DOI: http://dx.doi.org/10.7551/978-0-262-31050-5-ch050

Rewarding Reactivity to Evolve Robust Controllers without Multiple Trials or Noise

Joel Lehman, Sebastian Risi, David B. D'Ambrosio, and Kenneth O. Stanley

University of Central Florida, Orlando, FL 32816
jlehman@eecs.ucf.edu, risi@eecs.ucf.edu, ddambro@eecs.ucf.edu, kstanley@eecs.ucf.edu

Abstract

Behaviors evolved in simulation are often not robust to variations of their original training environment. Thus often researchers must train explicitly to encourage such robustness. Traditional methods of training for robustness typically apply multiple non-deterministic evaluations with carefully modeled noisy distributions for sensors and effectors. In practice, such training is often computationally expensive and requires crafting accurate models. Taking inspiration from nature, where animals *react* appropriately to encountered stimuli, this paper introduces a measure called *reactivity*, i.e. the tendency to seek and react to changes in environmental input, that is applicable in single deterministic trials and can encourage robustness without exposure to noise. The measure is tested in four different maze navigation tasks, where training with reactivity proves more robust than training without noise, and equally or more robust than training with noise when testing with moderate noise levels. In this way, the results demonstrate the counterintuitive fact that sometimes training with no exposure to noise at all can evolve individuals significantly more robust to noise than by explicitly training with noise. The conclusion is that training for reactivity may often be a computationally more efficient means to encouraging robustness in evolved behaviors.

Introduction

A significant challenge in artificial life and evolutionary robotics (ER) is to evolve robust controllers for robots or artificial creatures (Nolfi and Floreano, 2000). While natural organisms are remarkably robust (i.e. they function over a wide range of environmental conditions), controllers evolved in simulation are often fragile and dependent upon overly specific simulation details (Jakobi, 1998; Koos et al., 2010). For example, a practical manifestation within ER of this issue is known as crossing the reality gap (Jakobi, 1998; Koos et al., 2010). The *reality gap* is the barrier presented by inevitable discrepancies between a simulated model and its real-world analogue. That is, robot controllers developed in simulation will most likely fail when naively transferred onto a real robot, often because of noise (i.e. non-determinism in sensors and effectors) in the real world.

Most attempts to overcome this problem craft simulations that model the real robot and its environment as accurately as possible (Cliff et al., 1993; Jakobi, 1998; Miglino et al., 1995; Nolfi and Parisi, 1996). It is also common to introduce non-determinism through noise in the sensors and effectors of the robot in simulation (Cliff et al., 1993; Jakobi, 1998; Koos et al., 2010; Miglino et al., 1995; Nolfi and Floreano, 2000). However, training with noise is not without disadvantages, such as increased computational cost from multiple non-deterministic trials (necessary to counteract variance in fitness measurements) and the difficulty of crafting a sufficiently accurate model with the right distribution of noise. Because of these disadvantages, it would be preferable to train *without* noise if there existed alternatives that also provided robustness. In this spirit, this paper presents a preliminary investigation into the possibility of encouraging robust behaviors using only information from evaluations consisting of a single deterministic trial.

While robustness can only be *verified* over multiple trials, it is still possible that there are clues to robustness hidden within even a single trial. One possible such clue is illuminated by considering how the behaviors of real animals differ from those produced by artificial evolution. Animals are robust because they do not depend upon incidental aspects of the environment (e.g. a herbivore does not depend on a particular configuration of grass blades to feed successfully). However, the same phenomenon does not hold in general for artificial systems; artificial evolution tends to exploit features specific to the simulation not present in reality. Interestingly, observing an animal only once often leaves one with an impression of its robustness. Similarly, an experimenter observing a robot behavior in simulation may often suspect its fragile nature.

The question raised by such impressions is, what cues are being perceived to make such judgments? That is, what are we noticing about animals in nature that makes them seem so vigorous? Perhaps one heuristic for judging robustness is how *reactive* their behavior appears. That is, one clue to robust behavior is noticeably *seeking and reacting* to changes in the perceived environment, which is a trait exhibited widely by natural life. Importantly, by observing a behavior it is possible to estimate how reactive it is. For example, take the behaviors of students during a lecture. If the students nod when key concepts are introduced they are re-

acting appropriately to indicate that they understand; on the other hand, unreactive students with constant blank stares reveal less information. Similarly, a blind man with a cane trying to navigate a corridor often also exhibits reactivity. If the man taps his cane continually against a wall to verify his bearings, the behavior is more reactive than if the man relies completely on a memorized layout of the corridor without re-adjusting (as artificially evolved agents often do). Intuitively, the more reactive tapping behavior would also be more robust to unforeseen changes in the corridor or missteps made by the man. Thus the hypothesis in this paper is that individuals that demonstrate their reactivity by paying attention to the world may generally be stepping stones towards robust behavior. Therefore it may prove effective to directly encourage reactivity, which is the propensity to seek and react to information in the environment continually.

While there may be many ways to quantify the notion of reactivity, the measure in this paper is based on statistical *dependence* between changes in the sensors and the effectors of a robot. Two random variables are dependent if knowing the state of one variable helps predict the other; in other words, there is some relationship between the two variables. In this way, if the magnitude of changes in sensors and effectors of a robot are dependent, it may indicate that the robot is reacting consistently to its environment (i.e. the magnitude of change in environmental input consistently influences the corresponding magnitude of changes in behavior). In this paper such dependence is measured by *mutual information*, which thereby formally captures most closely the informal idea of reactivity introduced here. Indeed, Ay et al. (2008) previously showed an important theoretical connection between maximizing mutual information in sensory experience and effective exploratory behavior in robots. This paper thus suggests how such a measure can be exploited in evolving specific goal-directed behaviors that are resistant to noise.

The idea of incentivizing reactivity to encourage robustness is explored in four maze navigation tasks designed to be challenging under noisy conditions, which makes robustness difficult to achieve. The main result is that rewarding reactivity in single-trial deterministic evaluations *without noise* produces controllers with robustness to noise often rivaling or outperforming those produced by explicitly training with noise. This result is significant because it illuminates that there are hints to robustness observable within a single non-noisy trial, and also establishes a new practical approach to training for robustness, which is a property of general interest both to artificial life and ER.

Background

This section reviews past work in evolving robust controllers in ER, the NEAT and HyperNEAT methods applied in the experiments, and multi-objective optimization.

Evolving for Robustness

For practical reasons, controllers for robots in ER are often trained in a computer simulation rather than directly in reality (Nolfi and Floreano, 2000). However, discrepancies between simulation and reality may cause controllers that are effective in simulation to fail when transferred to a real robot. Because this problem of crossing the reality gap is a significant issue in ER there exist specific training methods that attempt to mitigate it (Bongard and Lipson, 2004; Jakobi, 1998; Koos et al., 2010). The reality gap is one facet of the larger difficulty of evolving general, robust controllers that are not overly dependent on simulation details.

Nearly all training strategies for evolving robust controllers involve training at least some individuals with multiple trials, often non-deterministically (Gomez and Miikkulainen, 2004; Jakobi, 1998; Koos et al., 2010). A common motivation for such training is that real-world sensors often do experience some degree of noise; however, a deeper motivation is that strategically applying noise to a robot's sensors or effectors can prevent evolution from exploiting features specific to a particular simulation (Jakobi, 1998).

While the motivations may be reasonable, the computational cost of training with noise is significant because noisy evaluations normally consist of multiple trials to reduce uncertainty about a policy's average performance (Koos et al., 2010). To reduce computational costs, some methods seek to evaluate only *some* individuals in a full suite of noisy trials by estimating transferability for other individuals (Koos et al., 2010). Yet this approach still requires additional potentially expensive evaluations and the estimates of transferability may not always be accurate. In addition to computational costs, it is not always clear how many trials, in what distribution, and with what intensity noise should be applied in training to ensure successful transfer (Gomez and Miikkulainen, 2004). While Jakobi (1998) lays out a principled methodology based on *minimal simulations*, it still requires painstaking measuring and modeling to implement.

An interesting unexplored question is whether there exist distinguishing properties of robust robot or animat controllers that are visible in a single deterministic trial. If such properties exist and can be explicitly encouraged by an appropriate training incentive, it may be possible to evolve robust robot policies without *any* non-deterministic trials. While interesting in its own right, such a training methodology would also reduce computational cost and the need to model a domain precisely. To this end, the experiments in this paper explore incentivizing the *reactivity* of an evolved controller to encourage its robustness.

Thus these experiments require a method to evolve robot controllers. Though other methods could be applied, here the HyperNEAT neuroevolution method was chosen as a well-established representative method in ER. The next section reviews the Neuroevolution of Augmenting Topologies (NEAT) approach, the foundation of HyperNEAT.

Neuroevolution of Augmenting Topologies

The NEAT method was originally developed to evolve artificial neural networks (ANNs) to solve difficult control tasks (Stanley and Miikkulainen, 2002, 2004). Like the SAGA method (Harvey, 1993) introduced before it, NEAT begins evolution with a population of small, simple networks and *complexifies* the network topology into diverse species over generations, leading to increasingly sophisticated behavior. A similar process of gradually adding new genes has been shown in natural evolution (Martin, 1999).

However, a key feature that distinguishes NEAT from prior work in complexification is its unique approach to maintaining a healthy diversity of complexifying structures simultaneously, as this section reviews. Complete descriptions of the NEAT method, including experiments confirming the contributions of its components, are available in Stanley and Miikkulainen (2002), and Stanley and Miikkulainen (2004). This section briefly reviews the key ideas on which the basic NEAT method is based.

To keep track of which gene is which while new genes are added, a historical marking is uniquely assigned to each new structural component. During crossover, genes with the same historical markings are aligned, producing meaningful offspring efficiently. In traditional implementations of NEAT, speciation protects new structural innovations by reducing competition between differing structures and network complexities, thereby giving newer, more complex structures room to adjust. Networks are assigned to species based on the extent to which they share historical markings. It is important to note that this aspect of NEAT was altered in this paper to replace speciation in NEAT with an explicit genetic diversity objective, which achieves a similar effect. That way, NEAT is easily integrated into a multi-objective framework, as explained shortly. Finally, complexification, which resembles how genes are added over the course of natural evolution (Martin, 1999), is thus supported by both historical markings and protecting innovation, allowing NEAT to establish high-level features early in evolution and then later elaborate on them. In effect, then, NEAT searches for a compact, appropriate network topology by incrementally complexifying existing structure.

The next section reviews HyperNEAT, an extension of NEAT applied in the experiments as a representative example of a modern neuroevolution method.

HyperNEAT

Many neuroevolution methods are *directly encoded*, which means each part in the phenotype is encoded by a single gene, making the discovery of repeating motifs expensive and improbable. Therefore, indirect encodings (Bongard and Pfeifer, 2003; Hornby and Pollack, 2002; Stanley and Miikkulainen, 2003) have become a growing area of interest in evolutionary computation and artificial life.

One such indirect encoding designed explicitly for neural networks is the Hypercube-based NeuroEvolution of Augmenting Topologies (HyperNEAT) approach (Gauci and Stanley, 2010; Stanley et al., 2009), which is an indirect extension of the directly-encoded NEAT approach (Stanley and Miikkulainen, 2002, 2004) reviewed in the last section. This section briefly reviews HyperNEAT; a complete introduction is in Stanley et al. (2009) and Gauci and Stanley (2010). Rather than expressing connection weights as distinct and independent parameters in the genome, HyperNEAT allows them to vary across the phenotype in a regular pattern through an encoding called a *compositional pattern producing network* (CPPN; Stanley, 2007), which is like an ANN but with specially-chosen activation functions.

Such CPPNs are used in HyperNEAT to represent the connectivity patterns of ANNs as a *function of geometry*. That is, if an ANN's nodes are embedded in a geometry, i.e. assigned coordinates within a space, then it is possible to represent its connectivity as a single evolved function of such coordinates. In effect the CPPN paints a pattern of weights across the geometry of a neural network. To understand why this approach is promising, consider that a natural organism's brain is physically embedded within a geometric space, and that such embedding heavily constrains and influences the brain's connectivity. Topographic maps (i.e. ordered projections of sensory or effector systems such as the retina or musculature) exist within brains that preserve geometric relationships between high-dimensional sensor and effector fields (Hubel and Wiesel, 1962; Udin and Fawcett, 1988). In other words, there is important information *implicit* in geometry that can only be exploited by an encoding informed by geometry.

In particular, geometric *regularities* such as symmetry or repetition are pervasive throughout the connectivity of natural brains. To similarly achieve such regularities, CPPNs exploit activation functions that induce regularities in HyperNEAT networks. The general idea is that a CPPN takes as input the geometric coordinates of two nodes embedded in the *substrate*, i.e. an ANN situated in a particular geometry, and outputs the weight of the connection between those two nodes. In this way, a Gaussian activation function by virtue of its symmetry can induce symmetric connectivity and a sine function can induce networks with repeated elements. Note that because CPPN size is decoupled from the size of the substrate, HyperNEAT can compactly encode the connectivity of an arbitrarily large substrate.

It is important to note that HyperNEAT is chosen here simply as a representative modern neuroevolution method. Because all experiments are based on HyperNEAT, the main distinctions among them will be the use of noise or reactivity in training rather than the training algorithm or its particular details. The next section reviews multi-objective optimization, which is combined later with HyperNEAT to enable optimizing both reactivity and fitness during a single run.

Multi-objective Optimization

Multi-objective optimization is a popular paradigm within EC that addresses how to optimize more than one objective at the same time in a principled way (Coello, 1999). The experiments in this paper apply an implementation of NGSA-II (Deb et al., 2002), a well-established Pareto-based multi-objective search algorithm, to optimize a traditional fitness objective and a reactivity objective concurrently.

The concept of dominance is central to Pareto-based multi-objective search; the key insight is that when comparing two individuals over multiple objectives, if both individuals are better on different subsets of the objectives then there is no meaningful way to directly rank such individuals because neither entirely *dominates* the other. That is, ranking such mutually non-dominating individuals would require placing priority or weight on one objective at the cost of another; traditionally one individual dominates another only if it is no worse than the other over all objectives and better than the other individual on at least one objective.

In this way, the best individuals in a population are those that are not dominated by any others. Such best individuals form the *non-dominated front*, which defines a series of trade-offs in the objective space. That is, the non-dominated front contains individuals that specialize in various combinations of optimizing the set of all objectives. Some will maximize one at the expense of all the rest, while some may focus equally on all of the objectives. In this way, various tradeoffs of competing objectives such as genomic diversity, fitness, and reactivity can be explored during a single evolutionary run. The hope is that particular trade-offs between fitness performance and reactivity (i.e. policies that perform as well as possible given the constraint that they must be reactive) may lead to more robust behavior.

Recall that a detail of combining NEAT or HyperNEAT with multi-objective optimization is that NEAT has a mechanism (called speciation) for preserving genomic diversity that does not fit naturally into NGSA-II. Thus in the experiments in this paper, speciation is replaced in NEAT with an explicit genomic diversity objective that is similar in spirit. In particular, the genomic diversity of a given genome is quantified as the average distance to its k-nearest neighbors in genotype space as measured by NEAT's genomic distance measure. In this way, multi-objective evolution with NEAT is incentivized to maintain genomic diversity in a similar way to how it is in the original formulation of NEAT.

The next section formalizes the measure of reactivity that will be used as an additional objective for training.

Approach: Training for Reactivity

While other measures may also in the future prove effective for encouraging robustness, the hypothesis in this paper is that an agent that is more reactive to its environment may also be more robust. For example, a robot in a maze that is constantly probing and reacting to the walls with its rangefinder sensors as it explores may be more robust than a robot that always executes a memorized plan (which could be disrupted easily by noise). Thus what is needed is a quantification of reactivity that can be directly encouraged during evolution.

In this paper the notion of reactivity is formulated as a measure of statistical dependence between the magnitude of changes in a robot's sensors and its effectors. In general, dependence between two variables implies some kind of relationship between them (e.g. an increase in one variable may tend to result in a decrease in the other). More specifically, it implies that knowledge of one variable helps predict the other. Encouraging such dependence makes sense because it provides evidence that an agent is paying *attention* to changes in its immediate situation. In particular, it implies that the magnitude of change in a robot's sensors influences the magnitude of change of its effectors. In this way, the measure is agnostic to the exact relationship between the two because the ideal such relationship may vary between domains. However, it ensures at least that reactions to sensory changes are consistent, which aligns well with the idea of reactivity.

For example, a particularly attentive student might nod vigorously when a particularly important concept is explained but only slightly when a trivial theorem is proved. However, for the blind man tapping his cane in a corridor, any sudden large change in distance from the wall may call for caution and minor adjustment. Although such a consistent nodding or adjustment policy might not be directly necessary to solve the task, it provides *evidence* that the behavior is reactive. The particular measure of statistical dependence applied here, motivated by Ay et al. (2008), is that of *mutual information* (Shannon, 1949).

The mutual information statistic for two continuous random variables takes the following form:

$$I(X;Y) = \int_Y \int_X p(x,y) \log\left(\frac{p(x,y)}{p(x)p(y)}\right) dx\, dy, \quad (1)$$

where $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$ and $p(y)$ are the marginal probability distributions of X and Y. The higher the absolute value of $I(X;Y)$, the more dependent are the two variables.

For the experiments in this paper, reactivity is measured by the mutual information between the magnitude of changes in a robot's rangefinder sensors and the magnitude of changes in its motor effectors (unlike in Ay et al. (2008), who only measure mutual information in sensors over time). However, this approach is general enough to be applied to different sensory setups in robots in other ER domains where probing and reacting is also important to robustness. Formally, the seven rangefinder sensors $i_1, .., i_7$ of the simulated robot are subtracted from their values on the previous timestep and the average magnitude of these differences

at timestep t is recorded as x_t. The average change in the robot's outputs y_t is computed accordingly.

Because the true distributions of X and Y are not known, $p(x)$, $p(y)$, and $p(x, y)$ are estimated through histograms (with a bin width of 0.05) of the sampled data x_t and y_t collected during an evaluation. That is, three histograms are created: two one-dimensional histograms (one over x_t for $p(x)$ and one over y_t for $p(y)$), and one two-dimensional histogram (over both x_t and y_t for $p(x, y)$). Riemann sums are then applied to approximate the integrals from equation 1. However, any reasonable means of estimating the distributions or of numerical integration could be substituted.

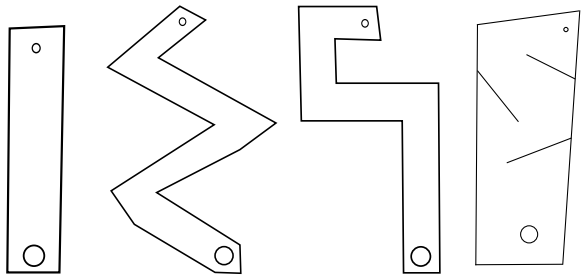
While optimizing this formalized measure of reactivity alone would not necessarily lead to successful task performance, it can alternatively be added as an *additional objective* to fitness by employing a multi-objective optimization algorithm. In this way, individuals might be evolved that both solve a given task and provide evidence of potential robustness by being reactive, without multiple noisy trials. The motivation is that if robust solutions could be evolved through this approach, computational costs would be reduced, as would the need for precisely modeling a domain (including appropriate levels of noise).

Maze Navigation Experiments

Because reactivity is intended to encourage robust behaviors, a domain for testing reactivity should be challenging under noisy conditions. Thus four maze navigation domains (figure 1) that create such a challenge in different ways are explored in this paper.

In all of the mazes, a Khepera robot controlled by an ANN must navigate from a starting point to an end point in a fixed time limit that requires direct traversal. The Straight maze (figure 1a) is designed to be simple but incorporate situations that only become *necessary* to experience when an evolved behavior is exposed to significant levels of noise. That is, although an unconditional "always go forwards" policy will be effective without noise, sufficient effector noise may cause the robot's heading to veer into walls. To further accentuate such situations, in this maze the robot is disabled for the remainder of a trial if it collides with a wall. The Zigzag maze (figure 1b) is slightly more complicated because of the need to turn, but it and the remaining mazes allow the robot to recover if it hits a wall. The Winding maze (figure 1c), with its right-angle turns and narrower corridors, creates significant opportunity for the robot to get stuck or confused with increasing noise. Finally, the most challenging maze, the Deceptive maze (figure 1d), has a deceptive cul-de-sac that may complicate training in addition to sharp corners that are difficult to navigate with noise.

The simulated robot is modeled after the Khepera III (K-Team, 2010), and training and testing noise levels are in line with established models of the robot (Cyberbotics, 2012). The robot has six rangefinders that indicate the distance to the nearest obstacle. Its three effectors produce forces that respectively turn and propel the robot. At each simulated timestep, the robot moves forward at a velocity of $9F$ centimeters per second, where F is the forward effector output. The robot also turns at $120(R - L)$ degrees per second, where R is the right effector output and L is the left effector output. The fitness of an individual is calculated as its distance to the goal at the end of the evaluation, which is a standard measure of progress in maze navigation tasks.



(*a*) Straight (*b*) Zigzag (*c*) Winding (*d*) Deceptive

Figure 1: **Domains.** The goal of the agent in the maze navigation domains is to navigate from the starting position (large circle) to the goal (small circle). Note that mazes are not drawn to scale.

Three different approaches are compared to investigate the potential of training for reactivity:

- In the **Standard** setup there is a single deterministic trial evaluated on two objectives: genomic diversity and the domain-dependent fitness measure.
- In the three **Noise** setups the objectives remain the same as in the standard setup, but each robot is evaluated in *eight* non-deterministic noisy trials to determine its fitness. The amount of both sensor and effector noise for the three different noise setups is respectively 10%, 20%, and 30%, applied as follows: Noise is computed according to the weighted average $(1.0 - x)v + x(n)$, where x is the noise level, v is the before-noise value, and n is randomly chosen from the unit uniform distribution.
- In the **Reactivity** setup an additional reactivity objective (as described earlier) complements the genomic diversity and fitness objectives. As in the Standard setup, the robot is evaluated only in a *single* deterministic trial with no noise.

Experimental Parameters

Because HyperNEAT differs from original NEAT only in its set of activation functions, it uses the same parameters (Stanley and Miikkulainen, 2002). The experiments were run with a modified version of the public domain SharpNEAT package (Green, 2006). The size of each population was 250 with 20% elitism. Asexual offspring (50%) had 0.96 probability of link weight mutation, 0.03 chance of link addition, and 0.01 chance of node addition. The coefficients

for determining genomic similarity were 1.0 for nodes and connections and 0.1 for weights. The available CPPN activation functions were sigmoid, Gaussian, absolute value, and sine. Parameter settings are based on standard SharpNEAT defaults and prior reported settings for NEAT (Stanley and Miikkulainen, 2002, 2004). They were found to be robust to moderate variation through preliminary experimentation. Runs of the Straight, Zigzag, and Winding mazes lasted 400 generations, while because of its increased difficulty runs of the Deceptive maze lasted $1,000$ generations.

Results

In *training*, the Reactivity setup did not significantly differ in performance from the other setups in the Straight or Winding mazes. However, the Reactivity setup did solve the Deceptive maze more often (in 17 out of 20 runs) than any other setup (Fisher's exact test; $p < 0.001$). In comparison, the Standard setup solved the maze in 8 runs, and the 10%, 20%, and 30% Noise setups solved the maze in 3, 1, and 0 runs, respectively. The Reactivity setup also solved the Zigzag maze significantly more often than the 20% or 30% Noise setups (Fisher's exact test; $p < 0.001$). These results support the hypothesis noise may often complicate training. However, training performance may not reflect robustness to noise; the Standard and Reactivity setups in fact both had no exposure to noise at all. It is important to note that even when a complete solution is not evolved in training, a partial evolved solution might still sometimes solve the task in the more lenient generalization test that is described next.

Because the motivation for this paper is to investigate the robustness of evolved controllers, a generalization test was devised to measure how well an evolved controller would perform in noisy distributions not encountered during training. The generalization test consisted of 50 noisy trials with the length of evaluation doubled from training to allow for greater leniency. Such leniency reflects that in transfer slight stumbles due to the reality gap are preferred to catastrophic failure (i.e. if a policy will never solve the task irrespective of how much time is allotted). An individual receives a score on the generalization test in accordance with the fraction of trials in which it is able to navigate the maze successfully (i.e. if it comes within 20 units of the goal at any time). For each run, the individual scoring the overall highest on this test from sampling the population every 100 generations is recorded (except in the Deceptive maze experiment in which every 200 generations is recorded because of its longer duration), and averaged over each of the 20 runs. This approach to testing gives a sense of the most robust controller one can hope to find with each approach. The generalization test is repeated with noise distributions from 0% to 35% at 5% intervals. Thus over five setups (three training levels of noise, standard, and reactivity) with eight testing noise levels each, there are 40 total generalization scenarios per domain, and 32 possible pairwise comparisons between Reactivity and the other setups in each domain. The results of applying this generalization test are shown in figure 2.

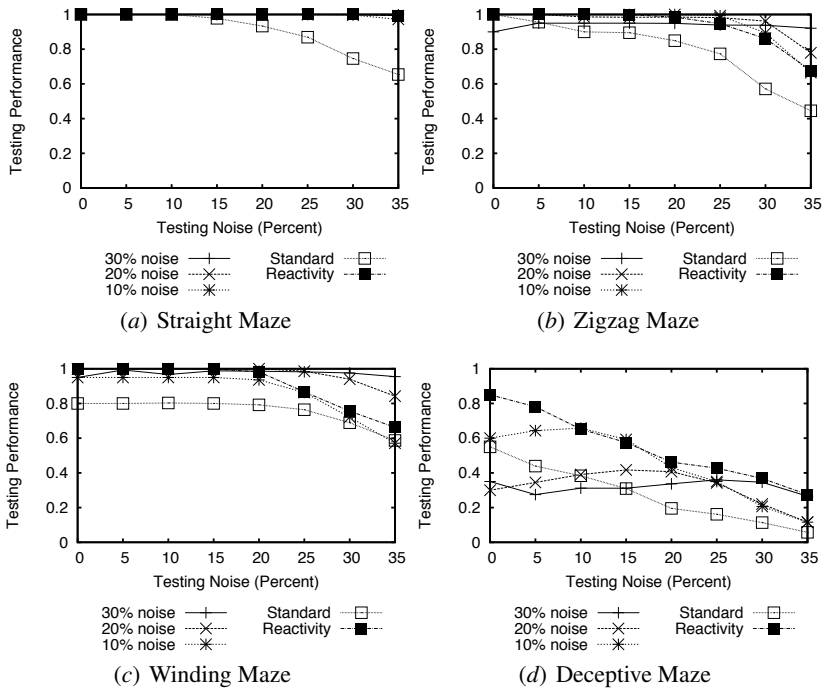
To assess statistical significance on the generalization test for each domain, a one-way ANOVA test was first applied across the five experimental setups for each level of generalization noise to demonstrate that the distributions are significantly different (at least $p < 0.05$). If at a particular noise level this first test was passed, then Student's t-tests were applied to measure the significance of pairwise differences between Reactivity and the other experimental setups.

The Straight maze, as might be expected, proved challenging only to the Standard setup because this setup provided no incentive to learn to interact with walls. Supporting its motivation, the Reactivity setup, despite not being exposed to noise nonetheless discovers policies that robustly react to walls. There were only two significant differences (among 32 total pairwise comparisons) between Reactivity and the other setups in the Zigzag maze (Reactivity was better than Standard in one scenario and 30% Noise was better than Reactivity in another), indicating perhaps that in some relatively simple domains it may make little difference what training setup is chosen. In the Winding maze, training with higher levels (20% or 30%) of noise provided a significant advantage over Reactivity for generalization with higher levels of noise ($\geq 25\%$), demonstrating that sometimes knowing the distribution of noise in reality can inform training. Finally, the Deceptive maze proved the most challenging for all methods (no method scored above 50% success on the highest noise level in the generalization test), although Reactivity was significantly better than the 30% or 20% Noise setups when testing generalization on low levels of noise ($< 15\%$). This result suggests that an inaccurate noise model can hurt noisy training while reactivity can sometimes circumvent the need for such modeling entirely.

Over all four domains, training with the Reactivity setup was never significantly worse at generalizing than training with the Standard setup, and was significantly better in 15 out of the 32 pairwise comparisons. Training with the Reactivity setup was significantly better at generalizing than the Noise setups in 7 out of 96 comparisons while Noise also was significantly better than Reactivity in 7 pairwise comparisons. Interestingly, the occasional significant advantages for the Noise setups only occurred when the noise level in the generalization test was 25% or greater, which suggests that reactivity training may generally be most advantageous when dealing with moderate levels of noise.

Discussion

The motivation for reactivity is to encourage an agent to *pay attention* to its environment and thereby make full use of its sensory experience. While ultimately the most reactive solution may not be the best performing or most robust, such reactivity may still be desirable because it can potentially act as a stepping stone on the way to a robust policy.



(*a*) Straight Maze (*b*) Zigzag Maze

(*c*) Winding Maze (*d*) Deceptive Maze

Figure 2: **Maze Navigation Generalization Test Results.** The average probability of the best individual from a run to solve the generalization test at various levels of noise is shown for different training methodologies over the four maze domains. The main result is that training with reactivity in all four domains is never significantly worse than training with noise (10%, 20%, or 30%) on the generalization test at moderate levels of noise ($< 25\%$).

The experiments in this paper provide evidence for this idea because the Reactivity setup often significantly outperforms the Standard setup in generalization testing and never underperforms it, meaning that simply encouraging an agent to be reactive often promotes robust behaviors. Additionally, reactivity is always at least as good at generalizing as the Noise setups when exposed to moderate noise, and in some cases is significantly better. Thus reactivity demonstrates that it is possible to evolve controllers that perform well in noisy situations without ever exposing those controllers to noise. In addition to providing a compelling proof of concept, reactivity also can reduce computational cost (i.e. it took eight times fewer trials per evaluation than noisy training) and the need to model a domain precisely.

One major benefit of training with reactivity is that training with noise requires several noisy trials to be run per evaluation to evaluate a behavior effectively, while reactivity can be accurately measured with a single deterministic trial. Computing an agent's reactivity does require calculating a statistical measure, but this cost is generally insignificant when compared to the computation required to simulate a domain. So even when reactivity does not outperform noisy training, it may still be preferable because of the decreased runtime. Additionally, reactivity can facilitate training robots in complex domains in which the computational costs incurred by multiple, noisy trials are prohibitive.

Another benefit of reactivity is that it can reduce the need for precise domain models. Accurately modeling a robot, its environment, and the actual levels of sensor and effector noise is often a difficult and laborious task, and perfect accuracy is generally impossible (Jakobi, 1998). However, with noisy training model accuracy can be important; selection of the right level of training noise is necessary to outperform reactivity in the Winding maze or to avoid underperforming reactivity in the Deceptive maze. Thus when training with noise, unless the model is accurate, generalization performance may be suboptimal. Interestingly, the Reactivity setup does not require a model of noise and performance degrades gracefully as the amount of noise increases. Even without any exposure to noise it is rarely significantly worse than any of the Noise setups; in as many cases it is significantly better. Thus it is possible to exploit reactivity to avoid crafting an accurate noise model, which is oftentimes difficult or time-consuming. In future work evolved reactive behaviors will be transferred to the real world to verify these potential benefits for crossing the reality gap.

While training with noise has established itself as the dominant means of producing robust controllers (Gomez and Miikkulainen, 2004; Jakobi, 1998; Koos et al., 2010; Nolfi and Floreano, 2000), the effort required to produce an accurate noise model and the computational cost of training with noise make it a kind of "necessary evil" for real-world transfer. The preliminary results in this paper demonstrate that reactivity provides an alternative to training with

noise that offers performance gains and reduced computational cost in some cases. However, there are still significant avenues for future research in this area. First, the measure of reactivity expressed in this paper is simple and intuitive: The magnitude of the change in outputs should depend on the magnitude of the change in inputs. However, more sophisticated or domain-dependent properties of evolved behaviors may exist that better encourage robustness. Additionally, reactivity could be *combined* with noisy training to further boost performance by encouraging controllers to react appropriately in noisy environments. Ultimately the results in this paper highlight that the idea of rewarding reactivity or other behavioral properties indicative of robustness is a promising research direction that merits further study.

Conclusion

This paper introduced the idea of encouraging properties of evolved controllers observable in single deterministic evaluations that correlate with increased robustness and generality. Motivated by the insight that robust behaviors tend to probe and react to their environment, the reactivity of a controller is suggested as one promising such property. Experiments showed that training with reactivity most often performs as well as training explicitly with noise, and is also significantly better as often as it is worse. The benefit is the reduced computation from considering only one deterministic evaluation and the eliminated need for accurate noise models. While the investigated measure does not always outperform training with noise, it is interesting and counterintuitive that even sometimes training without noise can be more effective in the face of noise than explicitly training with it. The conclusion is that reactivity is a viable new perspective on training for robustness that demonstrates that there may often be hints to robustness or generality hidden within single trials.

Acknowledgements

This research was supported by DARPA and ARO through DARPA grant N11AP20003 (Computer Science Study Group Phase 3), and US Army Research Office grant Award No. W911NF-11-1-0489. This paper does not necessarily reflect the position or policy of the government, and no official endorsement should be inferred.

References

Ay, N., Bertschinger, N., Der, R., Güttler, F., and Olbrich, E. (2008). Predictive information and explorative behavior of autonomous robots. *The European Physical Journal B*, 63(3):329–339.

Bongard, J. and Lipson, H. (2004). Once more unto the breach: Co-evolving a robot and its simulator. In *Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (ALIFE9)*, pages 57–62.

Bongard, J. C. and Pfeifer, R. (2003). *Evolving complete agents using artificial ontogeny*, pages 237–258. Morpho-functional Machines: The New Species (Designing Embodied Intelligence). Springer-Verlag.

Cliff, D., Husbands, P., and Harvey, I. (1993). Evolving visually guided robots. In *Proceedings of the second intl. conf. on sim. of adaptive behavior*, pages 374–383. MIT Press.

Coello, C. (1999). A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information systems*, 1(3):129–156.

Cyberbotics (2012). Webots. Commercial Mobile Robot Simulation Software.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Gauci, J. and Stanley, K. O. (2010). Autonomous evolution of topographic regularities in artificial neural networks. *Neural Computation*, page 38. To appear.

Gomez, F. and Miikkulainen, R. (2004). Transfer of neuroevolved controllers in unstable domains. *Lecture Notes in Computer Science*, pages 957–968.

Green, C. (2003–2006). SharpNEAT homepage. *http://sharpneat.sourceforge.net/*.

Harvey, I. (1993). *The Artificial Evolution of Adaptive Behavior*. PhD thesis, School of Cognitive and Computing Sciences, University of Sussex, Sussex.

Hornby, G. S. and Pollack, J. B. (2002). Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246.

Hubel, D. H. and Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160:106–154.

Jakobi, N. (1998). Minimal simulations for evolutionary robotics. Technical report.

K-Team (2010). Khepera III mobile robot. *http://www.k-team.com*.

Koos, S., Mouret, J., and Doncieux, S. (2010). Crossing the reality gap in evolutionary robotics by promoting transferable controllers. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 119–126. ACM.

Martin, A. P. (1999). Increasing genomic complexity by gene duplication and the origin of vertebrates. *The American Naturalist*, 154(2):111–128.

Miglino, O., Lund, H., and Nolfi, S. (1995). Evolving mobile robots in simulated and real environments. *Artificial life*, 2(4):417–434.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics*. MIT Press, Cambridge.

Nolfi, S. and Parisi, D. (1996). Learning to adapt to changing environments in evolving neural networks. *Adaptive behavior*, 5(1):75–98.

Shannon, C. (1949). A mathematical theory of communication. *Bell Systems Technical Journal*, 27:379–423.

Stanley, K. O. (2007). Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162.

Stanley, K. O., D'Ambrosio, D. B., and Gauci, J. (2009). A hypercube-based indirect encoding for evolving large-scale neural networks. *Artificial Life*, 15(2).

Stanley, K. O. and Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10:99–127.

Stanley, K. O. and Miikkulainen, R. (2003). A taxonomy for artificial embryogeny. *Artificial Life*, 9(2):93–130.

Stanley, K. O. and Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, 21(1):63–100.

Udin, S. and Fawcett, J. (1988). Formation of topographic maps. *Annual review of neuroscience*, 11(1):289–327.