Dynamical Analysis of Recurrent Neural Circuits in Articulated Limb Controllers for Tool Use

Han Wang, Qinbo Li, Jaewook Yoo and Yoonsuck Choe
Department of Computer Science and Engineering
Texas A&M University
Email: {hanwang, lee, jwookyoo, choe}@tamu.edu

***Abstract*—Recurrent neural networks (RNNs) often show very complicated temporal behavior. In this paper, we will investigate the dynamics of a simple recurrent neural network used in a non-trivial articulated limb control task in a tool use domain. The RNN for the task is evolved by the NeuroEvolution of Augmenting Topologies (NEAT) algorithm. As a non-linear dynamical system, RNNs exhibit strong correlation between their external behavior and internal dynamics. Discovery of the fixed points and limit cycles in the system dynamics will help us understand the roles of neurons and connections in the neural network, and provide insights on how to analyze the function of biology-based nervous systems. We hope our results can provide new criteria to the evaluation of the evolved RNNs, and help the diagnosis of the failures in control problems using neuroevolution.**

I. Introduction

Unlike feedforward neural networks, a recurrent neural network (RNN) allows neurons to send feedback signals which create an internal memory state of the network. As the topological complexity grows, the diversity of the temporal behaviors of the network increases dramatically. This diversity makes RNN applicable to time series problems in many different categories such as robot control [1] and speech recognition [2].

A recurrent neural network can usually be modeled as a non-linear dynamical system. Beer pioneered dynamical approaches to cognitive science. In 1995, he introduced the use of the language of dynamical system theory as a general theoretical framework for the synthesis and analysis of autonomous agents controlled by neural networks [3][4]. However, even though non-linear dynamical systems have been studied for more than 100 years, there are still many parts that are not well understood. For example, Hilbert's 16th problem which is about the determination of the upper bound for the number of limit cycles and their relative locations, remains unsolved [5].

Controllers for tool-use may require RNNs. The use of tools has been extensively studied and is still an attractive topic for scientists from both neuroscience and robotics. The use of tools in animals indicates high levels of cognitive capability, and, aside from humans, is observed only in a small number of higher mammals and avian species [6][7][8][9]. Chung and Choe [10] showed that simple neural circuits can in fact be evolved to use the simplest form of tool, i.e., a "bread-crumb" dropped in the environment to serve as external memory. But it is yet unknown how such a capability could have emerged from simple organisms. In our previous paper [11], we evolved RNN controllers using NEAT for target reaching tasks with a tool. The NeuroEvolution of Augmenting Topologies (NEAT) algorithm by Stanley and Miikkulainen [12] is an evolutionary algorithm for evolving RNNs of arbitrary topology. In this paper, we will analyze the behavior and internal dynamics of the evolved recurrent neural networks.

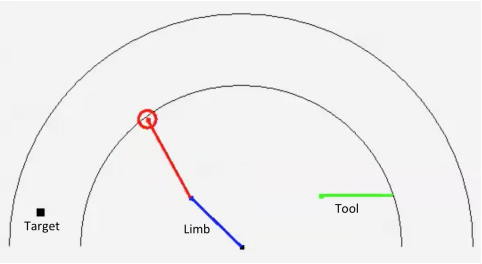
II. Approach

A. Overview

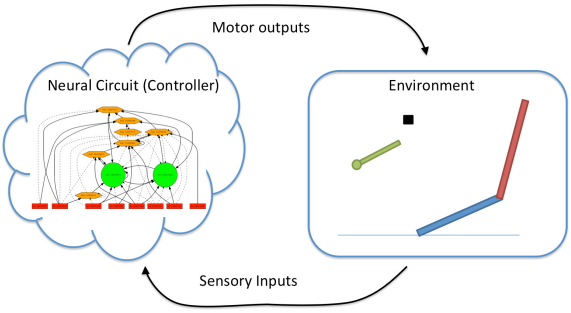
In our previous paper [11], we evolved recurrent neural networks with NEAT algorithm for a simple tool use task. The neural circuit works as a controller for a two degree-of-freedom articulated limb and the task is to reach a target which is out of the range of the arm alone but reachable if the arm grabs a tool (Fig. 1 and 2). The NEAT algorithm creates generalized RNNs which have arbitrary topology. The RNNs can evolve in all aspects: number of neurons, connectivities and weights of connection. The activation function in the hidden and output neurons is a sigmoid function, making the network exhibit rich non-linear dynamics. Since the scale and complexity of the RNNs increases as they evolve, we limited the number of generations to make the scale of the RNNs under control.

In this paper, we evaluated the performance of the RNNs. The correlation of success rate and many parameters such as the starting angles of the arm, and different locations of the tool and the target. Given all the initial values of the input vector, the results are deterministic, however, the results show irregular distributions dependent on some parameters.

Then we examined the neural activity in different parametric trials. We find that the behavior of the arm is encoded in the activities of the neurons. As a non-autonomous dynamical system, in successful trials (target reached), the set of hidden neurons activities approach a stable state when the arm reaches the target. And in failed trials, the hidden neurons fall into many kinds of limit cycles so the arm either oscillates or



(a) Example Task Conditions



(b) Controller-environment Interaction

Fig. 1. The task and the environment. (a) The environment consists of a two degree-of-freedom articulated limb, a target object, and a tool, all on a 2D plane. Note that the tool's initial locations are variable. The two half-circles indicate the original reach (inner) and the reach with tool use (outer). (b) The interaction cycle between the neural circuit controller and the environment. When the end-effector reaches the tool end, the tool is automatically attached to the limb and the end of the tool becomes the end effector. Note: the controller is not explicitly notified of the limb extension due to the tool pick-up. This figure is adapted from [11].

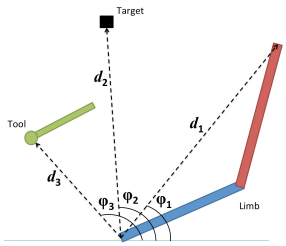
gets stuck before reaching the target.

B. Evolving the Neural Circuits Using NEAT

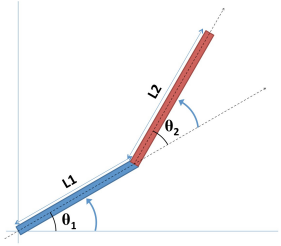
This part closely follows our previous paper [11]:

1) Setup of the Task: The task is to control a two-limbed articulated arm in an object reaching task. The degree of freedom of the arm is 2, and the arm moves on a 2-D plane. The environment was equipped with a tool (a stick) that can be picked up and used to reach objects beyond the range of the limb (Fig. 1).

2) Input and Output: It is important to define the proper sensory inputs to represent the environment properly, especially for artificial agents learning through action and feedback loops [13]. In [14], the author compared two forms of representations: world-centered sensory representation (WC) and agent-centered sensory representation (AC). AC can use polar coordinates while WC can use Cartesian coordinates. The author further proposed the relative agent-centered sensory representation (RAC) which is the relative difference between two ACs. In our experiment, we use RAC as the sensory inputs to the limb controller neural circuit. The details of AC and



(a) Sensory Representation



(b) Kinematics

Fig. 2. Agent-centered sensory representation (AC) and kinematics of the joint arm. (a) AC uses a polar coordinate system. ϕ_1, ϕ_2 and ϕ_3 represent the angles for the end effector, the target and the tool, respectively. d_1, d_2 and d_3 indicate the distances to the end effector, the target, and the tool, respectively. (b) The limb consists of two joints θ_1 and θ_2, and the arm segments $L1$ and $L2$ (with the length ratio of $L1 : L2 = 1 : 1.25$). The two joint angles θ_1 and θ_2 are controlled by the neural circuit output. This figure is adapted from [11]

RAC are as follows:

$$AC_{end\_eff} = (\phi_1, d_1) \tag{1}$$
$$AC_{target} = (\phi_2, d_2) \tag{2}$$
$$AC_{tool} = (\phi_3, d_3) \tag{3}$$
$$RAC_{end\_eff\&target} = (\phi_2 - \phi_1, d_2 - d_1) \tag{4}$$
$$RAC_{end\_eff\&tool} = (\phi_3 - \phi_1, d_3 - d_1) \tag{5}$$

The input layer includes both perceptual and proprioceptive neurons [15]. 6 input neurons are needed: two joint angles θ_1 and θ_2, relative distance and angle to the target $\phi_2 - \phi_1$ and $d_2 - d_1$, relative distance and angle to the tool $\phi_3 - \phi_1$, $d_3 - d_1$. (Fig. 2a)

The output layer serves as motor neurons to control the change of two joint angles $\Delta\theta_1$ and $\Delta\theta_2$ with maximum step size of $\pm 1.5°$ (Fig. 2b).

We also use joint limit detectors as two additional input neurons:

$$\vartheta_{\{\theta_1,\theta_2\}} = \begin{cases} 1 & \text{if } \{\theta_1, \theta_2\} \geq 150° \\ -1 & \text{if } \{\theta_1, \theta_2\} \leq 150° \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

Overall, 8 input neurons and 2 output neurons are used in the neural circuit, and these are the only neurons in the initial network (i.e., no hidden neurons).

3) Fitness Criteria: We tested three basic fitness criteria and their different combinations to evolve the neural circuit controller. For each controller, by the end of 100 trials, the following quantities were calculated: (1) D: distance between the end effector and the target; (2) S: steps taken to reach the

target; and (3) T: tool pick-up frequency.

$$D = 1 - \frac{\sum_k \|\overrightarrow{o_k} - \overrightarrow{e_k}\|}{K \times D_{max}}, \tag{7}$$

$$S = 1 - \frac{\sum_k s_k}{K \times S_{max}}, \tag{8}$$

$$T = \frac{\sum_k t_k}{K}, t_k = \begin{cases} 1 & \text{if tool was picked up} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where $\overrightarrow{o_k}$ and $\overrightarrow{e_k}$ are the coordinates of the target object and the end effector of the limb, respectively. The Euclidean distance $\|\cdot\|$ is normalized by the maximum radius of the environment D_{max}. K indicates the total number of the trials ($K = 100$ in the experiment) and k indicates kth trial. s_k indicates the number of steps taken before reaching the target, and S_{max} is the maximum movement steps for each trial ($S_{max} = 500$ in the experiment). $t_k = 1$ when the tool is picked up during the trial and 0 otherwise. Different combinations of the fitness criteria elements are tested: D, S, DS, DT, ST, S^2T and DST. Multiplication ("$\times$") is used when combining the fitness criteria elements (e.g., DS means $D \times S$).

III. Experiments and Results

A. Behavior Analysis

1) Fitness Criteria and Success Rate: When measuring the performance of the different fitness criteria, comparing the raw directly is unfair because different fitness criteria produces different scale of fitness scores. So we simply used the success rate (the percentage of successful target reach events during a fixed number of trials) to compare the performance across different criteria. The results (mean success rate) were as follows [11]: D (17.78%), S (31.66%), DS (28.65%) for the fitness conditions lacking T (group $\{D, S, DS\}$); and DT (93.70%), ST (90.47%), SDT (94.07%) for the condition with T (group $\{DT, ST, DST\}$). The success rates appeared similar within each group (t-test, $n = 4$, $p > 0.1$ for all cases) but significantly different across the two groups (t-test, $n = 4$, $p < 0.01$ for all cases). The term T (tool pick-up frequency) turns out to be important.

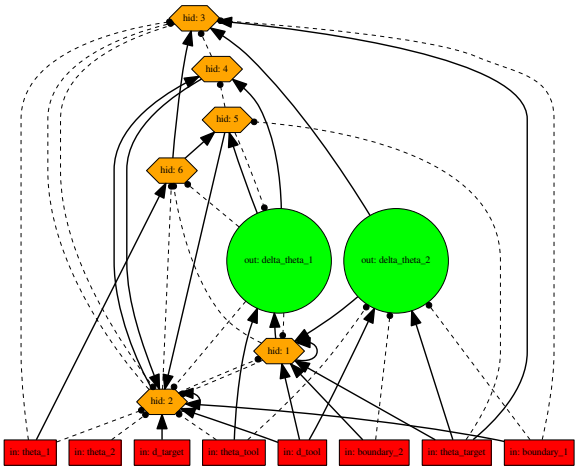


Fig. 3. An example of the neural circuit controller. This is the connectivity graph of one the best controllers (success rate = 80.68%) in the evolution with NEAT algorithm. This controller emerged after 116 out of 120 generations. 8 input neurons (rectangles) and 2 output neurons (circles) are adapted from the original architecture of the neural circuit (before evolution) and 6 hidden neurons (hexagons) are the evolved ones. Dashed lines indicate connections with negative weights. The neuron level analysis is mainly based on this circuit because this controller has relatively simple topology compared with the other best controllers.

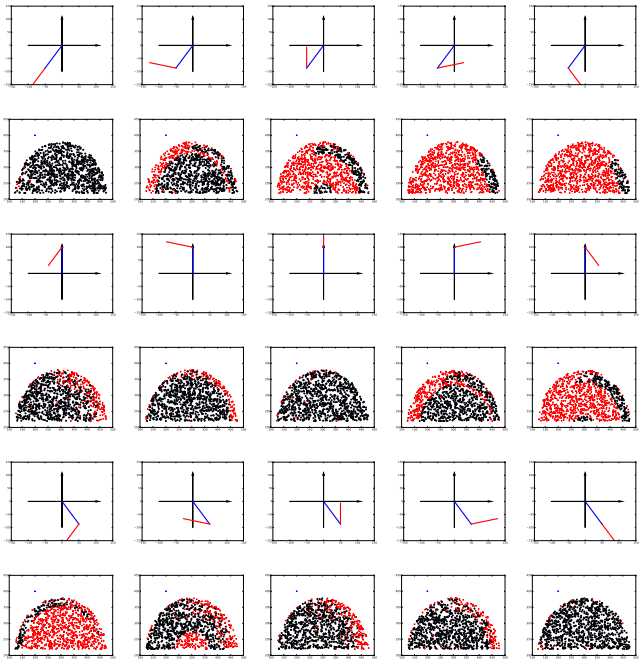


Fig. 4. Results of evaluation with different initial trial conditons. Rows 1,3,5: the starting angles of the joints. Rows 2,4,6: the corresponding random sampling of different legal tool locations (all within the reach of the arm). Each dot indicates a tool location in a trial. Dark dots indicate successful trials, and red dots indicate failed trials, respectively. The distribution changes gradually in each row as the starting angles of the arm joints change.

2) Success Rate and Initial Trial Configurations: Given the initial configuration of the trial (location of the tool, target and the starting angles of the arm), the result of the trial is deterministic. We evaluated the neural circuit controller shown in Fig.3 with different initial configurations. To investigate the correlation between the success rate and each variable, we controlled the number of variables. As shown in Fig. 4, the correlation between success/fail distribution and starting angles of the arm joints is significant.

We also noticed that in Fig. 4, there are many areas with mixed outcomes. We further investigated these areas (Fig. 5a). The arm shows some interesting behavior (Fig. 5b): very similar initial trial configurations (same target location, same starting angles of arm joints, adjacent tool locations) can lead to totally different results. Thus, the results are deterministic

but chaotic.

3) Strategy Analysis: The strategy of the neural circuit controller can be evaluated from the activities of the input neurons since they contain relative agent-centered sensory representations of the environment that are changed as a result of the controller's action. We recorded the activities of 4 input neurons: relative distance and angle to the target, relative distance and angle to the tool (Fig. 6a). The convergence speed of these 4 variables can clearly reveal the strategy of the controller.

The relative angles to the tool converge to 0 first, then the relative distances to the tool converge to 0, finally the relative angles and distances to the target converge to 0. So the strategy can be summarized as:

1) Minimize the relative angle to the tool,
2) Minimize the relative distance to the tool and reach the tool,
3) Reach the target.

This method is also used on failed trials (Fig. 6b) which indicates this strategy does not always work well. So we have to further investigate the internal state of the controller not only when the arm reaches the target, but also when the arm gets stuck and fails to reach the target.

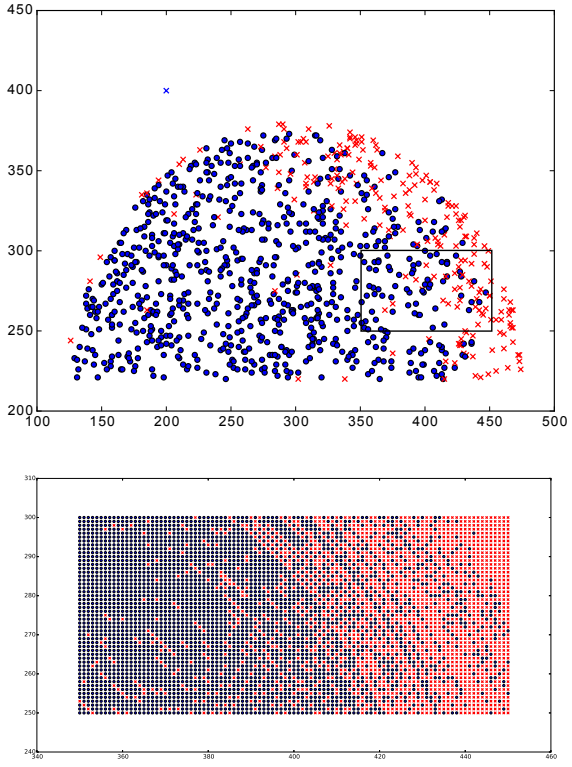
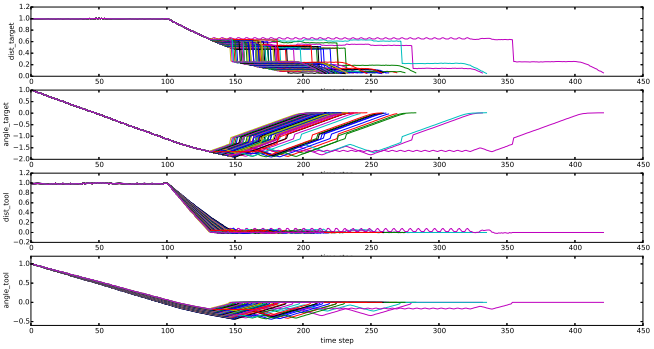
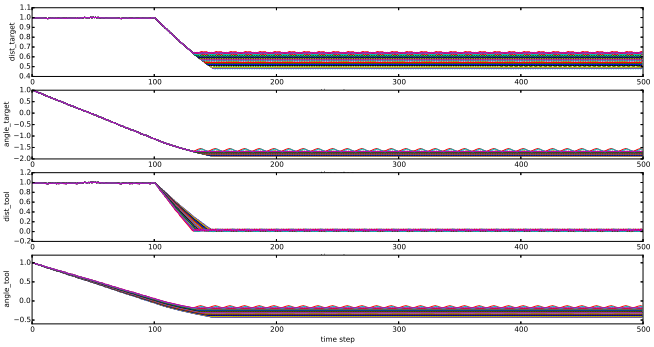


Fig. 5. Success/fail distribution with different tool locations. Each dot indicates a tool location in a trial. Dark and red dots indicate successful and failed trials, respectively. Above: Magnified figure of the 4th row, 1st figure in Fig. 4. There are mixed color (mixed outcome) areas, for example, $x = [350, 450], y = [250, 300]$ (marked with the rectangle). Below: Enumeration of different tool locations with integer coordinates in the marked region $x = [350, 450], y = [250, 300]$ of the above figure. The two colors are not separable, which indicates the chaotic behavior of the arm.



(a) Temporal activity of 4 input neurons in successful trials



(b) Temporal activity of 4 input neurons in failed trials

Fig. 6. Analysis of task strategy. From top to bottom: relative distance and angle to the target, relative distance and angle to the tool. The values are normalized (divided by initial value when $t = 0$). Each plot shows multiple curves from multiple trials. (a) The 4 variables have different convergence speed: relative distance to the tool is the fastest (row 3), then the relative angle to the tool (row 4), finally the relative distance to the target (row 1) and the relative angle to the target (row 2). (b) All the failed trials fail to minimize the relative angle to the tool which means they get stuck before reaching the tool, then the activities turn out to be low amplitude oscillation.

B. Neuronal Activation Analysis

We recorded the activity of all the neurons in each trial (Fig. 7), aiming to find internal evidence why the arm shows such complex behavior. Since the activation function of the hidden neurons is a sigmoid function $\sigma(x) = \frac{1}{1+e^{-\gamma x}}$ which is non-linear, the recurrent neural circuit can be interpreted as a non-linear dynamical system. We investigated the local stability of this dynamical system.

We are interested in stability analysis near the end states of both successful and failed trials. More formally, the state function of the hidden neurons is

$$x_{i(t+1)} = \frac{1}{1 + e^{-\sum_j \gamma w_{ji} x_{j(t)}}} \tag{10}$$

where x_i is the i-th hidden neuron and x_j can be any type of neuron that connects to x_i with connection weight w_{ji}. It

indicates the state of the hidden neurons at the current time step depends on the state at the previous time step. The change of the state is

$$\Delta x_{i(t+1)} = x_{i(t+1)} - x_{i(t)} \tag{11}$$

$$= \frac{1}{1 + e^{-\sum_j \gamma w_{ji} x_{j(t)}}} - x_{i(t)} \tag{12}$$

To test if there is a fixed point of the hidden neurons' activity state when the arm reaches the target, we need to know the values of the input and the output neurons. Since the location of the target is given as initial configuration, the angles of the arm joints when the arm reaches the target can be algebraically solved, therefore all the values of input neurons are known. The output neurons at steady state should satisfy $\Delta\theta_1 = 0$, $\Delta\theta_2 = 0$ and can also be solved directly. Therefore, all the values of input and output neurons are known.

The set of non-linear equations

$$\Delta\overrightarrow{X_{(t)}} = 0 \tag{13}$$

is always solvable with numerical methods and the solutions (often not unique), given the values of the input and the output neurons, are fixed points. To analyze the local stability of the fixed points, we need to solve the Jacobian matrix of Eq.11 since it is differentiable:

$$J = [j_{ij}] - I \tag{14}$$

$$\text{where } j_{ij} = \frac{\partial x_{i(t+1)}}{\partial x_{j(t)}} = w_{ji} x_{i(t+1)}(1 - x_{i(t+1)}) \tag{15}$$

The Jacobian matrix is sparse since the network is not fully connected. In the successful trials of our experiment, all the eigenvalues of the Jacobian matrix with the solution of Eq.13 turn out to be real and negative, which indicates that the fixed points are stable. The fixed point is also an attractor for all the successful trials (Fig. 8).

An example of the Jacobian matrix is (using the fixed point p in Fig. 8):

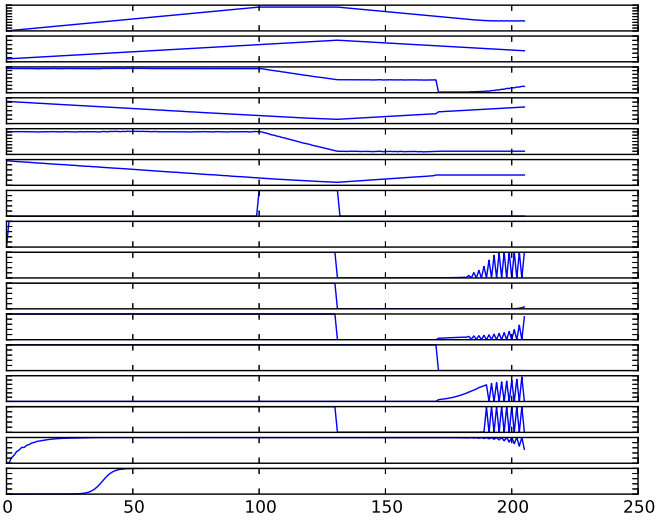
$$p = \begin{pmatrix} 1.0000 \\ 2.7098\text{E-}11 \\ 4.3269\text{E-}3 \\ 8.1991\text{E-}1 \\ 9.9951\text{E-}1 \\ 9.7924\text{E-}3 \end{pmatrix}^T \tag{16}$$

$$J = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0.0007 & 0 & 0 \\ 0 & 0 & 0 & -1 & -0.5516 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 \\ -0.0022 & 0 & 0 & 0 & 0 & -1 \end{pmatrix} \tag{17}$$

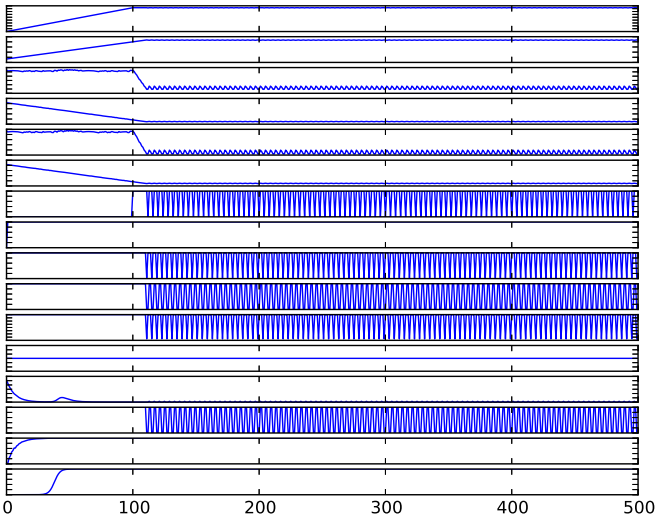
The eigenvalues were $[-1, -1, -1, -1, -1, -1]$.

In the failed trials, the arm falls into low amplitude oscillation and cannot reach the target. The oscillation is generally caused by the back and forth output $\Delta\theta_1$ and $\Delta\theta_2$, and the output oscillation is cause by the back and forth activity of the hidden neurons (Fig. 7b). Apparently in the failed trials, there is no clear fixed point attractor for the state of hidden neurons. We tried to investigate if there are limit cycles for the state of hidden neurons. All the failed trials (1912 out of 5000 trials) in the bottom part of Fig. 5 were tested.

As shown in Table 1, 97.59% of the failed trials can be categorized as periodic orbits with period $\leq$16. The period-2



(a) Example neuronal activity of a successful trial



(b) Example neuronal activity of a failed trial

Fig. 7. Analysis of neuronal activity. From top to bottom in each graph: 8 input neurons, 2 output neurons, and 6 hidden neurons (see Fig. 3). (a) The trial succeeded in about 210 time steps. The sudden switch of values at about 130 time step indicates the pick-up of the tool. (b) In this failed trial, the hidden neurons started oscillation after about 120 time steps.

TABLE I
DISTRIBUTION OF DIFFERENT TYPES OF LIMIT CYCLES IN FAILED TRIALS

Type	Number of trials	Percentage
Period-2 orbit	1693	88.55%
Period-4 orbit	102	5.33%
Period-6 orbit	23	1.20%
Period-8 orbit	11	0.58%
Period-12 orbit	23	1.20%
Period-16 orbit	14	0.73%
Other	46	2.41%
Total	1912	100.00%

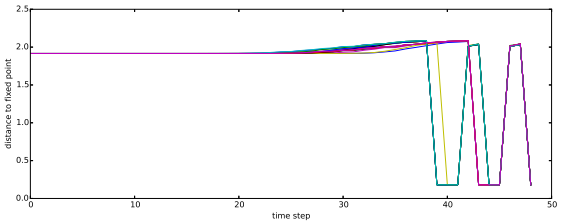


Fig. 8. The Euclidean distances from the state vector of the hidden neurons (each curve represents a successful trial) to the theoretical fixed point. The data is captured from the last 50 time steps of successful trials in our experiment. The distance is relatively large ($\geq$ 1.9) before the last 10 time steps, and after a short period of oscillation, the distances in all the trials decrease to very small values ($\leq$ 0.18), indicating the end state is close to the theoretical fixed point.

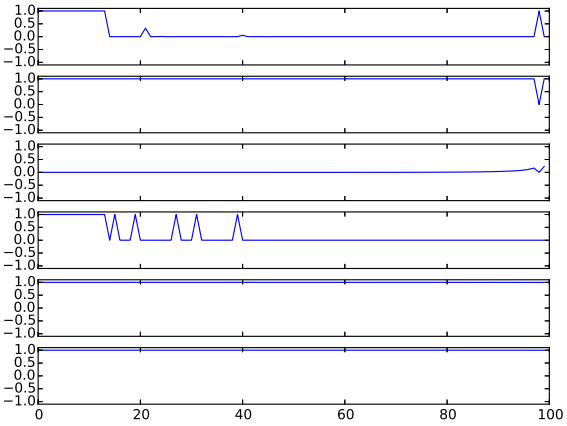


Fig. 9. An example of chaotic hidden neuron activity in a failed trial. The activity of the last 100 time steps of 6 hidden neurons were recorded. They do not show any periodic behavior.

orbits (88.55%) are the dominating type of limit cycles. The trials in "other" category are either chaotic or periodic orbits with long (>16) periods (Fig. 9). However, we cannot prove if the periodic orbits are attractors because there is still not a good method to estimate the number of the periodic orbits and their topological properties.

IV. DISCUSSION

At present, not only in our previous experiment, but also in many complex control problems using neuroevolution [16], fitness criteria and success rate estimation are mainly done by simple random sampling of initial task conditions. However, simple random sampling often ignores the variation in behaviors and internal dynamics across different subpopulations. Stratified sampling we used in this paper can be more helpful in finding the difference of behavior among subpopulations across trials (Fig. 5). The overall task performance can be evaluated from the performance of many sub-tasks, but not vice versa.

Though global stability analysis is not practical for a high-dimensional non-linear dynamical system [12] like the one in our experiment, to some extent, local dynamical analysis can still explain why and how a trial can succeed or fail. The target in the tool-use task puts an attractor in the state space of hidden neurons. However, some trials may step into limit cycles so that the arm fails to reach the target. We hope our study can provide new performance criteria (e.g. the comparison of performance among subpopulations with different configurations, the stability of the goal state, etc.) in the evaluation of evolved RNNs.

The limitations of the study in this paper are as follows: most world properties were heavily encoded in the inputs themselves, e.g., target location, tool location, etc; our method still cannot predict when will the state of hidden neurons get trapped in limit cycles.

To improve the neural circuit controller, object recognition and grounding [17][18], and efficient codes for motor encoding [19] can be incorporated into our system for increased realism. We can also integrate our behavior analysis with novelty search [20]. Since the behavior analysis is independent of the fitness function, it is possible to systematically build up the behavior space based on behavior analysis. Lesion study [21] on the internal dynamics of the RNNs can also be used. This will help understand thoroughly the functionality of each component in the network and to improve the robustness of the RNNs.

V. CONCLUSION

In this paper, we analyzed the behavior and the neuronal activation of the RNN which was used in our pervious paper as an articulated limb controller in a tool use task, and showed the correlation between the behavior of the limb and the internal dynamics of the RNN. We also investigated the local stability of the end states in both successful (goal state reached) and failed trials in our experiment. The existence of fixed points and limit cycles could explain the cause of success and failure. We expect our results to help better understand RNNs and their dynamics. .

ACKNOWLEDGMENT

The simulated neural circuit evolution experiments were conducted using ANJI (Another NEAT Java Implementation.

anji.sourceforge.net), an open-source Java implementation of the algorithm by Stanley and Miikkulainen [12].

REFERENCES

[1] J. C. Latombe, *Robot motion planning*. vol. 124. Springer Science & Business Media, 2012.

[2] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE*, 2013.

[3] R. D. Beer, "A dynamical systems perspective on agent-environment interaction," *Artificial Intelligence*, vol. 72, issue 1-2, pp. 173-215, 1995.

[4] R. D. Beer, "On the dynamics of small continuous-time recurrent neural networks," *Adaptive Behavior*, 3 (4), pp.469-509.

[5] Y. Ilyashenko, "Centennial history of Hilbert's 16th problem", *Bulletin of the American Mathematical Society*, vol. 39, no. 3, pp. 301-354.

[6] A. Whiten, J. Goodall, W. C. McGrew, T. Nishida, V. Reynolds, Y. Sugiyama, C. E. Tutin, R. W. Wrangham, and C. Boesch, "Cultures in chimpanzees," *Nature*, vol. 399, no. 6737, pp. 682-685, 1999.

[7] C .Boesch and H. Boesch, "Tool use and tool making in whild chimpanzees", *Folia primatologica*, vol.54, no. 1-2, pp. 86-99, 1990.

[8] P. Foerder, M. Galloway, T. Batthel, D. E.Moore III, and D. Reiss, "Insightful problem solving in an asian elephant," *Plos one*, vol. 6, no. 8, p. e23251, 2011.

[9] J. Boswall, "Tool-using and related behavior in birds: more notes," *Avicultural Magazine*, vol. 89, no. 2, pp. 94-108, 1983.

[10] J. R. Chung and Y. Choe, "Emergence of memory in reactive agents equipped with environmental markers," *Autonomous Mental Development, IEEE Transactions on*, vol. 3, no. 3, pp. 257-271, 2011.

[11] Q. Li, J. Yoo, Y. Choe, "Emergence of Tool Use in an Articulated Limb Controlled by Evolved Neural Circuits", in Proceedings of the International Joint Conference on Neural Networks, 2015. DOI: 10.1109/IJCNN.2015.7280564.

[12] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99-127, 2002.

[13] S. D. Whitehead and D. H. Ballard, "Learning to perceive and act by trial and error," *Machine Learning*, vol. 7, no. 1, pp. 45-83, 1991.

[14] T. A. Mann and Y. Choe, "Prenatal to postnatal, transfer of motor skills through motor-compatible sensory representations," in *Development and Learning (ICDL), 2010 IEEE 9th International Conference on.* IEEE, 2010, pp. 185-190.

[15] R. Murphy, *Introduction to AI robotics*, MIT press, 2000.

[16] F. J. Gomez, "Robust non-linear control through neuroevolution", PhD Thesis, Department of Computer Sciences, The University of Texas at Austin.

[17] C. Yu and D. H. Ballard, "On the integration of grounding language and learning objects," in *Nineteeth AAAI Conference on Artificial Intelligence*, vol. 4, 2004, pp. 488-493

[18] J. Modayil and B. Kuipers, "Autonomous development of a grounded object ontology by a learning robot," in *Twenty-Second AAAI Conference on Artificial Intelligence*, vol. 22, no. 2, 2007, p. 1905.

[19] L. Johnson and D. H. Ballard, "Efficient codes for inverse dynamics during walking," in *Twenty-Eighth AAAI Conference on Artificial Intelligence*, pp. 343-349, 2014.

[20] J. Lehman, and K. O. Stanley, "Abandoning objectives: Evolution through the search for novelty alone." *Evolutionary computation* vol. 19, no. 2, pp. 189-223, 2011.

[21] R. D. Beer, et al. "A distributed neural network architecture for hexapod robot locomotion." *Neural Computation* vol. 4, no. 3, pp. 356-365, 1992.