PUNCTUATED ANYTIME LEARNING FOR EVOLUTIONARY ROBOTICS

GARY B. PARKER and KAREN J. LAROCHELLE
Computer Science, Connecticut College, New London, CT 06320
parker@conncoll.edu, kjlar@conncoll.edu

ABSTRACT

Two main issues will be addressed in this paper: anytime learning without immediate feedback and dynamic simulator accuracy in evolutionary robotics. A solution for both of these issues is the use of a punctuated anytime learning system. Training with some form of evolutionary computation takes place off-line on a simple model while periodic checks on the actual robot help to improve the learning system's output. We discuss this concept by learning gaits for a hexapod robot, which is modeled after the ServoBot. We use anytime learning with fitness biasing. A learning system that helps link the actual robot to its model during evolutionary computation by biasing the solution fitnesses computed on the model. This is accomplished by comparing the model fitnesses to the fitnesses produced when using the actual robot. Results of tests done on a simulated robot and preliminary tests done on an actual robot are reported.

KEYWORDS: genetic, anytime learning, robot, hexapod, gait, control

INTRODUCTION

Anytime learning was used by Grefenstette and Ramsey [1] to allow a learning component to continually compute a best control solution while the robot operated in the environment using the latest best solution. This system could adapt quickly to changes in the robot or the environment by having the robot's sensors continually update the status of the environment and the robot's capabilities in the learning component's simulation. This is where a difficulty develops when we try to apply this methodology to ServoBot learning. The ServoBot, a small inexpensive hexapod, lacks the computational power for an onboard learning module. Its learning system can only depend on global observation, instead of precise sensors, to determine the robot's capabilities.

Evolutionary Robotics involves the use of evolutionary computation for learning in robots. Evolutionary computation requires that a population of possible solutions be tested over several iterations. This training can be done on a model of the robot [3], entirely on the robot [2], or first on the model, then on the robot [4]. If all of the training is done off line and the results transferred to the actual robot, then significant attention must be paid to the accuracy of the model. If the fitness can be accurately judged in minimal time, all of the training can be done on line. This method precludes the need for any model of the robot, but is only applicable to simple tasks. If most of the training is done off line and then transferred to the actual robot for some remaining generations, then a less accurate model is required, but it can still take significant time to do the on line training. All of these techniques require that we either put time into the model or into the training on the actual robot. In addition, none refine the solution while the robot is in operation unless it is only doing the task being learned.

A solution for both of these issues (anytime learning without immediate feedback and dynamic simulator accuracy in evolutionary robotics) is the use of punctuated anytime learning. Training with some form of evolutionary computation takes place off-line on a simple model while periodic checks on the actual robot help to improve the learning system's output. This type of anytime learning is referred to as punctuated because the learning, although it is continuous, has break points where it is working on newly updated information about the actual robot.

In this paper, we discuss this concept by learning gaits for a simulated hexapod robot, which is modeled after the ServoBot. We use anytime learning with fitness biasing [8]. A learning system that helps link the actual robot to its model during evolutionary computation by biasing the solution fitnesses computed on the model. Results of tests on the simulated robot, plus preliminary tests on an actual robot are shown.

THE ROBOT

The task being solved by the learning system is the dynamic generation of gaits for the ServoBot robot, an inexpensive hexapod robot that has two degrees of freedom per leg. Twelve hobby servos, two per leg, provide thrust and vertical movement. A control sequence is transmitted to a controller through lines that connect to an IBM PC. The controller can store and execute the sequence of primitive instructions downloaded. Some section of the sequence of instructions can be designated for repetition and the controller will continually loop through these instructions.

The ServoBot controller is a BASIC Stamp II (sold by Parallax, Inc.). It uses an input 12 bit number (each bit represents a servo) to generate control commands to the servomotors. A signal of 1 moves the leg back if it is a horizontal servo and up if it is a vertical servo. A signal of 0 moves it in the opposite direction. An activation can be thought of as 6 pairs of actuations. Some sequence of these activations developed by the learning system will result in a pattern of leg movement that will produce a viable gait.

The model created for training was a simple data structure that held each leg's capabilities (horizontal and vertical movement limits and rates, measured before training) and current state (current horizontal and vertical position). Input activations were used with these capabilities to determine how much to change the current state of the model during training.

CYCLIC GENETIC ALGORITHMS:

Cyclic Genetic Algorithms were developed [5] to allow for the representation of a cycle of actions in a circular chromosome. The genes of the chromosome represent tasks that need to be completed in a preset amount of time. These tasks can be simple or complicated. A simple task could be a primitive representing a set of activations while a more complicated task could be cyclic sub-chromosomes, which can themselves be trained separately by a CGA. In the tests reported in this paper, the genes represent a set of servo activations to be upheld for two servo pulses or approximately 50 ms. The trained chromosome will contain the sequence of these primitive instructions that will be continually repeated by the robot's controller to produce a gait.

The effectiveness of the CGA has been tested on a fully functional robot [6] and partially disabled robots [7]. The training was done on a model of the robot while testing was done both in simulation and on the actual robot. CGA's were very successful at evolving gaits for the robots tested, but the learning system lacked the capability to make adjustments while the robot was in operation, because the measurements taken to create the model could not be adjusted.

PUNCTUATED ANYTIME LEARNING

A punctuated learning system is one where, although there is continuous learning, there are periods of accelerated learning. These periods come directly after checks on the actual robot. In this type of learning system, the learning module may not be part of the robot. As a separate unit, it does not have direct input to the controller. Updated control programs must be sent via some form of external communication. In addition, feedback cannot be sent via a direct link. It can be transmitted, such as the control program, if there is an evaluation unit on the robot. Alternatively, it may be provided to the learning module from an outside observer. The second option becomes more likely as the complexity of the robot goes down.

Anytime learning with fitness biasing [8] is a punctuated anytime learning system that employs off-line learning, using evolutionary computation, with the control program being downloaded to the on-line controller. The off-line learning does not require internal sensors but uses global observation to make the required adjustments to guide the evolutionary computation. The results of periodic tests, done on the actual robot, are used to bias the fitnesses calculated by the evolutionary computation, which uses a model of the robot. This allows the system to modify the CGA based on the performance of the robot. The model is not affected; its parameters were set before anytime learning began. The periodic checks on the actual robot alter the processing within the CGA in an attempt to improve the result of training.

Probability for selection is determined by computing each individual's fitness on the robot model. This fitness is computed for each individual in the population. After each *n* generations all solutions of the CGA population are tested on the actual robot. These measurements are used to bias the fitnesses found on the model to make them equal to the actual robot fitnesses. These biases are used as the CGA continues training. In this way, solutions that are good on the actual robot but poor on the model have boosted fitnesses during training, which results in their production of more offspring. This solution requires *population-size* actual tests every *n* generations.

Biases are computed for each solution by dividing the actual fitness by the model's fitness. Each bias is stored with its corresponding solution. It is used in subsequent generations of the CGA to alter the fitness of each solution computed on the model of the robot by multiplying this fitness by the bias. These corrected fitnesses are used for selection during the subsequent training being done by the CGA. Pairs of individuals are stochastically selected to form a new offspring through crossover and mutation. The new offspring's bias is computed by averaging the biases of its parents.

TESTS IN SIMULATION

Two tests were used to verify that punctuated anytime learning employing fitness biasing was an effective means of improving robot performance during training. Each test involved training on the model, with tests on a simulation (more accurate model) of the robot. The calculated distance (in centimeters) traveled by the simulated robot after ten seconds of activation of the current gait was recorded. Each test was done using five randomly generated starting populations. The results reported are the average performance of these five for each test. Both tests used a *population-size* of 64 and an *n* (number of generations between actual tests) of 50.

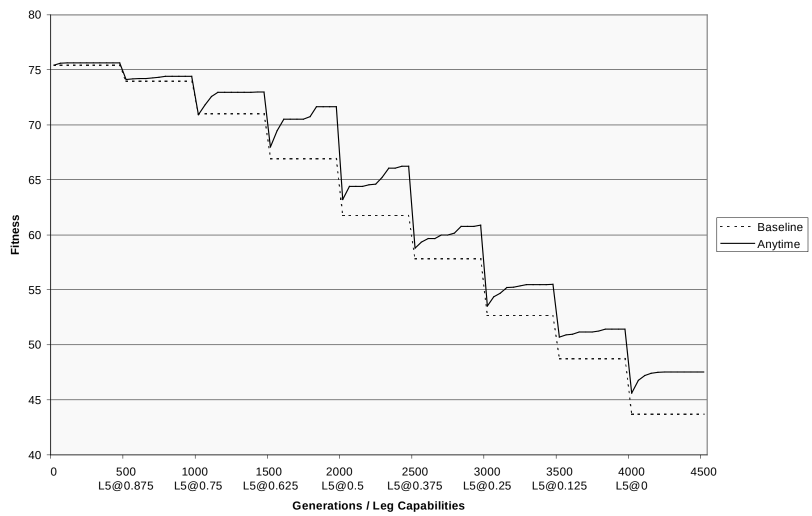

Figure 1. Decreasing capabilities test. Fitness biasing (average of five tests) versus no anytime learning.

The first test (decreasing capabilities) was a series of changes where the simulated robot's capabilities continually decreased (Figure 1). At generation 0 the simulated robot was at full capability. At each subsequent 500 generations the robot's capabilities decreased (leg 5 lost $\frac{1}{8}$ of its original capability). The model used by the CGA was a close estimate of the robot with full capabilities. This model (without anytime learning) was used for 500 generations to generate a population of gaits before generation 0 shown on this graph. The baseline shows the results of using the best gait generated in these 500 pre-graph generations (the optimal for the model) for each of the capability degradations. This would be the resulting fitness without anytime learning since the learning system would assume the model is current. The solid line shows the resultant fitnesses throughout the degradations if anytime learning with fitness biasing is being used. The best gait generated is used, but now the learning system makes some compensation for the inaccuracies in the model. As can be observed, this method significantly improved performance over the baseline.

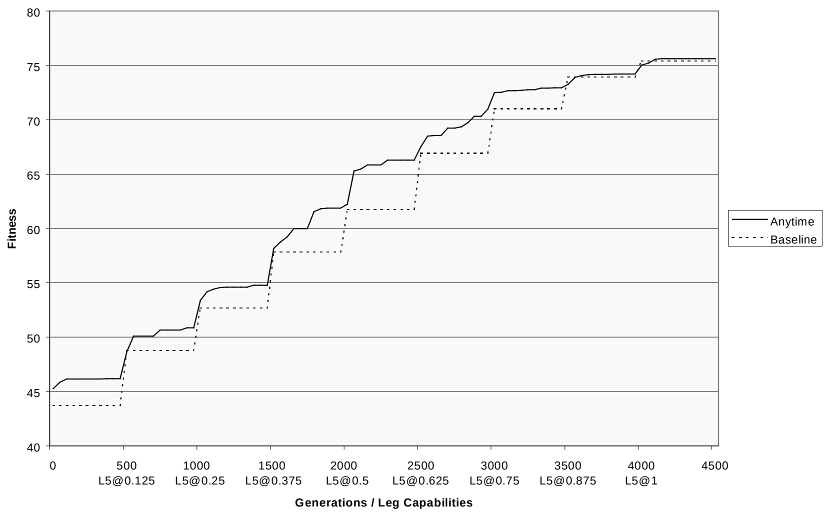

Figure 2. Increasing capabilities test. Fitness biasing (average of five tests) versus no anytime learning.

The other test (increasing capabilities) simulated the gradual return of horizontal movement capabilities of leg 5 (Figure 2). At generation 0 the simulated robot is without horizontal movement capability for leg 5. At each subsequent 500 generations the robot's capabilities increase (leg 5 gains $\frac{1}{8}$ of its original capability). The model used by the CGA is the same as what was used in the previous test. The experiment starts (generation 0) with normal gait populations. These are trained for 500 generations in the no leg 5 condition. At 500 generations the capabilities start to return. The baseline shows the results of using the best gait generated without compensating for the capability degradations as described in the previous experiment. This would be the resulting fitness without anytime learning. The solid line shows the resultant fitnesses throughout the regeneration of capabilities if anytime learning with fitness biasing is being used. As can be observed, this method was again a significant improvement over the baseline.

The system with punctuated anytime learning almost always outperformed the system without anytime learning. The notable exceptions were when capability changes initially occurred. The anytime learning system, since it had optimized for the previous capabilities, occasionally caused a drop below the baseline until it could readjust, which seldom took more than 50 generations.

PRELIMINARY TESTS ON THE ACTUAL ROBOT

Preliminary tests of this learning system have been made on the actual robot. Future plans include the construction of a learning environment where the global observation is being done by an overhead camera. Periodically, the learning system will test all of its solutions on the actual robot to create biases. The distance moved in each case will be calculated by using the camera to determine the starting and stopping positions. Since there is currently no such system, our preliminary check needed to be done though laborious work by the operator. The learning system computer would alert the operator when it was ready to test the solutions. Each was downloaded to the controller and its resultant movement was measured by the operator. The value attained was entered into the computer, which then delineated the next solution to be run on the robot.

One test was performed, and in order to reduce the workload, the population size was reduced to 16 and tests were done every 10 generations. The initial population was optimized for a robot model that had all of its horizontal movement distance measurements cut in half. This resulted in a gait on the actual robot that had short and choppy steps. Its initial distance moved in 200 servo pulses (approximately 5 sec) was 28 cm. After 150 generations of training (this was 15 generations of actual tests), it improved to attain a distance moved of 40 cm. The resultant gait was better than the CGA gait formed using the proper measurements (39 cm). The system employing punctuated anytime learning using fitness biasing had biased the learning to explore gaits that were less fit in the model to gain gaits more fit on the actual robot.

This preliminary test tends to indicate that a system using punctuated anytime learning with fitness biasing will work on actual robots to dynamically improve their performance.

ACKNOWLEDGMENTS

A portion of this research was performed at Indiana University's Adaptive Systems Laboratory, supported in part by NSF Graduate Research Traineeship Grant GER93-54898.

REFERENCES

1. Grefenstette, J. J. and Ramsey, C. L. "An Approach to Anytime Learning." *Proceedings of the Ninth International Conference on Machine Learning*, (1992), 189-195.
2. Husbands, P., Harvey, I., Cliff, D., and Miller G. "Artificial Evolution: A New Path for Artificial Intelligence?" *Brain and Cognition* v. 34, (1997), 130-159.
3. Lee, W.-P., Hallam, J., and Lund, H. "Applying Genetic Programming to Evolve Behavior Primitives and Arbitrators for Mobile Robots." *Proceedings of IEEE 4th International Conference on Evolutionary Computation*. (1997).
4. Miglino, O., Lund, H., and Nolfi S. "Evolving Mobile Robots in Simulated and Real Environments." Technical Report, Institute of Psychology, C.N.R., Rome. (1995).
5. Parker, G. and Rawlins, G. "Cyclic Genetic Algorithms for the Locomotion of Hexapod Robots." *Proceedings of the World Automation Congress (WAC'96), Volume 3, Robotic and Manufacturing Systems*. (1996), 617-622.
6. Parker, G., Braun, D., and Cyliax I. "Evolving Hexapod Gaits Using a Cyclic Genetic Algorithm." *Proceedings of the IASTED International Conference on Artificial Intelligence and Soft Computing (ASC'97)*. (1997), 141-144.
7. Parker, G. and Cyliax I. "Locomotion Control Cycles Adapted for Disabilities in Hexapod Robots." *Proceedings of the World Automation Congress (WAC '98), Volume 7, Robotic and Manufacturing Systems*. (1998), 359-364.
8. Parker, G. and Mills, J. "Adaptive Hexapod Gait Control Using Anytime Learning with Fitness Biasing." *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)*. (1999).