Emergence of Tool Use in an Articulated Limb Controlled by Evolved Neural Circuits

Qinbo Li
Department of Computer Science and Engineering
Texas A&M University
Email: lee@tamu.edu

Jaewook Yoo
Department of Computer Science and Engineering
Texas A&M University
Email: jwookyoo@tamu.edu

Yoonsuck Choe
Department of Computer Science and Engineering
Texas A&M University
Email: choe@tamu.edu

***Abstract*—Tool use requires high levels of cognitive function and is only observed in higher mammals and some avian species such as corvids. In this paper, we will investigate how the capability to use tools can spontaneously emerge in a simulated evolution of a two degree-of-freedom articulated limb. The controller for the limb was evolved as neural circuits that can gradually take on arbitrary topologies (NeuroEvolution of Augmenting Topologies, or NEAT). First, we show how very broad fitness criteria such as distance to the target, number of steps to reach the target, and tool pick-up frequency are enough to give rise to tool using behavior. Second, we analyze the evolved neural circuits to find properties that enable tool use. We expect our results to help us understand the origin of tool use and the kind of neural circuits that enabled such a powerful trait.**

I. Introduction

A. Background

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. Humans have used tools for at least two million years, while primates like chimpanzees use primitive tools such as stones or sticks [1], [2]. Elephants also use sticks or other objects to obtain food located out of normal reach [3]. Birds also use various tools to scratch their body [4] or even create tools to reach otherwise unreachable food [5].

Tool use requires detailed understanding of the causal physical properties of the environment, curiosity, nontrivial planning, and learning through trial-and-error. For example, Streri and Féron [6] showed in a haptics study that infants develop and generalize their knowledge of tool use through trial-and-error. The neural processes that form the basis of tool use has also been explored, where changes (extension) in visuo-somatosensory receptive fields have been observed when a tool was held and used [7].

Given the apparent complexity of the underlying mechanism, it is unclear how such a capability could have emerged from simple organisms. Chung and Choe [8] 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. In this paper, we will extend this simple tool to a more substantial one (a stick), for reaching unreachable objects.

B. Related Work

Tool use has been extensively studied in artificial intelligence and robotics (for a review, see [9]). The various existing works can be broadly grouped into four categories:

1) Programmed, hard-coded behavior [10];
2) Learning through demonstration [11], [12], [13], [14], [15];
3) Learning via random trial-and-error [16], [17], [18]; and
4) Evolved tool using behavior [19], [8]. (Cf. Sims [20] where body morphology [not tools] was co-evolved with the controller.)

However, most of the works listed above depended on some degree of designer knowledge regarding tool use and motor control, e.g., fully hard coded behavior, the tool being pre-attached to the limb, pre-defined tool features, pre-defined motor primitives, etc.

Evolution-based approaches [19], [8] were relatively free of these constraints, but in those cases the tools were more or less simple markers, not something than can be manipulated with a limb-like structure of the agent.

C. Approach

In this paper, we investigate how the capability to use tools can spontaneously emerge in an evolved neural circuit controller for a two degree-of-freedom articulated limb. The goals of the study were to find minimal and indirect fitness criteria for the emergence of tool use, and to analyze the properties of evolved neural circuits that permit tool use. We evolved the neural circuit controller by gradually augmenting the network topology (NeuroEvolution of Augmenting Topologies [NEAT], by Stanley and Miikkulainen [21]). The environment consisted of a two degree-of-freedom articulated limb, a target object, and a tool. The task was to reach a target object that may or may not be within reach of the limb, with or without the tool. Different fitness criteria were tested to investigate which one is good enough to give rise to tool using behavior for the limb. We define three basic fitness criteria and tested their different combinations to evolve the neural circuit for the controller for the limb. We avoided using direct heuristics such as predefined motions or programmed guidance. Our results indicate that simple, indirect criteria such as distance to target, number of steps to reach target, and if the tool was fetched was enough to give rise to tool using behavior. Quantification of

recurrent loops in the neural circuit topology showed that better controllers have more such loops. Furthermore, if recurrent loops are prohibited, the task performance greatly degraded and the evolved circuit had much more neurons than their recurrent circuit counterpart.

The rest of the paper is organized as follows. We will first describe our approach and method in Section II. Details of the experiments are described in Section III. Results are reported in Section IV, followed by discussion and future work in Section V and conclusion in Section VI.

II. METHODS

We evolved neural circuits of arbitrary topology to control a two-limbed articulated arm in an object reaching task. The environment was equipped with a tool (a stick) that can be picked up and used to get to objects beyond the reach of the limb.

A. Algorithm for Evolving Neural Circuits

Since tool use is a complex behavior that may not be easy to evolve using standard neuroevolution methods, we used the NeuroEvolution of Augmenting Topologies (NEAT) algorithm that allows arbitrary network topologies to be evolved [21]. NEAT is based on the idea of complexification, where behavioral complexity is achieved through complexifying neural circuit topology. NEAT has been used in various learning environment such as games and has been shown to be effective in evolving complex, non-trivial behavior [22], [23].

In NEAT, the chromosome encodes neurons and their connections separately, as well as the connection weights. Neurons and connections can be added or removed to change the network topology, thus the chromosome has a variable length. Mating of chromosomes with different network topology is achieved through the use of a quantity called "innovation number" that indicates the evolutionary origin of a particular gene. Innovation numbers allow only compatible genes to mate (i.e., genes that have the same ancestral origin). Another unique mechanism of NEAT is "speciation" that helps freshly changed topology to be preserved despite the initial plunge in fitness. The rest of the algorithm is similar to other neuroevolution or evolutionary algorithms: instantiation of phenotype from genotype $\rightarrow$ testing in the task environment $\rightarrow$ calculating fitness $\rightarrow$ selection and reproduction.

B. Fitness Criteria

Our aim in selecting the fitness criteria was to find measures that do not directly dictate the evolved tool-use behavior (i.e., minimal and indirect). We used three basic fitness criteria and tested different combinations of them to evolve the neural circuit controller for the limb. For each controller, by the end of 100 trials, the following quantities were calculated: (1) D: distance between the end effector and the target (Eq. 1); (2) S: steps taken to reach the target (Eq. 2); and (3) T: tool pick-up frequency (Eq. 3).

$$D = 1 - \frac{\sum_k \|\vec{o_k} - \vec{e_k}\|}{K \times D_{max}}, \quad (1)$$

$$S = 1 - \frac{\sum_k s_k}{K \times S_{max}}, \quad (2)$$

$$T = \frac{\sum_k t_k}{K}, \quad t_k = \begin{cases} 1 & \text{if tool was picked up} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where $\vec{o_k}$ and $\vec{e_k}$ 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 k_{th} 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 is 1 when the tool was picked up during the trial and 0 otherwise. Different combinations of the fitness criteria elements were tested: D, S, DS, DT, ST, and DST. Multiplication ("$\times$") was used when combining the fitness criteria elements (e.g., DS means $D \times S$). In addition, neural circuits evolved with and without recurrent connections were compared ($SST = S^2T$ and $SST_{noRecur} = S^2T_{noRecur}$).

C. Input and Output Interface for the Neural Circuit

It is important to define the right sensory inputs to represent the environment properly, especially for artificial agents learning through action and feedback loops [24]. It is widely believed that we encode the space around us in a body-centered coordinate system. In [25], the authors compared two forms of representations: world-centered sensory representation (WC) and agent-centered sensory representation (AC). For example, AC can use polar coordinates while WC can use Cartesian coordinates. The authors further proposed the relative agent-centered sensory representation (RAC), which is the relative difference between two ACs.

In our experiment, we used RAC as the sensory inputs to the limb controller neural circuit. To be specific, the RAC between end effector and target, and the RAC between end effector and the tool were used. The details for AC (Fig. 1) and RAC are as follows.

$$AC_{end\_eff} = (\varphi_1, d_1) \quad (4)$$

$$AC_{target} = (\varphi_2, d_2) \quad (5)$$

$$AC_{tool} = (\varphi_3, d_3) \quad (6)$$

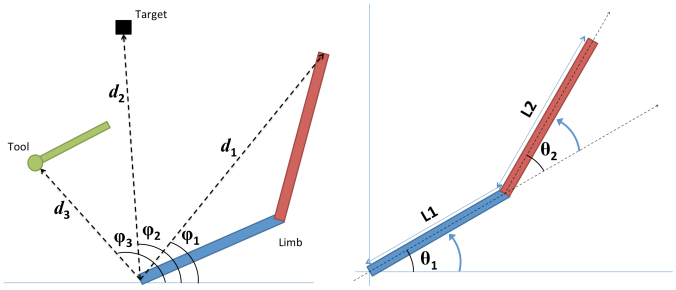
$$RAC_{end\_eff\&target} = (\varphi_2 - \varphi_1, d_2 - d_1) \quad (7)$$

$$RAC_{end\_eff\&tool} = (\varphi_3 - \varphi_1, d_3 - d_1) \quad (8)$$

Other inputs for the limb controller were the joint limit detectors (ϑ_1, ϑ_2) that can sense whether the joints reach the limitations as follows. The last two inputs were the current angle of the two joints (θ_1, θ_2).

$$\vartheta_{\{\theta_1,\theta_2\}} = \begin{cases} 1 & \text{if } \{\theta_1, \theta_2\} \geq 150^{\text{o}} \\ 1 & \text{if } \{\theta_1, \theta_2\} \leq -150^{\text{o}} \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

In a nutshell, each neural circuit consists of eight inputs, two outputs, zero or more hidden neurons, and positive (excitatory) and negative (inhibitory) connections (Fig. 2). The



(a) Sensory Representation (b) Kinematics

Fig. 1. 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 J1 (θ_1) : J2 (θ_2), and the arm segments $L1 : L2$ (with the length ratio of $L1 : L2 = 1 : 1.25$). The two joint angles $\theta_1 : \theta_2$ are controlled by the neural circuit output.

eight sensory inputs are RAC between the end effector and the target, RAC between the end effector and the tool, the two joints angles ($\theta 1$, $\theta 2$), and two limit detectors for $\theta 1$ and $\theta 2$. The input values are normalized between 0 and 2 except the d_1, and d_2 for RAC.

III. Experiment (Simulation)

The task environment consisted of a two degree-of-freedom articulated limb, a target object, and a tool (Fig. 2(b) and Fig. 3). The articulated limb moves on the 2D plane by changing θ_1 and θ_2. The task is to reach the target with or without the tool. The details about the limb, the task, and the experimental procedure are described below.

A. Reaching Task

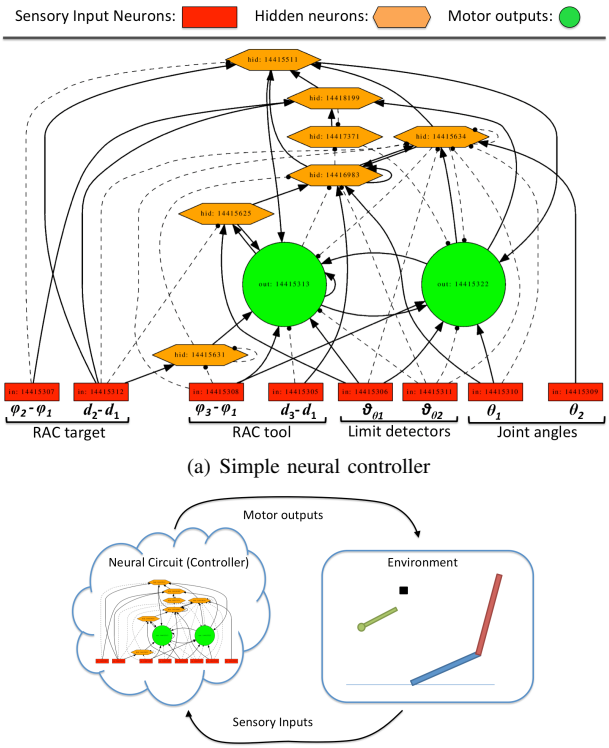
The task is to evolve a neural circuit controller to reach targets with or without the tool. Amant and Wood [9] argued that animals such as chimpanzees, elephants, and parrots have some level of intelligence that can be measured, and similar experiments and metrics can be applied to artificial agents. In [7], a task for reaching a distant object using a tool (stick) was used to observe tool use in monkeys. They found that the body schema changes in the monkey's brain when the tool was in use. The task of reaching a distant object using a tool can be categorized as simple tool use, according to the "tooling test" taxonomy proposed by Amant and Wood [9].

B. Articulated Limb

The articulated limb consisted of two joints ($\theta_1$and θ_2), and two links ($L1$ and $L2$) (Fig. 1). The limb was able to move on the 2D plane by changing θ_1 and θ_2. It can move the two joints left or right. At each time step, the velocity limit was from -1.5° to 1.5°. The two outputs of the neural circuit (the limb controller) were connected to the two joints to move the limb. The ranges of the two joints were from -150° to 150°.

C. Experimental (Simulation) Procedure

Each two degree-of-freedom articulated limb was given 100 trials to perform the target reaching task. In each trial, the limb was allowed to move up to 500 time steps (maximum



(a) Simple neural controller

(b) Controller-environment interaction

Fig. 2. Neural circuit controller and the jointed limb. (a) A neural circuit controller consists of eight sensory inputs, two motor outputs, and zero or more hidden neurons. The sensory inputs are RAC between the end effector and the target, RAC between the end effector and the tool, two joints angles ($\theta 1$, $\theta 2$), and two limit detectors for $\theta 1$ and $\theta 2$. The motor outputs are for the two joint angles. The neurons are connected with excitatory (solid) and inhibitory connections (dashed) including recurrent connections. (b) The interaction cycle between the neural circuit controller and the environment is shown. 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 was not explicitly notified of the limb extension due to tool pick-up.

(a) Target out of reach (b) Target within reach

Fig. 3. Example task conditions. The environment consists of 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).

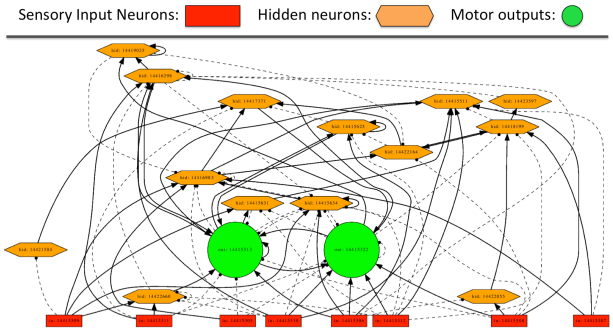
time step). For each trial, the target and the tool were placed at random locations. The distance of the target could be either within the reach of the limb or not (the tool was always within the reach of the limb; see Fig. 3). In the latter case, the limb must use the tool to reach the target (otherwise it will fail). If the limb successfully reached the target, the current trial ended and the number of time steps taken was recorded. If the agent failed to reach the target, the distance from the end effector to the target was recorded. The fitness of the agent was calculated based on the sum of final distances (Eq. 1), the total time steps (Eq. 2), and the number of times the tool

was picked up (Eq. 3), as described in the Methods section. Different combinations of the fitness criteria elements were tested as described in Section II.

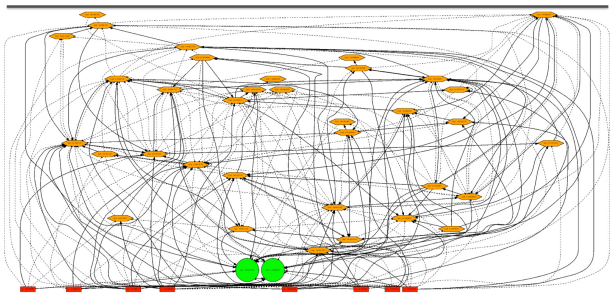
We evolved the the neural circuit (the limb controller) for 120 generations with a population size of 100. To evaluate the performance, after the 120 generations were done, we picked the best limb controller and repeated 10 times a set of 100 test trials (total 1,000 trials). With this, we compared the performance and the trends throughout the generations, as well as the network topology characteristics. For all experiments, the within-reach and out-of-reach conditions were balanced (50% each).

IV. Results

A. Evolved Neural Circuits and Observed Behavior



(a) Evolved neural circuit with fitness=S^2T.



(b) Evolved neural circuit with fitness=DS.

Fig. 4. Examples of evolved neural circuits. (a) Evolved neural circuit with fitness criterion S^2T (speed squared and tool pick-up frequency): 24 neurons (8 sensory inputs, 2 motor outputs, and 14 hidden neurons) and 90 connections (49 excitatory and 41 inhibitory connections; 10 are recurrent). (b) Evolved neural circuit with fitness criterion DS (distance and speed): 45 neurons (8 sensory inputs, 2 motor outputs, and 35 hidden neurons) and 252 connections (124 excitatory and 128 inhibitory connections; 10 are recurrent).

Fig. 4 shows the network topology of evolved circuits under different fitness criteria. The one that included tool pickup frequency (T) in its fitness showed a much simpler network while maintaining the same level of performance. In Fig. 5, limb movements are shown as time-lapse images. Time is encoded as the pixel intensity, where darker means more recent state. The three images in the top row show examples of successful movements. Starting position is marked as ⓢ, tool pick up event as ⓣ, and ending position as ⓔ. Target location is marked □. The limbs (blue and red lines) in the three images first move towards the tool (green line with a circle at the handle) to pick it up, and then the extended limb moves towards the target. The trajectories of the end effectors

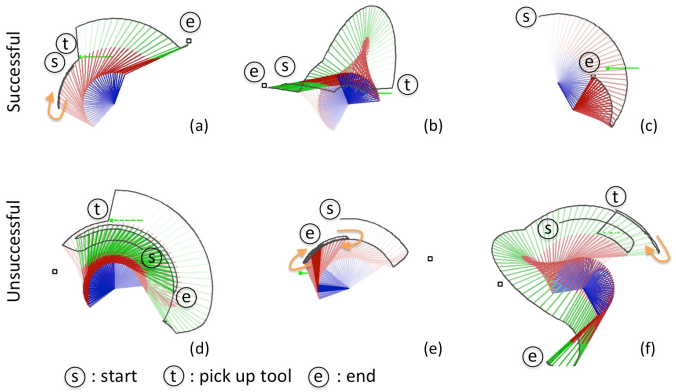


Fig. 5. Time-lapse images of representative limb movements. The three images in the top row ((a), (b), and (c)) show examples of successful movements (with fitness criteria S^2T). In the movements of (a) and (b), the limbs (blue and red lines) first move towards the tool (green horizontal line) to pick it up (i.e., the limb is extended), and then move towards the target (□). In the movement of (c), the limb moves directly toward the target without picking up the tool. The trajectories of the end effecters are shown as black curves, and parts of the trajectories that may be hard to keep track of are annotated with orange →. The three images in the bottom row ((d), (e), and (f)) show examples of unsuccessful movements (using fitness criteria S^2T without recurrent connections), where the limbs failed to reach the target.

are shown as black solid curves. In the bottom row, the three images show examples of unsuccessful movements, where the three movements of the limb could not reach the target (ⓔ is not near □).

In successful trials, the limbs slowed down toward the target, but in unsuccessful cases they moved with almost the same speed at all times. It seems that the successful neural circuits learned to move slowly when fine control is needed. Also, another interesting behavior was observed. When approaching the target after picking up the tool, the limbs first aligned the AC_{end_eff} angle (φ_1) to the AC_{target} angle (φ_2), and then reduced the difference between AC_{end_eff} distance (d_1) and AC_{target} distance (d_2), by extending the limb toward the target.

B. Success Rate for Each Fitness Criterion

When measuring the performance of the different fitness criteria, comparing the raw fitness scores is not fair because different fitness criteria produce different fitness score scales. Therefore, we used success rate to compare the performance between different criteria. Success rates are the percentage of successful target reach events during the 1,000 test trials. For each fitness criterion, we selected the best neural circuit from the last generation and then ran 1,000 trials. The entire process of evolution and testing was repeated four times ($n = 4$). The results are summarized in Fig. 6.

In general, DT, ST,and DST showed superior success rates than their counterparts that did not use the tool pick-up frequency criterion T (D, S, and DS). Within the same group (within $\{DT, ST, DST\}$ or within $\{D, S, DS\}$), the success rates were similar (t-test, $n = 4$, $p > 0.1$ in all cases). However, differences across the two groups were statistically significant (t-test, $n = 4$, $p < 0.01$ for all cases).

The fitness criteria only using D (distance) or S (number of steps) did not show good performance (around 50%), neither

did the combination DS. However, combining with T (tool pick-up frequency) boosted the performance (DT, ST, around 70%). This indicates that giving reward for simply picking up the tool, even without any further implications given, could lead to the emergence of adaptive tool use. Related observation was reported in an experimental study. Amant and Wood [9] noted that in the experiment in Visalberghi and Limongelli [26], capuchin monkeys frequently achieved tasks requiring tool-using ability, by quickly trying many inappropriate strategies with tools (e.g., pick up something and wield it without a clear plan).

In most cases, the fitness criterion S (number of steps) seems to be useful (since high S fitness also means target has been reached, early). The limbs evolved with S (such as ST and DST) reached the target directly without picking up the tool, if the distance to the target is within the limb's reach. For the fitness criteria without S but with T (such as DT), we observed that the limb mostly tried to get to the tool first even when the target is within reach.

As mentioned above, about 50% of the trials were within-reach condition and the other 50% out-of-reach. However, note that even in cases where the success rates are around 50%, the tool was used to reach out-of-reach targets. That means performance of about 50% included cases where tool was used successfully to reach out-of-reach targets, and cases where within-reach targets were not reached. For example, the success rates for the beyond-reach trials were D (17.78%), S (31.66%), and DS (28.65%) for the fitness conditions lacking T; and DT (93.70%), ST(90.47%), DST(94.07%) for the conditions with T. In sum, D/S/DS conditions did utilize the tool, although less frequently than DT/ST/DST.

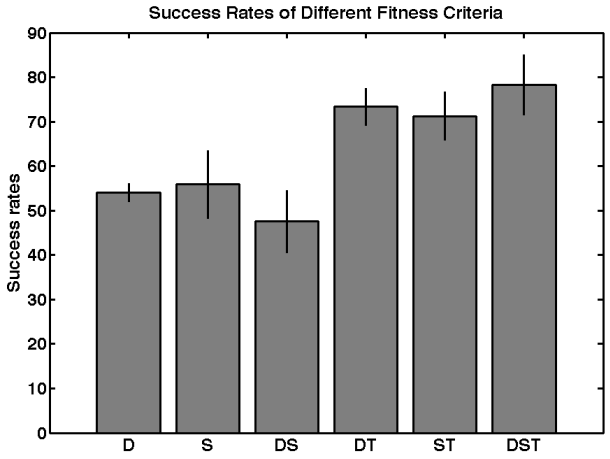


Fig. 6. Average success rates of controllers based on different fitness criteria. Four sets of experiments (evolution of the neural circuit controller followed by testing) were ran for each fitness criterion. In general, fitness criteria that included T (tool pick-up frequency) did significantly better than those that did not include T (t-test, $n = 4$, $p < 0.01$). Results with S^2T were similar (data not reported here). Note that the target was placed within the limb's reach only ∼50% of the time during the trials. See text for details.

C. Contribution of Recurrent Connections

We compared neural circuits evolved with and without recurrent connections. The S^2T fitness criterion was used to evolve the neural circuit. In the first case recurrent connections were allowed (S^2T), and in the second case only

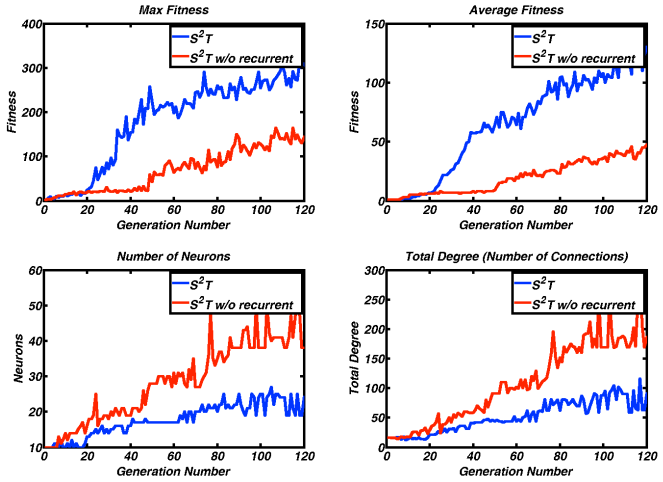


Fig. 7. Fitness over generations and network size, with or without recurrent connections. S^2T with (blue line) and without recurrent connections (red line) are compared. Top-left: max fitness scores through the generations. Top-right: average fitness scores. Bottom-left: number of total neurons. Bottom-right: number of total degrees (number of connections) in the neural circuits through the generations.

feedforward connections were allowed ($S^2T_{noRecur}$). First, we compared the success rates of the best chromosomes for the two conditions (S^2T: mean = 80.68%; $S^2T_{noRecur}$: mean = 52.58%). The difference was statistically significant (t-test, n=4, $p < 0.01$). Next, we compared the maximum fitness scores, average fitness scores, the number of total neurons, and the total degrees (the total number of connections) throughout the generations. In Fig. 7, top-left and top-right panels show the maximum fitness scores and the average fitness scores throughout the generations when evolving the neural circuits. The one with recurrent connections shows better performances throughout. However, interestingly, the number of neurons and connections in the neural circuits shows the opposite trend, as shown in the bottom-left and the bottom-right panel. The neural circuit without recurrent connections has more neurons and more connections even though the controller (neural circuit) shows significantly lower performance than the one with recurrent connections. This observation emphasizes the importance of recurrent connections in evolving neural circuit controllers with continuous action and feedback loop. Recurrent connections in neural networks can provide memory of the past. Also, recurrent connections can provide the ability to predict future internal dynamics, which is an important ability for neural circuit controllers, especially for challenging control tasks [27].

Also, we analyzed the correlation between (1) the number of recurrent connections and (2) success rates under different fitness conditions (Fig. 8). Self loop, 2-hop, and 3-hop loops were counted as recurrent connections. The six fitness criteria compared earlier were analyzed (D, S, DS, DT, ST, and DST). For each fitness criterion, the best neural circuits from each generation (total of 120 per condition, generation 1 to generation 120) were analyzed to see the correlation between the number of recurrent connections and success rates (same as before). The correlation plots show a trend that the fitness criteria with comparably high correlation coefficients (such as DT, ST, and DST, the ones with T [tool pick-up

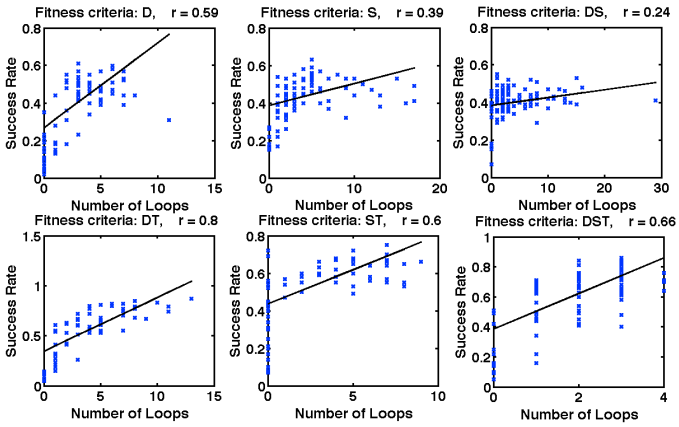


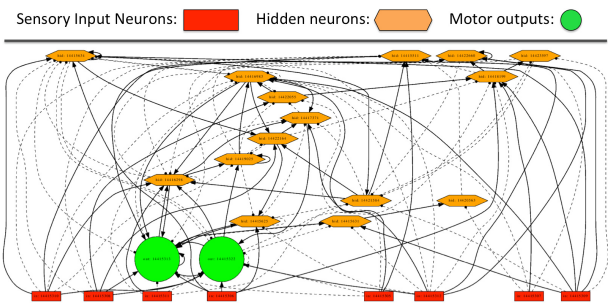
Fig. 8. Correlations between the number of recurrent connections (self loop + 2 hops + 3 hops) and success rates. The data points (blue "x"), the linear regression lines, and correlation coefficients (r) are plotted, for the six fitness criteria. For each fitness criteria, the 120 best neural circuits for each generation (generation 1 to 120) were analyzed to see the correlation between the number of recurrent connections and success rates (average of 1,000 trials each). The total number of cycles (number of loops) is found to be positively correlated with success rate. Note that the correlation is even higher for those that included T (tool pick-up frequency) in the fitness.

frequency]) have high success rates in Fig. 6. This indicates that the recurrent connections affect positively the performance of the evolved neural circuits (or vice versa). However, the number of recurrent connections themselves may not be critical for the performance as long as recurrence is allowed. Instead, how they are connected could be more important: recurrent connections that are able to predict future internal dynamics [27] could result in neural circuits with higher success rates.

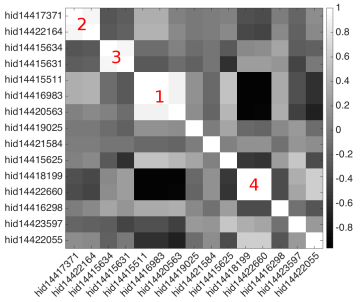
D. Internal Dynamics

Studying evolutionary autonomous agents (EAAs) has important potentials to understand structure, function, and behavior of biological neural systems [28]. Analysis of neural information processing including internal neurons dynamics in EAAs can provide insights on the function of biological nervous systems. However, even with EAA, fully understanding each neuron's role and their connection to behavior is not a trivial task, especially as the network size increases. In this section, we present some preliminary analysis of the internal dynamics of evolved neural circuits and correlate the dynamics with behavior.

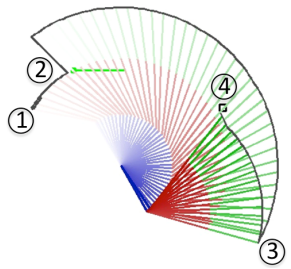
Fig. 9 shows time series values of the fifteen hidden neurons of an evolved neural circuit. In the time series correlation matrix (Fig. 9(b)), we can observe four groups of neurons that are highly correlated. The four clusters are cluster 1 (hid14415511, hid14416983, hid14420563), cluster 2 (hid14417371, hid14422164), cluster 3 (hid14415634, hid14415631), and cluster 4 (hid14418199, hid14422660). As we can see in Fig. 9(c) and (d), the groups of neurons respond (change activity) to the behavioral events ①, ②, ③, and ④. At the event ①, the limb changes the movement towards the tool, and at ③, it changes towards the target. Tool pickup happens at the event ②, and the limb reaches the target at ④. Cluster 3 responds to ① and ③, while cluster 4 to ② and ④. There are single neurons that also exhibit interesting behavior, e.g., hid14416298 (third row from the bottom of Fig. 9d showing



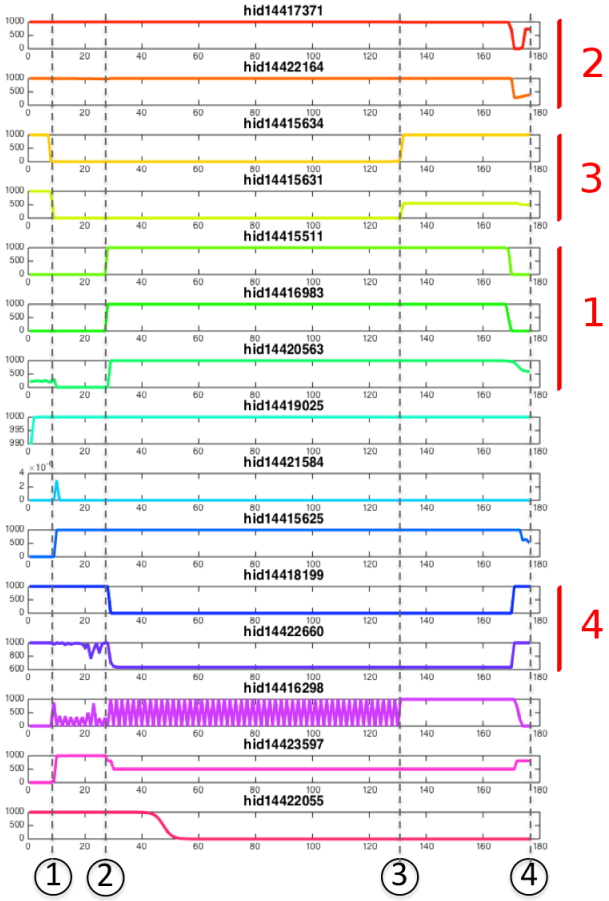
(a) Evolved neural circuit with fitness=DST.



(b) Time series correlation matrix of the fifteen hidden neurons



(c) Limb movement (time-lapse image)



(d) Time series values of the fifteen hidden neurons

Fig. 9. Internal dynamics of fifteen hidden neurons, their correlations, and matching limb movements. (a) Evolved neural circuit with fitness = DST. (b) Time series correlation matrix of the fifteen hidden neurons. (c) Time-lapse image of the limb behavior. (d) Time series values of the hidden neurons. Event in (c) are marked with dashed lines. See text for details.

rapid oscillation between events ② and ③).

V. Discussion

The main contribution of this paper is two-fold: (1) we showed that tool use behavior can be evolved with minimal, indirect fitness criteria, and (2) we found correlations between recurrent connections and tool use behavior. In most prior tool use research, the tool was either attached to the agent by default or hard-coded representations were used for the tool. In our case, the distinction between tool and target object was blurred by design, and whether the tool (or the target object) should be reached first was not dictated at all. Despite this paucity of information, our neural controller was able to successfully perform the target reaching task. Furthermore, we found that recurrent connections are key to the success (perhaps since memory of events is needed, e.g., tool picked up, then can reach the target).

The evolving neural circuit controllers also can be utilized for connectomics study. Studying the connectome - a complete neural diagram of the brain - is a challenging problem due to the complexity and scale (see [29] for a review). Synthetic connectomics can benefit natural connectomics research by developing analysis methods based on behavior analysis, internal dynamics analysis, lesion studies, and social context analysis [30]. As our preliminary analysis showed (Section IV. C and D), the synthetically evolving neural circuit controllers can be a good resource for developing such analysis methods for connectomics.

Our approach is not without limitations. One main limitation is that most world properties were heavily encoded in the inputs themselves, e.g., target location, tool location, etc. In future work, object recognition and grounding [31], [32] and efficient codes for motor encoding [33] can be incorporated into our system for increased realism.

VI. Conclusion

In this paper, we investigated how the capability to use a simple yet non-trivial tool can spontaneously emerge in an evolved neural controller for a two degree-of-freedom articulated limb in a target-reaching task. For evolution of the controllers, we used a neural circuit evolution algorithm that permits incrementally changing topology (NEAT). Our results show that minimal, indirect fitness criteria such as distance to goal (D) , speed to reach the goal (S), and tool pick-up frequency (T) are enough the give rise to tool use behavior. Although the controller was not aware or not made aware of the significance of tool fetching event, inclusion of T in the fitness significantly improved performance. We also found that among the successful neural circuit controllers, those with more recurrent loops were even more effective. When recurrent connections were prohibited, performance suffered. We expect our results to help us better understand the origin of tool use and the kind of neural circuits that enabled such a powerful trait. As a future work, we plan to evolve the neural circuit controllers for a robot arm to investigate how the controllers can evolve in a real world environment, where higher degree-of-freedom, 3D environment, various sensory feedbacks from cameras and motor sensors, and the dynamics of the robot arm can be incorporated.

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 [21].

References

[1] 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.

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

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

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

[5] G. R. Hunt and R. D. Gray, "The crafting of hook tools by wild new caledonian crows," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 271, no. Suppl 3, pp. S88–S90, 2004.

[6] A. Streri and J. Féron, "The development of haptic abilities in very young infants: From perception to cognition," *Infant Behavior and Development*, vol. 28, no. 3, pp. 290–304, 2005.

[7] A. Maravita and A. Iriki, "Tools for the body (schema)," *Trends in cognitive sciences*, vol. 8, no. 2, pp. 79–86, 2004.

[8] 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.

[9] R. S. Amant and A. B. Wood, "Tool use for autonomous agents." in *Twenteth AAAI Conference on Artificial Intelligence*, 2005, pp. 184–189.

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

[11] D. Lee, H. Kunori, and Y. Nakamura, "Association of whole body motion from tool knowledge for humanoid robots," in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, 2008, pp. 2867–2874.

[12] A. M. Arsenio, "Learning task sequences from scratch: applications to the control of tools and toys by a humanoid robot," in *Control Applications, 2004. Proceedings of the 2004 IEEE International Conference on*, vol. 1. IEEE, 2004, pp. 400–405.

[13] R. Saegusa, G. Metta, G. Sandini, and L. Natale, "Developmental perception of the self and action," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 183–202, Jan 2014.

[14] Y. Wu and Y. Demiris, "Learning dynamical representations of tools for tool-use recognition," in *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2664–2669.

[15] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, "Learning and generalization of motor skills by learning from demonstration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. IEEE, 2009, pp. 763–768.

[16] S. Nishide, J. Tani, T. Takahashi, H. G. Okuno, and T. Ogata, "Tool–body assimilation of humanoid robot using a neurodynamical system," *Autonomous Mental Development, IEEE Transactions on*, vol. 4, no. 2, pp. 139–149, 2012.

[17] A. Stoytchev, "Behavior-grounded representation of tool affordances," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 3060–3065.

[18] D. Katz and O. Brock, "Manipulating articulated objects with interactive perception," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 272–277.

[19] B. Schäfer, N. Bergfeldt, M. J. Riveiro Carballa, and T. Ziemke, "Evolution of tool use behavior," in *Proceedings of the First IEEE Symposium on Artificial Life*. Citeseer, 2007, pp. 31–38.

[20] K. Sims, "Evolving 3d morphology and behavior by competition," *Artificial life*, vol. 1, no. 4, pp. 353–372, 1994.

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

[22] K. O. Stanley, B. D. Bryant, I. Karpov, and R. Miikkulainen, "Real-time evolution of neural networks in the nero video game," in *Twenty-First AAAI Conference on Artificial Intelligence*, vol. 6, 2006, pp. 1671–1674.

[23] J. Gauci and K. O. Stanley, "A case study on the critical role of geometric regularity in machine learning." in *Twenty-Third AAAI Conference on Artificial Intelligence*, 2008, pp. 628–633.

[24] 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.

[25] 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.

[26] E. Visalberghi and L. Limongelli, "Acting and understanding: Tool use revisited through the minds of capuchin monkeys," *Reaching into thought: The minds of the great apes*, pp. 57–79, 1996.

[27] J. Kwon and Y. Choe, "Predictive internal neural dynamics for delay compensation," in *Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on*. IEEE, 2010, pp. 443–448.

[28] E. Ruppin, "Evolutionary autonomous agents: A neuroscience perspective," *Nature Reviews Neuroscience*, vol. 3, no. 2, pp. 132–141, 2002.

[29] Y. Choe, "Connectome, general," *Encyclopedia of Computational Neuroscience*, 2014.

[30] Y. Choe, Q. Li, J. Huang, and J. Yoo, "Synthetic connectomics to tackle big data challenges in its natural counterpart," in *US-Korea Conference on Science, Technology, and Entrepreneurship (UKC 2015)*, 2015.

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

[32] 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. 1095.

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