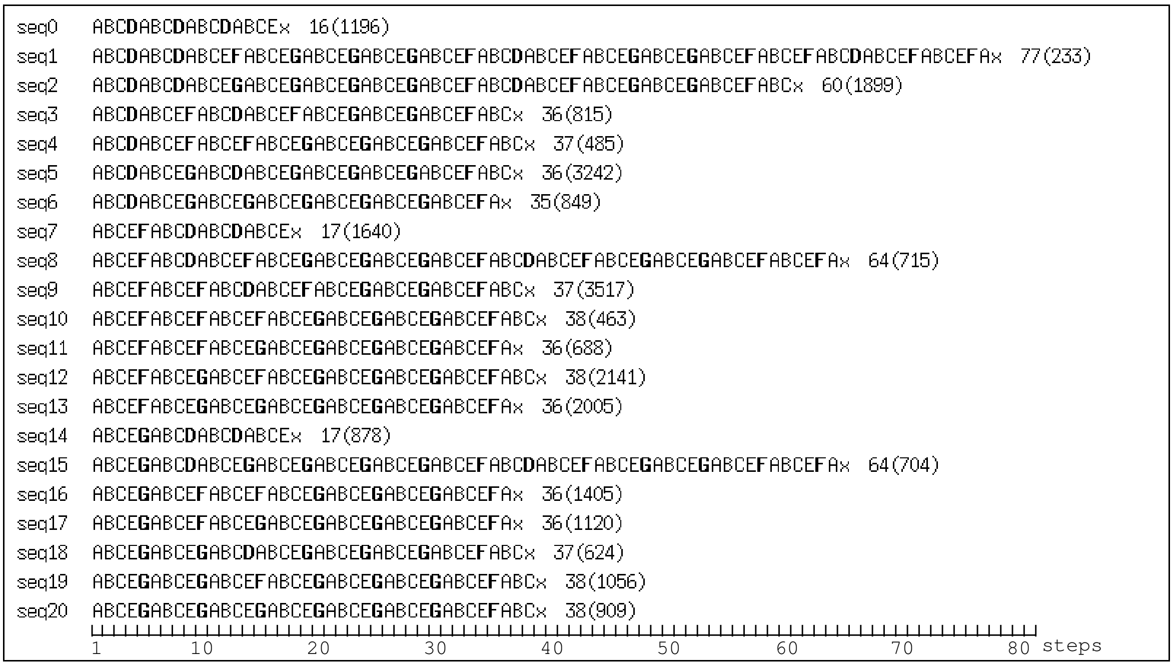
(b) 35700step

(c) 36000step

(d) 49900step

(e) 46500step

Figure 6: The phase plot diagram and the initial state sensitivity map are shown for the cases of 5000, 35700, 36000 and 49900 learning steps. Two phase plots in the case of 46500 learning steps indicate generation of multiple attractors.



```
seq0   ABCDABCDABCDABCEx  16(1196)
seq1   ABCDABCDABCEFABCEGABCEGABCEGABCEFABCDABCEFABCEGABCEGABCEFABCEFABCDABCEFABCEFAx  77(233)
seq2   ABCDABCDABCEGABCEGABCEGABCEGABCEFABCDABCEFABCEGABCEGABCEFABCx  60(1899)
seq3   ABCDABCEFABCDABCEFABCEGABCEGABCEFABCx  36(815)
seq4   ABCDABCEFABCEFABCEGABCEGABCEGABCEFABCx  37(485)
seq5   ABCDABCEGABCDABCEGABCEGABCEGABCEFABCx  36(3242)
seq6   ABCDABCEGABCEGABCEGABCEGABCEGABCEFAx  35(849)
seq7   ABCEFABCDABCDABCEx  17(1640)
seq8   ABCEFABCDABCEFABCEGABCEGABCEGABCEFABCDABCEFABCEGABCEGABCEFABCEFAx  64(715)
seq9   ABCEFABCEFABCDABCEFABCEGABCEGABCEFABCx  37(3517)
seq10  ABCEFABCEFABCEFABCEGABCEGABCEGABCEFABCx  38(463)
seq11  ABCEFABCEFABCEGABCEGABCEGABCEGABCEFAx  36(688)
seq12  ABCEFABCEGABCEFABCEGABCEGABCEGABCEFABCx  38(2141)
seq13  ABCEFABCEGABCEGABCEGABCEGABCEGABCEFAx  36(2005)
seq14  ABCEGABCDABCDABCEx  17(878)
seq15  ABCEGABCDABCEGABCEGABCEGABCEGABCEFABCDABCEFABCEGABCEGABCEFABCEFAx  64(704)
seq16  ABCEGABCEFABCEFABCEGABCEGABCEGABCEFAx  36(1405)
seq17  ABCEGABCEFABCEGABCEGABCEGABCEGABCEFAx  36(1120)
seq18  ABCEGABCEGABCDABCEGABCEGABCEGABCEFABCx  37(624)
seq19  ABCEGABCEGABCEFABCEGABCEGABCEGABCEFABCx  38(1056)
seq20  ABCEGABCEGABCEGABCEGABCEGABCEGABCEFABCx  38(909)
       1        10        20        30        40        50        60        70        80 steps
```

Figure 7: The valid sequences generated for each initial state obtained in 49900 learning steps. The sequences are plotted with the sensory state transitions and terminated with the halting state denoted by “x”. Each sequence is followed by numbers indicating the valid step length and the transient step length.