Bridging the gap between robot simulations and reality with improved models of sensor noise

Appeared in the *Proceedings of the Third Annual Genetic Programming Conference*, 1998, edited by Koza, J.R. et. al.,

Morgan Kaufmann Publishers, San Francisco, CA, pages 824–831

Lisa Meeden
Computer Science Program
Swarthmore College
500 College Ave
Swarthmore, PA 19081 USA
meeden@cs.swarthmore.edu

ABSTRACT

Traditionally sensors have been assumed to behave independently of one another. In this paper evidence is presented that shows that for certain types of sensors this assumption of independence is incorrect. In fact, in some cases groups of sensors respond in a highly correlated fashion. A new model of sensor noise is introduced which combines independent noise with dependent noise to produce sensor responses with varying degrees of correlation. This new model is then compared to the standard model in a set of evolutionary computation experiments. The results reveal that by adopting the new model transfer of simulation results to reality is improved.

1 Introduction

In evaluating possible control structures for evolutionary robotics one has three options: use a physical robot, use a simulated robot, or use a hybrid of the two (Nolfi et al., 1994). Using a physical robot is obviously the most desirable choice but is often too time consuming. Using a simulated robot leads to the correspondence problem: How will results obtained in simulation transfer to the real world? Using a hybrid approach, one can begin the evolutionary process on a simulated robot to quickly develop a high performing population. Assuming that the simulation was developed through close observation of the actual robot, this should provide a good starting point for the slower evolutionary process on the physical robot. Also, a simulated robot can be used to prune the set of possible experiments one may want to eventually conduct on a physical robot (Dorigo & Colombetti, 1997). For these reasons, the hybrid approach to evolutionary robotics may offer the best compromise between speed and accuracy.

By adopting the hybrid approach, the correspondence problem is lessened but not eliminated. How should we construct simulations to make the transfer to reality as reliable as possible? Some specific suggestions have been made: base the simulation on carefully collected empirical data from a real physical robot; add noise to the simulated sensor readings and the actuator outcomes; and calibrate the simulation through tests on the real physical robot (Harvey et al., 1993).

Several studies have been done to investigate how best to add noise to simulations (Schultz et al., 1990) (Jakobi et al., 1995). Results confirmed the intuitive expectation: The closer the simulated environment matches the conditions expected in the real world—in terms of the amount of noise—the better the learned controllers will be. Additionally, if a perfect match is not possible, it is better to have too much noise rather than too little because learning algorithms may take advantage of spurious regularities.

The traditional method of simulating robot sensors follows these recommendations. For example, suppose that for a particular robot's light sensors in bright conditions the observed mean response is 66 and the observed range of response is 61–75. This could be modeled within a simulator by designating each light sensor value to be the observed mean plus or minus a random value based on the observed noise of 5 units. Each physical sensor is assumed to behave independently and therefore each

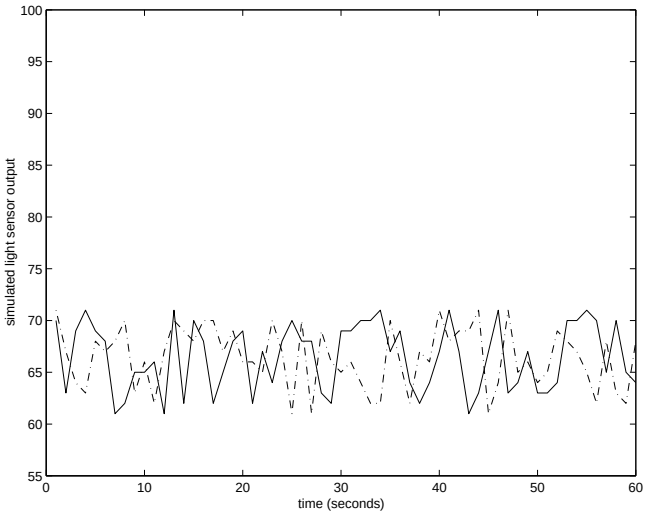

Figure 1 Independently simulated light sensor responses (correlation -29%).

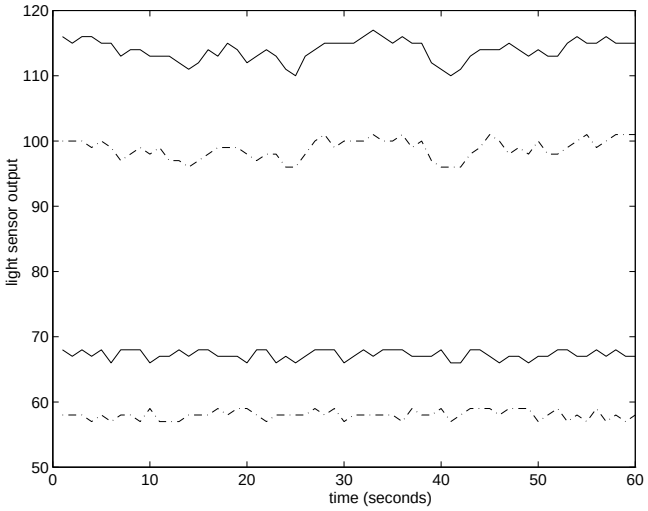

Figure 2 Physical light sensor responses: Using a 75 watt bulb placed 1 ft away (lower curves, correlation 5%) and 2 ft away (upper curves, correlation 85%).

simulated sensor reading is an independent random variable. Figure 1 shows a simulation of two such sensors' responses over time.

An examination of physical sensor data from several modalities indicates that this technique for simulating sensors may only provide a crude approximation of physical sensor behavior. Some fluctuations in physical sensor outputs are not independently random; instead, they reflect a structured variability that presumably is present in the real world. In other words, the sensor values fluctuate synchronously. There may be information in these correlations which a simulated robot has no access to but that a physical robot could exploit to improve its behavior. If robot simulations could more accurately mimic this structured type of real world noise, we might improve the transfer from simulation to reality.

In Section 2 the synchronous behavior of several types of physical sensors is examined. In Section 3 a new sensor simulation model is proposed that better reflects this synchronous behavior. In Section 4 an experiment is described that demonstrates that the new sensor simulation model enhances the transfer of results from simulation to reality as compared to the standard model. In Section 5 the results are summarized.

2 Sensor Synchrony

Three sensor modalities were examined: light, video, and sonar. The light sensors were read using the Handy Board controller (Martin, 1998) and the video and sonar were read using the Pioneer 1 Mobile Robot (ActivMedia, 1998). In each case the sensor values were recorded from a fixed location every second for a minute.

Light sensors read from the Handy Board return values between 0 and 255, with low values indicating more light. Two light sensors were placed one inch apart at a fixed location and were tested in a variety of light conditions. To ensure that the light readings were not affected by the power source, each light sensor was read from a different Handy Board operating from a separate battery supply.

Figure 2 shows the two sensor responses to a 75 watt light bulb placed one foot away and two feet away. When the bulb is further away (the darker condition), the two light sensors are 85% correlated, while in the brighter condition they are only 5% correlated. In addition, the range of values produced in the darker condition is larger than in the brighter condition. These trends continue in subsequent tests with fluorescent lights.

In Figure 3, rather than using a direct light source, the effect of ambient light was examined. The upper two curves represent the sensor response to the ambient light in the room with all the shades drawn and the overhead lights off. The lower two curves represent the sensor response to the same conditions with the overhead fluorescent lights on. Again, in the darker condition, the two light sensors are highly correlated (98%), while in the brighter condition they are less correlated (76%). The synchronous quality of these physical light sensor readings is quite dramatic, especially when compared to the traditional simulated sensors shown previously in Figure 1.

To examine the real world noise in a video image, a Pioneer robot equipped with a standard CCD video camera was aimed at a corner of a room. The values of adjacent pixels were monitored under differing lighting conditions. The examination was limited to the blue intensity component of the RGB triplet. This value ranged between 0.0 and 1.0, with 1.0 representing total blueness. The

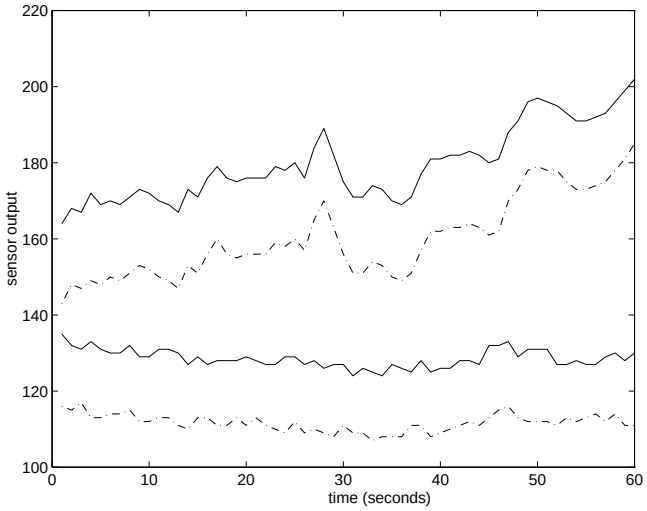

Figure 3 Physical light sensor responses: Using ambient light with overhead fluorescent lights (lower curves, correlation 76%) and without (upper curves, correlation 98%).

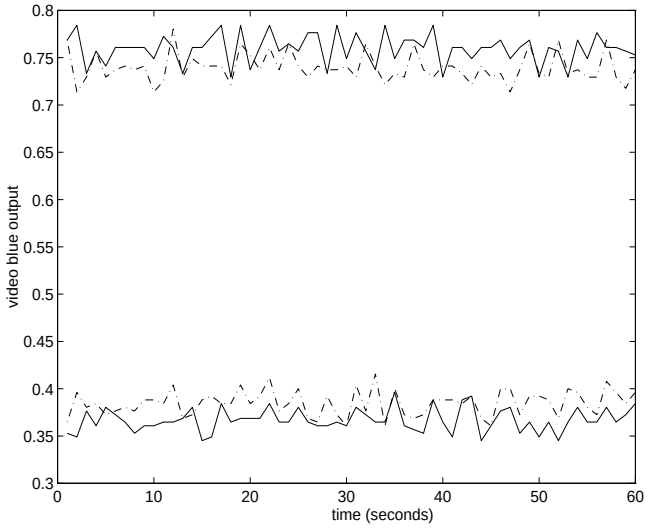

Figure 4 Video (blue component from two adjacent pixels): Using ambient light with overhead fluorescent lights (upper curves, correlation 37%) and without (lower curves, correlation 14%).

two curves in the top portion of Figure 4 represent two adjacent pixels near the middle of the scene under fluorescent lights and are 37% correlated. Likewise, the two lower curves represent the same two pixels viewing the same scene, except with the fluorescent lights turned off and are 14% correlated. Although not as strongly correlated as the light sensors, the pixel pairs do show some synchronous behavior.

To examine the real word noise in sonar responses, a Pioneer robot was positioned in front of a closed wooden door. The Pioneer's seven sonar sensors are situated in an arc around the the front of the robot, with *sonar*1 pointing to the left, *sonar*4 pointing straight ahead, and *sonar*7 pointing to the right. The sonar responses were rarely noisy producing a nearly flat profile. Perhaps this is due to the fact that sonar is an active rather than a passive sensor. By creating its own signal to measure, the effect of real world noise may be lessened. If the robot had been moving rather than at a fixed location more noise would be expected. However when there was variation in the stationary readings, the sonar sensors did show some synchrony, see Figure 5.

We have seen that in at least some modalities, the real world noise between independent sensors is correlated. The next section describes one possible method for modeling correlated noise.

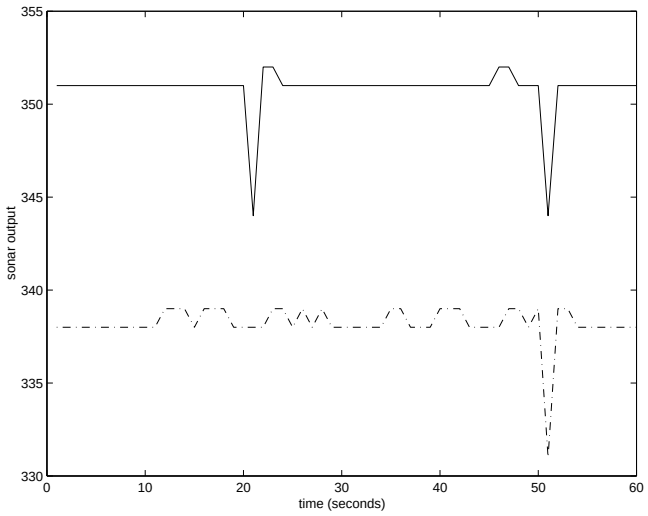

Figure 5 Sonar: Responses of a Pioneer robot placed 1 ft away from a closed wooden door (*sonar*4 and *sonar*7, correlation 65%).

3 A New Sensor Model

As the data in the previous section revealed, physical sensors exhibit a full range of interdependence, from none to nearly complete. To accommodate this wide range of possible relationships, the new sensor model divides

the overall level of noise into dependent and independent portions. For example, if the overall noise level is 10% and the sensors exhibit strong correlation, the dependent portion might be set to 8% and the independent portion to 2%. If instead the sensors exhibit weak correlation, the ratio could be reversed.

Thus the first step in constructing this new model (as before) is to observe the robot being simulated to determine the range of possible sensor values. In addition, however, one must observe which sets of sensors show some correlation and designate these as dependence groups. These dependence groups will typically be collections of adjacent sensors of the same modality. Finally, the degree of correlation within each dependence group should be noted.

The following C function is useful in calculating the noisy sensor values. Given a base sensor value v and a percentage of noise p, *getnoise* returns a random integer between $-vp$ and vp. The function *realrand*() returns a random real number between 0 and 1, inclusive. For the experiments in this paper, *realrand*() returned uniformly distributed values, but a function that produced some other distribution, such as normal, could be easily substituted.

```
int getnoise(int v, float p) {
  return (int)(2*p*v*realrand() - p*v);
}
```

To calculate a simulated sensor value requires the following steps (where n represents the sensor number):

1. For each sensor $s(n)$, get a base estimate $base(n)$ of the reading (for a light sensor this would depend on the distance of the sensor from a light source).

2. Average together the base estimates within a dependence group and obtain a shared dependent noise value for the entire group (where $d\%$ represents the percentage of dependent noise):
   $dnoise = getnoise(groupavg, d\%)$.

3. To obtain the final sensor values, combine the base estimate with the dependent noise and independent noise (where $i\%$ represents the percentage of independent noise):
   $s(n) = base(n) + dnoise + getnoise(base(n), i\%)$

By manipulating the ratio of dependent to independent noise, the degree of correlation can be controlled. For example, Figures 6–7 show how the simulation of two fixed sensors with base values of 100 and 110 can produce very different profiles simply by modifying the division of the noise. This is a relatively simple model which is not computationally expensive and it captures the kind of synchronous real world noise seen in the previous section.

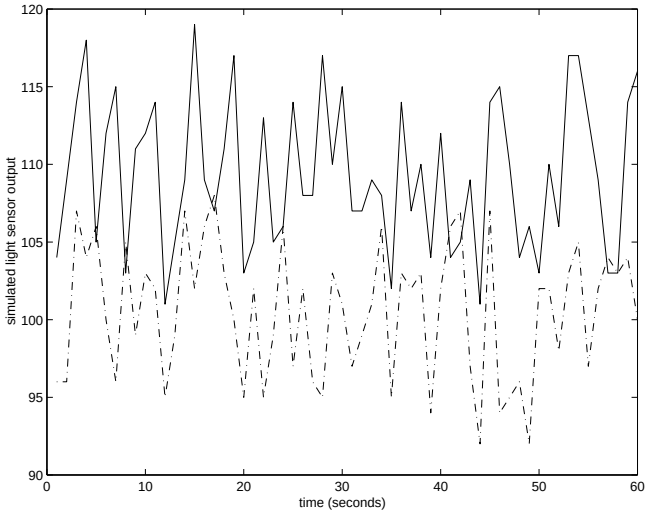

Figure 6 A low correlation of 14% created with 8% independent noise and 2% dependent noise.

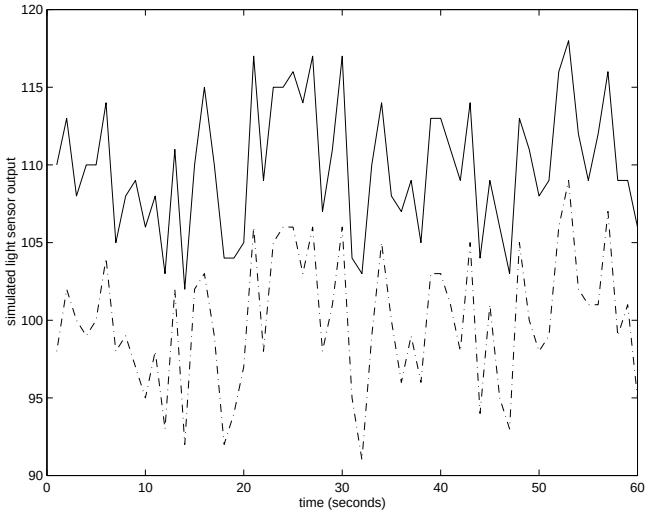

Figure 7 A high correlation of 96% created with 2% independent noise and 8% dependent noise.

4 The Experiment

Our experimental hypothesis is that a combination of dependent and independent noise is more similar to real world noise than independent noise alone, and therefore a simulation that adopts a combination model of noise will exhibit better transference than a simulation that adopts the standard independent model of noise. Because light sensors showed the strongest synchronous effects, the experiment focuses on a light task. A population of neural network controllers is evolved to solve the task. This section will describe the robot, the task, the controller, the experimental procedure, and the results.

4.1 The robot

A Khepera robot (Mondada et al., 1993) and its shareware simulation (Michel, 1997) were used. The simulation source code (written in C) was modified to allow the noise model to be manipulated. The Khepera is circular in shape and miniature (diameter 55 mm, height 30 mm, and weight 70g). It has two DC motors which power two wheels, calibrated from -10 (full reverse) to +10 (full forward) in integer increments. Its standard sensory apparatus consists of eight infra-red proximity sensors, which can also measure light. Six of the sensors are positioned in an arc around the front, while the remaining two sensors are positioned in the back. For the purposes of this experiment, only the light values of the front six sensors were used. The Khepera light values range from 0 to 525, with 0 being the brightest and 525 being the darkest.

4.2 The task

The robot's primary goal was to center itself on a light source. Its position was fixed, but the robot could modify its heading by controlling the power and direction of turning. A secondary goal was that the robot had to continually turn, i.e. the power of a turn should be nonzero. To increase the difficulty of the task, every five time steps the robot's heading was randomly perturbed. Performance was measured as the average fitness over 60 time steps.

Fitness was a real number between 0 and 1 and was the product of three factors: *balance* of the light values, *intensity* of the light values, and *turning*. For the *balance* factor, the absolute value of the difference between the average of the three left sensors and the average of the three right sensors was calculated. This difference was then adjusted, $1 - (difference/525)$, to produce values close to 1 when the light readings were balanced. For the *intensity* factor, the average of the two front sensors was calculated and then adjusted, $1 - (centeravg/525)$, to produce values close to 1 when the front light readings were very bright. Finally, the *turning* factor was defined to be 0 if the power of the turn was 0, otherwise it was 1.

Figure 8 shows the fitness performance of a successfully evolved controller during testing. Notice that the fitness drops every fifth step as the robot's heading is randomly perturbed. The average fitness for this performance test was 0.818.

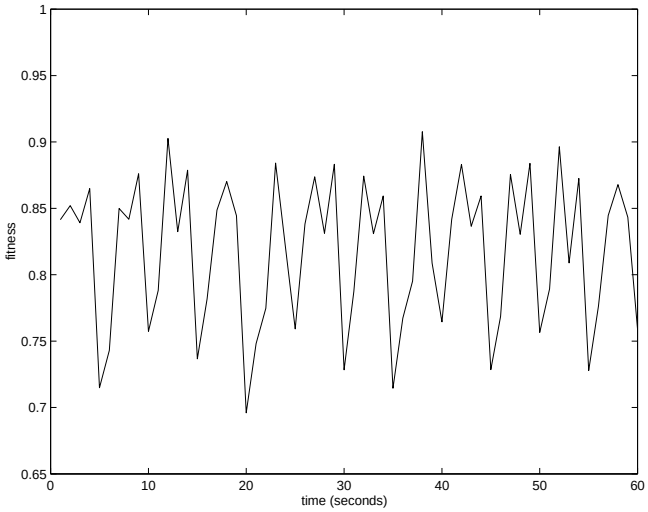

Figure 8 Fitness of a successful controller during testing.

4.3 The controller

The controller was a feed-forward neural network with a fixed architecture: six-unit input layer, two-unit hidden layer, two-unit output layer, and four biases. The input layer received adjusted light readings: $input(n) = 1 - (sensor(n)/525)$ so that brighter light created higher input values. The first output unit coded the direction of a turn, where a value less than 0.5 indicated left and otherwise right. The second output unit coded for the power of a turn from 0 to 9, by taking the real-valued output between 0 and 1, multiplying by 10, and then truncating.

4.4 The experimental procedure and results

The details of the evolutionary algorithm used are given in the appendix. Below is a high-level description of the procedure.

1. In simulation, evolve an initially random population of neural network controllers to develop a reasonable solution to the task.
2. In simulation, test the best individual found in step 1.
3. In the real world, test the best individual found in step 1.
4. In the real world, evolve a population of neural network controllers seeded with the best individual found in step 1.

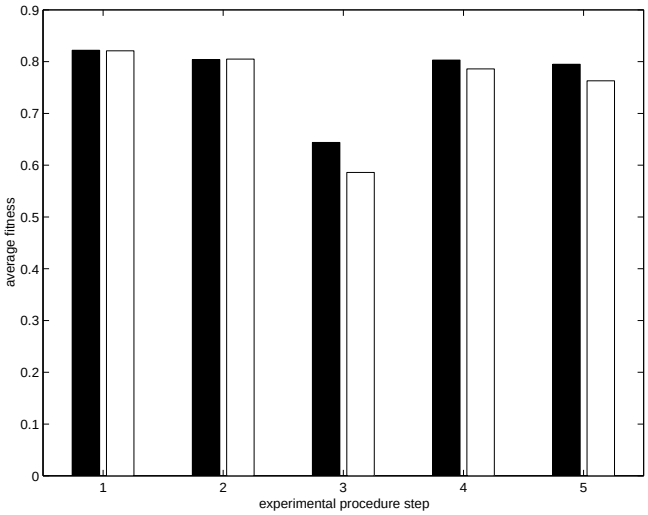

Figure 9 Comparison of the training and testing performance of the two noise models using 5% noise: black=combination noise model; white=independent noise model.

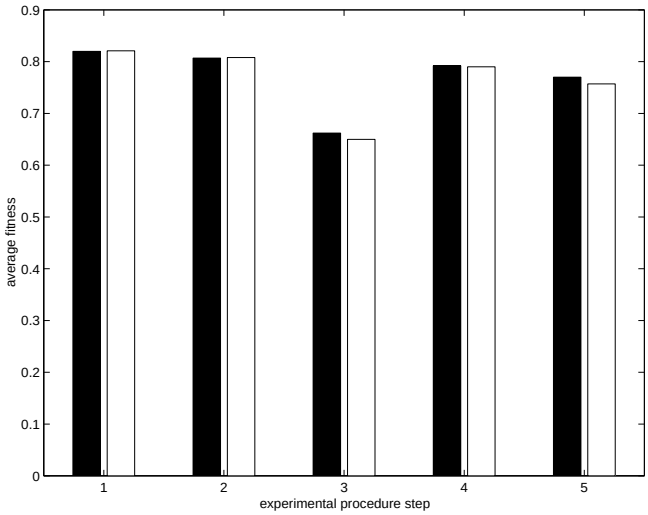

Figure 10 Comparison of the training and testing performance of the two noise models using 10% noise: black=combination noise model; white=independent noise model.

5. In the real world, test the best individual found in step 4.

For each noise model—the traditional independent model and the new combination model—ten trials of this process were completed. The average fitness across these ten trials was tracked and compared using t-tests. To prove the experimental hypothesis we must show that using the combination noise model is significantly better than the independent noise model for transferring results from the simulator to the real world. There are two points of transfer in this process—immediately after simulation training (step 3), and later after additional real world training (step 5).

Two sets of experiments were run, assuming in one case an overall real world noise level of 5% and in the other of 10%. Within the combination model, an essentially equal division of noise was used at both noise levels. Because the correlation of real world light sensors seems to depend on light intensity, an equal division should offer a good compromise for both bright and dark conditions.

At the 5% level, 5% independent noise was compared to a combination of 3% dependent noise and 2% independent noise. At the 10% level, 10% independent noise was compared to a combination of 5% dependent noise and 5% independent noise. See Figure 9 for results at the 5% level and Figure 10 for results at the 10% level. It is clear from these results that both models perform similarly at the 10% level but show more marked differences at the 5% level. Let's consider the 5% results in more detail.

The best fitnesses achieved during simulation training (step 1) and simulation testing (step 2) were almost identical for both noise models. The real world test performance (step 3) dropped off dramatically from the simulation test performance for both noise models, with the independent model showing a larger drop (though not statistically significant). After real world training (step 4), both models showed improvement, but not quite to the initial simulation level (step 1). In the final real world testing phase (step 5), controllers developed from a population seeded with the combination noise model significantly outperformed controllers developed from a population seeded with the independent noise model (0.795 versus 0.763 with $p < 0.05$).

An additional difference was also discovered for the 5% noise level. In simulation training (step 1), on average the evolutionary algorithm achieved its best fitness significantly faster ($p < 0.05$) using the combination noise model (at generation 35.0) as compared to the independent noise model (at generation 42.8).

5 Conclusions

Table 1 Ranking of each model's final real world performance average.

Rank	Noise model	Final performance
1	5% combination	0.7952
2	10% combination	0.7696
3	5% independent	0.7627
4	10% independent	0.7567

Table 1 summarizes the final real world performance averages for all the experimental variations. These rankings suggest that the combination model is a better reflection of the real world than the independent model.

Furthermore they suggest that within a model, 5% noise is a better reflection of the real world noise for this task and environment than 10% noise.

We have seen that real world noise is not independently random for certain types of sensors but displays synchrony. When a robot simulation was modified to more accurately mimic these structured fluctuations in the sensor outputs, the transfer of simulation results to reality was significantly improved. Future simulations for adaptive robotics should try to model the more complex properties of real world noise. The combination noise model, using both dependent and independent noise, is one relatively straight-forward and computationally inexpensive method for creating the right style of synchrony. There are certainly other methods that could be explored in future research.

In the introduction we argued that the hybrid approach, using both simulation and real world evaluation, is currently the most promising method for conducting evolutionary robotics. However, as the complexity of robots increases it will become more and more difficult to build faithful simulations. This does not bode well for the hybrid approach, given that the consensus view on the correspondence problem is that a simulation must be a high fidelity model of the real world for successful transfer to occur (Mataric & Cliff, 1996). Indeed it does not bode well for evolutionary robotics in general because without simulation we are left with real world evaluation which is inherently slow.

Some recent research contradicts these dire conclusions by providing a methodology for constructing minimal simulations that are easy to build, computationally cheap, include complex sensory capabilities such as vision, and yet still allow successful transfer to reality (Jakobi, 1998). The key idea is that the only features of the world that need to be modeled accurately in the simulation are those that the control system will use to perform the desired behavior. All other features can be filled in arbitrarily and should be made so unreliable that no successful control system would depend on them. Using this style of simulator we can ensure that the solution will be based on the aspects of the world we consider relevant. On the other hand we limit the possibility that the evolutionary process will opportunistically discover features that may unexpectedly prove to be relevant. On the whole, these results show promise that simulations will scale up and continue to be important tools for doing evolutionary robotics in the future.

Acknowledgements

Thanks to Doug Blank for conducting the video experiments and for his helpful discussions about the problem. Thanks to the Naval Center for Applied Research on Artificial Intelligence for hosting me during my sabbatical while I conducted this research.

Appendix

In this evolutionary algorithm, the individual neural networks were represented as real-valued strings of length 20 (6×2 input to hidden weights, 2×2 hidden to output weights, and 4 biases). The initial population could be created randomly or from a seed network. If random, each weight and bias was assigned a random value between -0.4 and +0.4. If seeded, each weight and bias was assigned the associated seed value plus a random value between -1.0 and +1.0. See (Grefenstette, 1987) for a discussion of seeding techniques.

A generation consisted of a number of tournaments equal to the population size. A tournament consisted of selecting two members from the population randomly. The member with the better fitness was designated the winner and the other the loser. The loser was replaced by a mutation of the winner, where the winner served as a seed for the new individual (just as described above). See (Meeden, 1996) for a discussion of the merits of this style of evolutionary algorithm.

In the experiment, the first phase of evolution (conducted in the simulator) consisted of 50 generations with a population of 25, while the second phase of evolution (conducted in the real world) consisted of 5 generations with a population of 10. For the first phase, the initial population was random. For the second phase, the initial population was seeded from the best individual produced in the first phase.

Bibliography

ActivMedia 1998. Pioneer 1 Mobile Robot. World Wide Web, URL is http: // www.activmedia.com / RealWorld /.

Dorigo, M. and Colombetti, M. 1997. Precis of robot shaping: An experiment in behavior engineering. *Adaptive Behavior*, 5(3–4):391–405.

Grefenstette, J. 1987. Incorporating problem specific knowledge into genetic algorithms. In Davis, L., editor, *Genetic Algorithms and Simulated Annealing*, pages 42–60. Morgan Kaufmann.

Harvey, I., Husbands, P., and Cliff, D. 1993. Issues in evolutionary robotics. In Meyer, J.-A., Roitblat, H., and Wilson, S., editors, *From Animals to Animats: Proceedings of the Second International Conference on Simulation of Adaptive Behavior*, pages 364–373, Cambridge, MA. MIT Press.

Jakobi, N. 1998. Evolutionary robotics and the radical envelope-of-noise hypothesis. *Adapdive Behavior*, 6(2):325–368.

Jakobi, N., Husbands, P., and Harvey, I. 1995. Noise and the reality gap: The use of simulation in evolutionary robotics. In Moran, F., Moreno, A., Merelo,

J., and Chacon, P., editors, *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life*, pages 704–720, Berlin, Germany. Springer-Verlag.

Martin, F. 1998. The Handy Board. World Wide Web, URL: http: // lcs.www.media.mit.edu / groups / el / Projects / handy-board /.

Mataric, M. and Cliff, D. 1996. Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems*, 19:67–83.

Meeden, L. 1996. An incremental approach to developing intelligent neural network controllers for robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 26(3):474–485.

Michel, O. 1997. Khepera simulator package version 2.0: Freeware mobile robot simulator written at the University of Nice Sophia–Antipolis. World Wide Web, URL: http: // wwwi3s.unice.fr / ~om / khep-sim.html.

Mondada, F., Franzi, E., and Ienne, P. 1993. Mobile robot miniuaturization: A tool for investigation in control algorithms. In *Proceedings of the Third International Symposium on Experimental Robots*, Kyoto, Japan.

Nolfi, S., Floreano, D., Miglino, O., and Mondada, F. 1994. How to evolve autonomous robots: Different approaches in evolutionary robotics. In Brooks, R. and Maes, P., editors, *Artificial Life IV: Proceedings of the Fourth International Workshop on the Synthesis and Simulation of Living Systems*, pages 190–197, Cambridge, MA. MIT Press.

Schultz, A., Ramsey, C., and Grefenstette, J. 1990. Simulation-assisted learning by competition: Effects of noise differences between training model and target environment. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 211–215, Austin, TX. Morgan Kaufmann.