How Robot Morphology and Training Order Affect the Learning of Multiple Behaviors

Joshua Auerbach Josh C. Bongard

Abstract— **Automatically synthesizing behaviors for robots with articulated bodies poses a number of challenges beyond those encountered when generating behaviors for simpler agents. One such challenge is how to optimize a controller that can orchestrate dynamic motion of different parts of the body at different times. This paper presents an incremental shaping method that addresses this challenge: it trains a controller to both coordinate a robot's leg motions to achieve directed locomotion toward an object, and then coordinate gripper motion to achieve lifting once the object is reached. It is shown that success is dependent on the order in which these behaviors are learned, and that despite the fact that one robot can master these behaviors better than another with a different morphology, this learning order is invariant across the two robot morphologies investigated here. This suggests that aspects of the task environment, learning algorithm or the controller dictate learning order more than the choice of morphology.**

I. Introduction

Robots with three-dimensional, articulated bodies that act in physical or physically-realistic environments must be able to coordinate motion of different subsets of their body parts during different phases of performing a task. In this work a behavior is defined as the successful coordination of one of these subsets to achieve part of a desired task. Ideally, the same controller should be able to direct these different behaviors and allow transitions between them.

Evolutionary robotics [1], [2] is an established technique for generating robot behaviors that are difficult to derive analytically from the robot's mechanics and task environment. In particular, such techniques are useful for realizing dynamic behaviors (eg. [3], [4]) in which individual motor commands combine in a nonlinear fashion to produce behavior, thereby making analytical derivations of optimal controllers infeasible. However, evolutionary algorithms alone are often not sufficient to evolve multiple dynamic behaviors: to date most reported efforts have primarily focused on realizing a single behavior, such as locomotion [3], [4] or grasping [5], [6].

Previous work has shown that it is possible to realize multiple behaviors in a robot by gradually incorporating more modules into its controller [7], [8]. However, this approach does not scale well as the number of modules, and therefore the size of the controller grows with the number of behaviors. A scalable approach to behavioral flexibility might allow the same dynamic controller to exhibit multiple attractor states, in which individual behaviors correspond to individual attractor states, an idea with some currency in the robotics literature [9], [10]. One of the main difficulties in this approach however is realizing multistability [11] in the controller: it should settle into different attractor states that correspond to the different desired behaviors in the face of the appropriate sensory stimulation. Another recent finding indicates that rather than different behaviors corresponding to different attractor states, they may correspond to distinct transients within the dynamical system composed of the agent's environment, body and brain [12].

This paper extends the results reported in [13] in which a virtual legged robot was trained to perform a mobile manipulation [14], [15] task. The robot in [13] learned to coordinate its legs to locomote toward an object and then coordinate the motions of a gripper to achieve object manipulation. It was demonstrated there that successful attainment of both these behaviors is dependent on the order in which they are learned. This result lends support to the growing body of evidence that incremental shaping ([16], [17] and [18]) – the gradual complexification of an agent's task environment, also known in the developmental psychology literature as scaffolding [19] – can improve the probability of successful learning. However, the selection of an appropriate scaffolding schedule that enforces the order in which behaviors should be learned greatly impacts the probability of the agents successfully learning all of the behaviors [20]. The question then arises as to what dictates this learning order: the task environment, the learning algorithm, the controller, the robot's morphology, or some combination of all four.

In the work presented here the dynamic scaffolding method described in [13] is extended to enable a virtual autonomous robot to overcome three learning milestones: object manipulation, dynamic forward legged locomotion toward an object, and directed legged locomotion toward an object, all using a single monolithic controller – a feat, insofar as the authors are aware, that has not been previously reported in the literature. It is shown that, from among several scaffolding schedules that attempt to train the robot to achieve these behaviors in different orders, that the one that selects for manipulation, then forward locomotion, and then directed locomotion significantly increases the probability of a robot successfully learning all three, and that this order is invariant across two different robot morphologies that were investigated. In the next section the virtual robots and the incremental shaping method are introduced; the following section reports results demonstrating how this method, with the proper scaffolding schedule, can produce controllers that

Morphology, Evolution and Cognition Laboratory – Department of Computer Science – University of Vermont – Burlington, VT 05401 – joshua.auerbach@uvm.edu

succeed in previously unseen environments, and the final sections provide some discussion and directions for future investigation.

II. Methods

This section first describes the two virtual robots used for this work followed by a description of their controllers. Next the incremental shaping algorithm used for training the robots is presented along with the various dynamic scaffolding schedules investigated here. The section concludes with a description of the metrics used to evaluate the robots' success.

A. The robots

In this work two virtual quadrupedal robots are used[1]. **Robot 1** (Fig. 1, left) is comprised of a main body, four legs and a front gripper. Each leg consists of an upper and lower part connected to each other and the main body. The gripper is comprised of a small spherical base connecting the main body to the gripper pincers. The gripper base can be rotated upward relative to the main body, and both the left and right pincers are comprised of a gripper arm (proximal to the gripper base) and gripper tip (distal to the gripper base). This robot is identical to the one used in [13] and the reader is referred there for more details regarding the robot's morphology.

Robot 2 (Fig. 1, right) is identical to robot 1 except for the orientation of the legs. Robot 2 has been modified by rotating the legs at the point they are attached to the main body such that each is positioned at a 45° angle to the main body. The upper legs in this robot move vertically in the plane defined by the vector lying along the upper leg and a downward-pointing vector, while the lower legs continue to move in the sagittal plane. This alteration was implemented to make turning easier.

Eight motors actuate the four upper and lower legs, another motor actuates the gripper base, and four motors actuate the base and distal parts of the left and right gripper pincers, for a total of 13 motors. A touch sensor and distance sensor reside in both the left and right gripper tips, a rotation sensor resides in the gripper base, and a distance sensor resides on the robot's back, for a total of six sensors. The touch sensors return a value of one when the corresponding body part touches another object and zero otherwise. The distance sensors return a value commensurate with the sensor's distance from the target object: they return zero if they are greater than five meters from the target object and a value near one when touching the target object. Object occlusion is not simulated here; the target object can be considered to be emitting a sound, and the distance sensors respond commensurately to volume.

The robots attempt to locomote toward, grasp and lift a rectangular target object that is placed at varying locations in relation to the robot. Unlike the robot's task in [13], in this work the target object is not constrained to being placed in front of the robot within its sagittal plane: additional target object placements away from the robot's centerline select for turning behavior.

[1]These results have not yet been validated on a physical robot, as the multiple morphologies would require constructing a morphologically-reconfigurable legged robot. However, this option will be explored in future work.

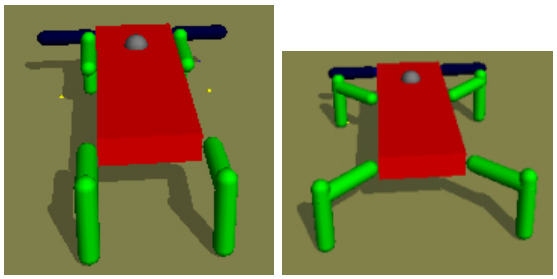

Fig. 1. The two virtual robots used in this work: **Robot 1** (left), **Robot 2** (right).

B. The controllers

Each robot is controlled by a continuous time recurrent neural network [21]. The CTRNN is composed of 11 motor neurons (the two gripper arm motors share the same motor neuron, as do the two gripper tip motors to ensure the gripper closes symmetrically). The remaining 9 motors each receive commands from their own motor neuron. Other network configurations such as those containing non-motor or hidden neurons were experimented with, but are omitted from the current work, because they were not found to improve performance.

The value of each motor neuron is updated according to

$$\dot{y_i} = \frac{1}{\tau_i}\left(-y_i + \sum_{j=1}^{11} w_{ji}\sigma(y_j + \theta_i) + \sum_{j=1}^{6} n_{ji}s_j\right) \quad (1)$$
$$\text{for } 1 \leq i \leq 11$$

where y_i is the state of neuron i, w_{ji} is the weight of the connection from neuron j to neuron i, τ_i is the time constant of neuron i, θ_i is the bias of neuron i, n_{ji} is the weight of the connection from sensor j to neuron i, s_j is the value of sensor j and $\sigma(x) = 1/(1 + e^{-x})$ is the logistic activation function.

The virtual robot with a given CTRNN controller is evaluated over a set number of simulation steps in a physical simulator[2]. For each simulation step, using a step size of 0.0005, the sensors, CTRNN, joint torques and resulting motion are updated.

C. Training

A version of incremental shaping extended from the algorithm presented in [13] is used for dynamically tuning the robot's task environment to facilitate learning. This method is outlined in Fig. 2. A random CTRNN is created by choosing all τ from the range [0.1, 0.5], all w from [-16, 16], all θ

[2]Open Dynamics Engine: www.opende.org

from [-1, 1], and all n from [-16, 16]; these ranges were found useful in previous work [13]. This gives a total of $11 + 11 * 11 + 11 + 6 * 11 = 209$ evolvable parameters. The robot is then equipped with this controller and allowed to behave in a task environment for 100 time steps in which the target object is placed directly in front of the robot. After evaluation the fitness of the controller is computed as

$$f_{sub} = \max_{k=1}^{t}(D(LeftgripperTip, k) * D(RightgripperTip, k)) \quad (2)$$

if the touch sensors in the left and right gripper tips fail to fire at the same time during any time step of the evaluation period, and

$$f_{sub} = 1 + \max_{k=1}^{t}(D(SensorNode, k)) \quad (3)$$

otherwise, where t is the evaluation time, and $D(x, k)$ indicates the value of the distance sensor affixed to body part x at time step k. Eqn. 2 rewards controllers for steering the robot toward the target object. Eqn. 3 rewards controllers for also lifting the target object onto the robot's back (where the sensor node is located) while it is touching the target object with both gripper tips.

One extension added to the algorithm used in this work over that of [13] is that a single CTRNN controller is evaluated in multiple environments in which the target object is placed at different locations. The final fitness of the controller is computed as

$$f = \min_{b=1}^{S} f_{sub}(b) \quad (4)$$

where S is the number of target object locations or sub-evaluations that the CTRNN is evaluated for and $f_{sub}(b)$ is the fitness of the CTRNN on sub-evaluation b (see eqns. 2,3). Using the minimum fitness over all sub-evaluations renders a given CTRNN only as fit as it is in its weakest sub-evaluation which prevents finding CTRNNs that specialize at picking up the target object in one location, but do not work well in others.

A hill climber [22] is used to optimize the initial random CTRNN against this fitness function. At each generation a child CTRNN is created from the current best CTRNN and mutated. Mutation involves considering each τ, w, θ and n value in the child, and replacing it with a random value in its range with a probability of $10/209 = 0.0478$. This ensures that, on average, 10 mutations are incorporated into the child according to a normal distribution. If the fitness of the child CTRNN is equal to or greater than the fitness of the current best CTRNN, and the child CTRNN is either successful at picking up the target object in either the current or previous environment, then the best CTRNN is replaced by the child; otherwise the child is discarded. This ensures that the grasping behavior learned in previous environments is retained while the locomotion behavior is adapted to the current environment.

After each possible replacement, the current CTRNN is considered in order to determine whether a failure condition has occurred, or whether it has achieved the success criteria. In the present work the failure condition is defined as 100 generations of the hill climber elapsing before a successful CTRNN is found. A successful CTRNN is defined as one for which, at some time step during the current evaluation both gripper tips touch the target object and it is lifted far enough onto the robot's back such that the distance sensor there fires above a certain threshold.

If the failure condition occurs, the task environment is eased; if the current CTRNN succeeds, the task environment is made more difficult. Easing the task environment involves increasing the current evaluation period by 10 time steps. This has the effect of giving the robot more time to succeed at the current task if it fails. Making the task environment more difficult involves moving the target object further away from the robot. This has the effect of teaching the robot to grasp and lift the target object when it is close, and learning to turn and locomote toward the target object, followed by grasping and lifting it, when it is placed further away.

```
1)  IncrementalShaping()
2)     Create and evaluate random parent p
3)     WHILE ~Done()
4)        Create child c from p, and evaluate
5)        IF Fitness(c) ≥ Fitness(p) AND
           ( PreviousSuccess(c) OR Success(c) )
           [see Eqns. 2,3,4]
6)           p = c
7)        IF Failure()
8)           EaseEnvironment()
9)           Re-evaluate p
10)       WHILE Success(p)
11)          HardenEnvironment()
12)          Re-evaluate p
13) Done()
14)    30 hours of CPU time have elapsed
15) Failure()
16)    100 generations since last success
17) EaseEnvironment()
18)    EvaluationTime ← EvaluationTime+100
19) Success(g)
20)    ∃k, k ∈ {1, ..., t} |
21)     T(LeftgripperTip, k)&
22)     T(RightgripperTip, k)&
23)     D(SensorNode, k) ≥ 0.825
24) PreviousSuccess(g)
25)    TargetDistance ← TargetDistance-0.01m
26)    success = Success(g)
27)    TargetDistance ← TargetDistance+0.01m
28)    RETURN success;
29) HardenEnvironment()
30)    TargetDistance ← TargetDistance+0.01m
```

Fig. 2. **Incremental Shaping pseudocode.** The algorithm executes a hill climber [1-14] (see text for description). If the current genome fails [15,16], the task environment is eased [17,18]; while it is successful [19-23], the task environment is made more difficult [24,25]. $T(x, k)$ returns 1 if body part x is in contact with another object and zero otherwise at time step k. $D(x, k)$ returns the value of the distance sensor located at body part x at time step k.

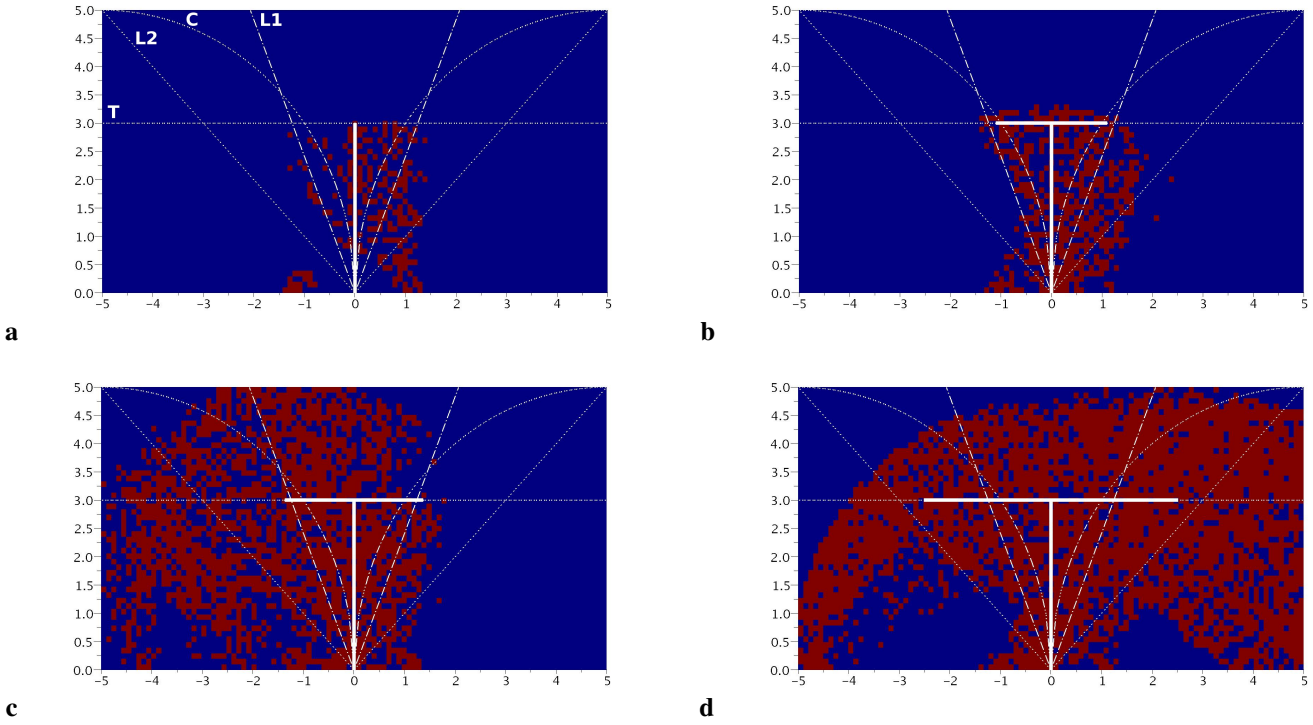

Fig. 3. Sample generalization plots from evolution of a generalized controller on robot 2 (red indicates the robot was successful at picking up the target object at that location) with the four scaffolding schedules superimposed. Specifically the plots shown are for controllers that were successful at distances of 3 meters (**a**), 3.2 meters (**b**), 3.3 meters (**c**) and 3.92 (**d**) the final training distance reached in this run.

As some CTRNNs that succeeded for a given target object distance also succeed when the target object is moved further away, the target object is continually moved until the current CTRNN no longer succeeds, at which time hill climbing recommences. In order to further speed the algorithm an individual evaluation is terminated early if the robot ceases to move before succeeding at the task.

D. Scaffolding Schedules

As mentioned above each CTRNN is evaluated at multiple target object locations. These locations are a function of the distance of the target object from the robot, which increases with each success. Specifically, four different such functions, or scaffolding schedules were compared in this work. All four attempt to select first for grasping followed by a combination of turning and locomoting. The schedules are created in this way because it was shown in [13] that selecting for grasping first proved the best way to achieve both grasping and locomotion.

The first scaffolding schedule, henceforth referred to as '**T**', begins with only one sub-evaluation and places the target object in front of the robot at increasing distance until the target object is a distance of three meters from the robot. It was observed that by this distance, the robot must have learned a stable gait to reach the target object. As distance is increased past three meters the target object is moved out in both directions along the line perpendicular to the robot's sagittal plane, requiring two sub-evaluations: one sub-evaluation with the target object placed in front and to the left, and another in which the target object is placed in front and to the right of the robot. Formally

$$(x, z) = \begin{cases} (0, L), & \text{if } L \leq 3.0 \\ (\pm\sqrt{L^2 - 9.0}, 3.0), & \text{otherwise} \end{cases} \quad (5)$$

where L is the distance of the target object from the robot's start location. This schedule is depicted graphically as the thick lines in Fig. 3. The next schedule used is

$$(x, z) = (\pm L^2/10.0, \sqrt{10.0|x| - x^2}) \quad (6)$$

that is the target object is moved concurrently along the perimeter of circles with radius 5 meters and centers at 5 and -5 meters ('**C**'). In this case two sub-evaluations are always used. The final two schedules both move the target object away from the robot linearly on both sides. One does so with a slope $m = 1/\tan(22.5°)$ ('**L1**') and the other does so with a slope $m = 1/\tan(45°) = 1$ ('**L2**'). In both these cases the function used is

$$(x, z) = (\pm L/\sqrt{(m^2) + 1}, |mx|) \quad (7)$$

See Fig. 3 for a graphical representation of these schedules.

In order to speed evaluation of child CTRNNs in schedules with multiple sub-evaluations, if the sub-fitness of the first sub-evaluation attempted by the child CTRNN is lower than the fitness of the current best CTRNN (which was set to its lowest scoring sub-fitness), then no additional sub-evaluations are performed and the child CTRNN is discarded.

1) **GeneralizationTest**()
2) NumSuccesses = 0;
3) FOR $x = -5$; $x \leq 5$; $x+ = 0.1$
4) FOR $z = 0$; $z \leq 5$; $z+ = 0.1$
5) Place target object at (x, z) and let simulation run for 10,000 time steps
6) If Success() [see Fig. 2]
7) NumSuccesses++;
8) RETURN (NumSuccesses / 5151)

Fig. 4. GeneralizationTest, the 10x5 grid is uniformly sampled at 101x51=5151 target object locations to determine percentage of grid co-ordinates where the controller is successful.

E. Measuring Performance

In order to evaluate the quality of an evolved CTRNN, two metrics are considered. The first is how far away the target object was placed at the end of 30 hours of training. While this metric is useful for judging how rapidly the robot can adapt to a changing environment it does not measure how successful a given CTRNN is in unseen environments. For this purpose a generalization metric has been devised. If the point directly in front of the robot is considered to be the origin of a Euclidean space, then a 10 meter by 5 meter grid extending from (-5,0) to (5,5) can be constructed and a controller can be systematically tested to determine how well it performs the task for a sampling of target object locations within this grid. Specifically, this grid is sampled as shown in Fig. 4. Additionally, for each grid position, whether or not the controller was successful there is recorded and can be plotted as shown in Figs. 3 and 5.

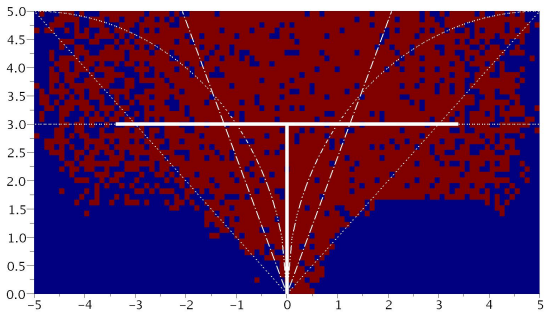

Fig. 5. Generalization plot from best controller for robot 1.

III. Results

For each robot and each scaffolding schedule mentioned above a set of 100 independent runs were conducted giving a total of $2 * 4 * 100 = 800$ total runs. Each run consisted of running the incremental shaping algorithm for 30 hours of CPU time. At the completion of each run, the generalization test as described in Fig. 4 was performed on the final CTRNN from that run to test its ability to generalize to unseen environments. For each set of runs, the mean final target object distance and the mean generalization percent of those final CTRNNs are plotted in Fig. 6. While the mean generalization score for each set of runs was under 10% in all instances, there were runs in each set that found controllers with much higher generalization values. The generalization scores for the final controllers from the top five runs from each set are given in Table I.

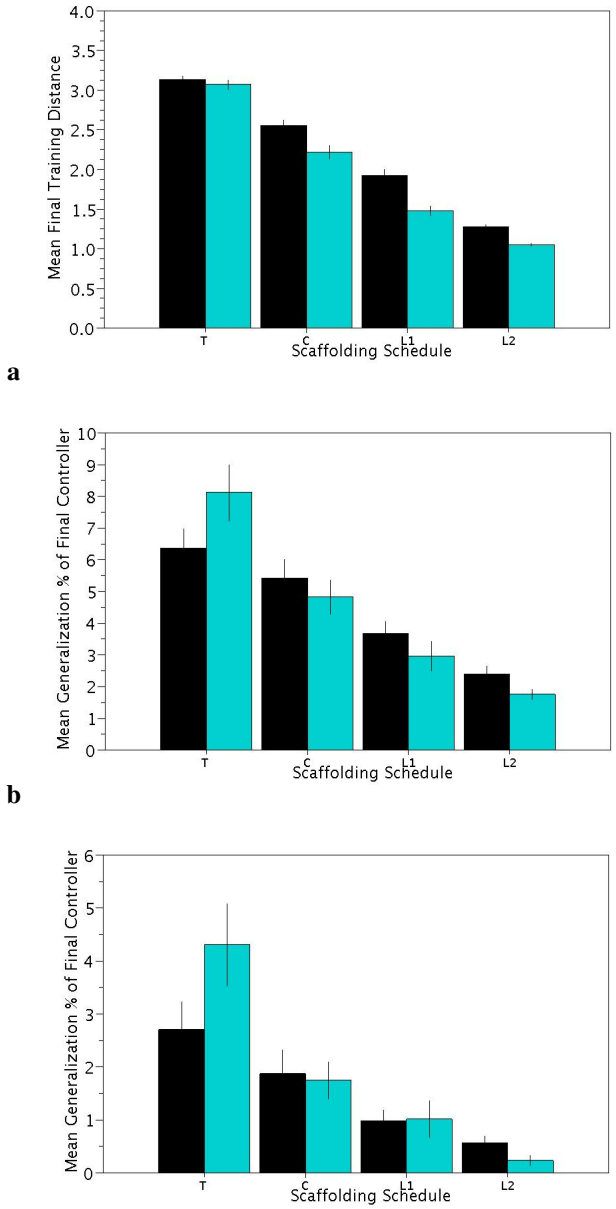

Fig. 6. Mean final distance achieved in training (**a**) mean generalization % of final CTRNN (**b**), and mean generalization % of final CTRNN for all locations where $x \notin [-1, 1]$ (**c**) across the 100 runs for each of the two virtual robots (robot 1 in black, robot 2 in blue) and each of the four scaffolding schedules. All plots include standard error bars.

The **T** scaffolding schedule significantly outperforms the other three schedules both in training distance achieved and generalization, for both robots. Comparing performances between robots, it is noted that the **T** schedule evolves

Schedule:	T	C	L1	L2
Robot 1:	53.6%	32.5%	23.3%	13.2%
	20.2%	28.3%	19.7%	12.7%
	16.6%	24.7%	14.9%	9.7%
	15.2%	24.3%	13.2%	9.2%
	15.1%	22.7%	11.5%	9.0%
Robot 2:	57.7%	26.3%	24.7%	12.6%
	40.4%	24.8%	24.1%	8.9%
	28.4%	21.4%	21.9%	7.6%
	27.4%	19.3%	19.6%	5.8%
	26.4%	19.1%	13.5%	4.8%

TABLE I

FIVE BEST GENERALIZATION VALUES OF FINAL CONTROLLERS FROM EACH SET.

significantly more generalized controllers with the second robot (left hand grouping in Fig. 6b,c) while reaching similar final training distances as the first robot (left hand grouping in Fig. 6a). While the relative performance of the four schedules remains consistent across robots, the three other schedules lead to slightly less generalized controllers with the second robot (three right hand groupings in Fig. 6b).

A. A Sample Evolved Controller

Fig. 7 shows the behavior of the controller that achieved the highest generalization score overall, which comes from using the **T** schedule with robot 2. Here it can be seen how the behaviors differ based on target object locations. Fig. 7a-h show the robot picking up the target object when it is located in front and to the right of the robot's initial position. The robot actually turns too far to the right while approaching the target object and then straightens itself out before picking up the target object. Fig. 7i-p show the same CTRNN controlling the robot to pick up the target object when it is located forward and to the left of the robot's initial position. In this case the robot does not turn too far, but approaches the target object at an angle that allows it to swing the target object onto its back.

The results of the generalization test performed on this same CTRNN are shown in Fig. 3d. This plot is colored red for all the locations where the CTRNN was successful in picking up the target object, and blue where it was not. This controller was able to pick up the target object in over 50% of target object locations. Specifically, there are large number of locations at which the CTRNN is successful even though it was never exposed to these locations during its training. Also it is noted that this CTRNN is successful for the majority of locations it would have experienced under any of the other scaffolding schedules, indicating it is possible for a controller to succeed at those locations, but that it can only do so after forward locomotion has been learned (as enforced by the **T** scaffolding schedule).

Fig. 3a-c show generalization plots for controllers from the same run as Fig. 3d that were saved when the robot was successful at training distances of 3, 3.2, and 3.3 meters respectively; that is, these controllers are ancestors of the final CTRNN from this run. It can be seen that there is a discontinuous jump in generalization between 3.2 and 3.3 meters. This illustrates how between these two distances, the increased pressure for the controller to learn turning resulted in a much greater ability to generalize to unseen environments once turning was mastered.

IV. DISCUSSION

A. Order Matters

The question presents itself as to why the **T** scaffolding schedule results in more successful controllers than any of the other schedules. The justification given in [13] is that the order in which the necessary behaviors needed to complete a task are selected for greatly affects the probability that all behaviors will be learned. In that work it was shown that if the robot was trained to pick up the target object first followed by training for locomotion it was more successful than if it was trained to locomote first and then trained to pick up the target object. Based on this result, all four schedules presented in this work select for grasping first, but the **T** schedule allows the robot to learn forward locomotion and then additionally learn the taxis behavior. The other three schedules each, to varying degrees, pressure the robot to learn turning toward the target object either before or while learning to locomote. This proves that, because these three schedules are less successful, forward locomotion should be learned before turning, for both robot morphologies. As can be seen in Fig. 6 the probability of training a controller to enable taxis and object manipulation is inversely proportional to the pressure to learn turning before locomotion: the **T**, **C**, **L1**, and **L2** schedules decline in performance, but increase in the pressure they exert to learn turning before locomotion.

B. Training Milestones

Another way to consider why the **T** schedule yields the most successful controllers is that it forces the evolved controllers to achieve certain milestones during training. Fig. 8 reports the rate at which both robots overcome these milestones using the **T** scaffolding schedule. Almost all runs rapidly reach around one meter, the furthest point at which the robot can pick up the target object by leaning or lunging forward without having to take any steps. The drop in learning rate (represented by increased slope) at this point denotes the difficulty in incorporating an oscillatory dynamic into the controller to allow stepping while retaining the dynamic that allows grasping and lifting once the target object is reached. This is the first learning milestone. Between one and three meters the learning rate is relatively constant: CTRNN parameters are tuned to enable stable oscillations, which induce rhythmic motion in the legs, thus carrying the robot to the target object. This is the second learning milestone.

When the target object is placed more than three meters from the robot and an increasing distance away from its sagittal plane, there is a growing asymmetry in the distance sensor values reported by the two claw tips at the outset of an evaluation. This point corresponds to an apparent slowing in the learning rate as shown by the greater slope to the right in Fig. 8. It is acknowledged that the learning rate is expected

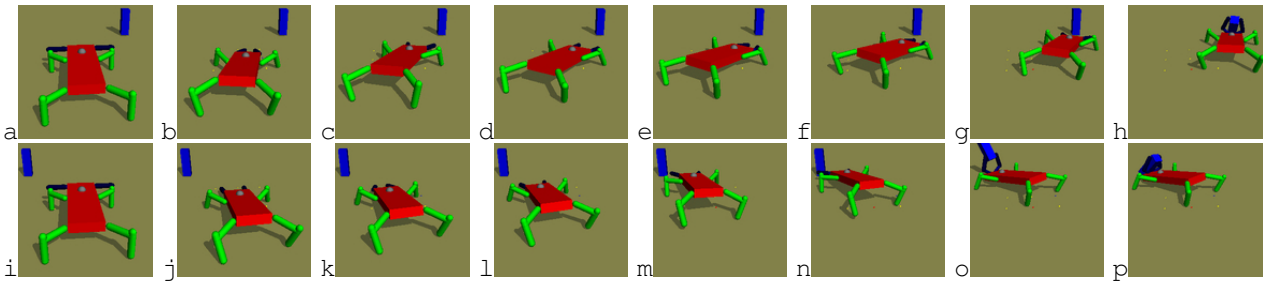

Fig. 7. A sample successful controller for **robot 2**. **a-h**: The robot moves toward the target object placed 3.9 meters ahead and to the right while turning (**a-d**), turns too far (**e,f**), compensates (**g**) and then picks up the target object (**h**). **i-p**: The robot moves toward the target object placed equal-distance away on the left side without overshooting (**i-n**) and swings it onto its back (**o-p**).

to slow somewhat as the controller is now evaluated in two environments instead of one (the target object is placed to the right and then to the left). However, it can be seen that the learning rates for the two robots are not the same: robot 2 more rapidly adapts to target object placements further from its sagittal plane than robot 1 does. This indicates that the slowed learning rate is not only a result of the increased evaluations, but is also a function of morphology and behavior: robot 2's morphology eases the transition from forward locomotion to directed locomotion better than robot 1's morphology does.

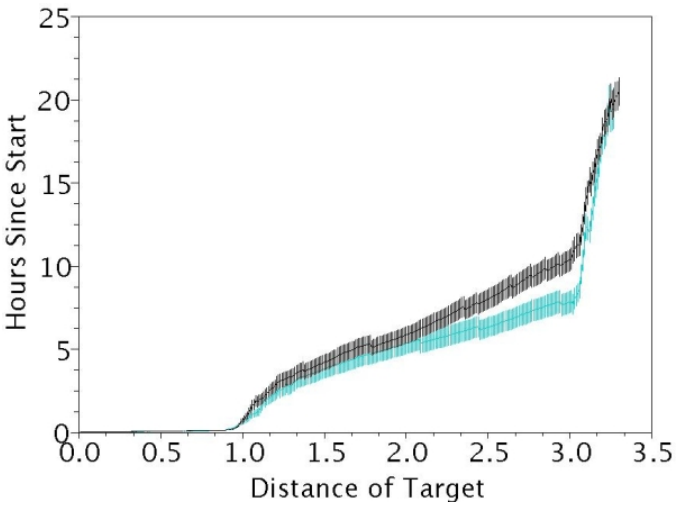

Fig. 8. This plot shows for the **T** schedule: robot 1 (black) and robot 2 (blue) the target object distance in training vs. mean time to reach that distance with standard error bars for all distances reached by at least 30 runs. Many runs surpassed this distance, but are not shown for the sake of clarity.

C. Morphology Matters

Fig. 9 plots the maximum distance to which the target object was moved during training against the number of runs (out of 100) that produced successful controllers for that distance before their time limit of 30 hours expired. It can be seen that more of the runs using robot 1 discovered controllers that drove the robot to a distance of three meters, compared to the runs using robot 2 (the blue line is above the black line between one and three meters in Fig. 9). This is presumably due to the fact that controllers can be more easily trained to produce forward locomotion in robot 1, which has legs parallel to its sagittal plane and therefore to its direction of travel. However, more runs using robot 2 discover controllers that allow the robot to reach and manipulate the target object when it is placed beyond three meters and away from its centerline, evidenced by the crossing of the lines around 3.2 meters. This is presumably due to the splayed legs of robot 2 allowing for directed locomotion more easily.

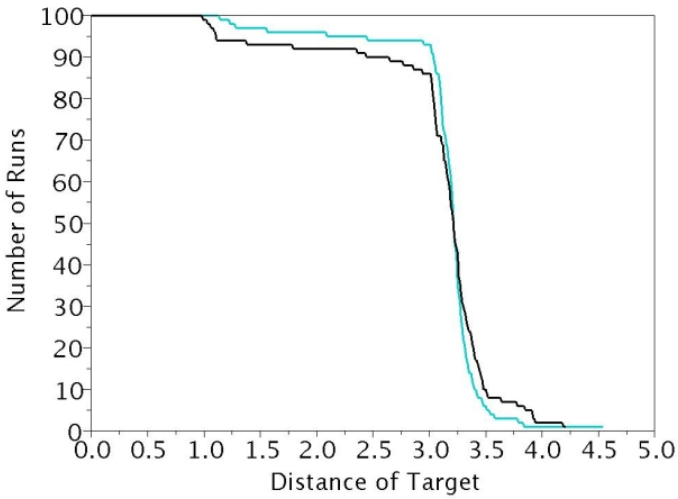

Fig. 9. This plot shows for the **T** schedule: robot 1 (black) and robot 2 (blue) the target object distance in training vs. number of runs reaching that distance.

This observation is strengthened by Fig. 5, which reports the generalization ability of the best controller evolved for robot 1. Despite the robustness of this controller (it guides the robot toward success in 53.6% of the target object placements), the robot is rarely successful in regions that require a small turning radius (the two regions in the lower left and right of Fig. 5). This further suggests that robot 2 is better able than robot 1 to learn turning.

One last piece of evidence supporting this observation can be seen in Fig. 6c. Here the generalization abilities of the two robots across all four scaffolding schedules are compared, but these values are calculated considering only target object placements outside of $x \in [-1, 1]$: locations

that require turning, because the target object is at least one meter away from the robot's sagittal plane. It is noted that the difference in scores between robot 1 and 2 using the **T** schedule are greater in this plot than in Fig. 6b, in which all target object locations are considered. This further confirms that controllers evolved for robot 2 are more likely to be able to pick up target objects at locations that require turning.

V. Conclusions and Future Work

This work has demonstrated that with the proper scaffolding schedule (**T**) it is possible to evolve controllers capable of performing a non-trivial sequence of behaviors even in previously unseen environments. Moreover it has demonstrated that altering morphology can impact the performance achievable through incremental shaping: robot 2 resulted in more generalized behaviors than robot 1.

However, for the two morphologies considered in this work it does not alter the sequence in which behaviors should be learned. Robot 2's splayed legs make turning easier, however scaffolding schedules that select for turning before locomotion is learned were not better able to integrate object manipulation, turning and locomotion into a controller using this body plan. Therefore it is concluded that the task environment, the learning algorithm, and/or the evolvability of CTRNNs dictate learning sequence more than morphology does.

In order to strengthen this conclusion more morphologies will need to be considered. Future work will investigate how additional morphologies perform under these scaffolding schedules. Additionally the authors intend to investigate how evolving the robot's body plan along with its controller may result in less sensitivity to the order in which behaviors are learned. This would simplify the application of shaping for realizing multiple dynamic behaviors in intelligent agents.

References

[1] I. Harvey, P. Husbands, D. Cliff, A. Thompson, and N. Jakobi, "Evolutionary robotics: the sussex approach," *Robotics and Autonomous Systems*, vol. 20, pp. 205–224, 1997.

[2] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology,Intelligence,and Technology*. Cambridge, MA, USA: MIT Press, 2000.

[3] T. Reil and P. Husbands, "Evolution of central pattern generators for bipedal walking in a real-time physics environment," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 159–168, Apr 2002.

[4] G. Hornby, S. Takamura, T. Yamamoto, and M. Fujita, "Autonomous evolution of dynamic gaits with two quadruped robots," *Robotics, IEEE Transactions on*, vol. 21, no. 3, pp. 402–410, June 2005.

[5] J. J. Fernandez Jr. and I. D. Walker, "A biologically inspired fitness function for robotic grasping," in *Proc. of the Genetic and Evolutionary Computation Conf. GECCO-99*, W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, Eds. San Francisco, CA: Morgan Kaufmann, 1999, pp. 1517–1522.

[6] A. Chella, H. Dindo, F. Matraxia, and R. Pirrone, "Real-time visual grasp synthesis using genetic algorithms and neural networks." in *AI*IA*, ser. Lecture Notes in Computer Science, R. Basili and M. T. Pazienza, Eds., vol. 4733. Springer, 2007, pp. 567–578.

[7] R. Brooks, "A robust layered control system for a mobile robot," *Robotics and Automation, IEEE Journal of [legacy, pre - 1988]*, vol. 2, no. 1, pp. 14–23, 1986. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1087032

[8] R. Calabretta, S. Nolfi, D. Parisi, and G. P. Wagner, "Duplication of modules facilitates the evolution of functional specialization," *Artif. Life*, vol. 6, no. 1, pp. 69–84, 1999.

[9] T. Inamura, I. Toshima, and H. Tanie, "Embodied symbol emergence based on mimesis theory," *International Journal of Robotics Research*, vol. 23, no. 4, pp. 363–377, 2004.

[10] M. Okada and Y. Nakamura, "Design of the continuous symbol space for the intelligent robots using the dynamics-based information processing," *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 4, pp. 3201–3206 Vol.4, 26-May 1, 2004.

[11] J. Foss, F. Moss, and J. Milton, "Noise, multistability, and delayed recurrent loops," *Physical Review E*, vol. 55, no. 4, pp. 4536+, April 1997. [Online]. Available: http://dx.doi.org/10.1103/PhysRevE.55.4536

[12] E. Izquierdo and T. Buhrmann, "Analysis of a dynamical recurrent neural network evolved for two qualitatively different tasks: walking and chemotaxis," in *Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, S. Bullock, J. Noble, R. Watson, and M. A. Bedau, Eds. MIT Press, Cambridge, MA, 2008, pp. 257–264.

[13] J. Bongard, "Behavior chaining: incremental behavioral integration for evolutionary robotics," in *Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, S. Bullock, J. Noble, R. Watson, and M. A. Bedau, Eds. MIT Press, Cambridge, MA, 2008, pp. 64–71.

[14] W. Carriker, P. Khosla, and B. Krogh, "Path planning for mobile manipulators for multiple task execution," *IEEE Transactions on Robotics and Automation*, pp. 403 – 408, June 1991.

[15] H. Seraji, "A unified approach to motion control of mobile manipulators," *The International Journal of Robotics Research*, vol. 17, no. 2, pp. 107–118, 1998.

[16] S. P. Singh, "Transfer of learning across sequential tasks," *Machine Learning*, vol. 8, pp. 323–339, 1992.

[17] M. Dorigo and M. Colombetti, "Robot shaping: Developing situated agents through learning," *Artificial Intelligence*, vol. 70, no. 2, pp. 321–370, 1994.

[18] L. Saksida, S. Raymond, and D. S. Touretzky, "Shaping robot behavior using principles from instrumental conditioning," *Robotics and Autonomous Systems*, vol. 22, pp. 231–249, 1997.

[19] D. Wood, J. Bruner, and G. Ross, "The role of tutoring in problem solving," *J Child Psychol Psychiatry*, vol. 17, no. 2, pp. 89–100, 1976.

[20] R. D. Beer, "The dynamics of brain-body-environment systems: A status report." in *Handbook of Cognitive Science: An Embodied Approach*, P. Calvo and A. Gomila, Eds. Elsevier, 2008, pp. 99–120.

[21] ——, "Parameter space structure of continuous-time recurrent neural networks," *Neural Comp.*, vol. 18, no. 12, pp. 3009–3051, 2006.

[22] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Prentice Hall, December 2002.