Evolving Gaits for the Lynxmotion Hexapod II Robot

DAVID TOTH
Computer Science, Worcester Polytechnic Institute
Worcester, MA 01609-2280, USA
toth@cs.wpi.edu, http://www.cs.wpi.edu/~toth

and

GARY PARKER
Computer Science
Connecticut College
New London, CT 06320, USA
parker@conncoll.edu, http://cs.conncoll.edu/parker

ABSTRACT

Gait generation for hexapod robots can be accomplished with a two-phase process, whereby leg cycles that effectively provide thrust are learned and then the proper selection and coordination of these cycles needed to produce a gait are learned. Genetic algorithms can be used to learn both the leg cycles and the coordination. In previous work, this technique was used successfully to generate gaits for the ServoBot. In this paper, we apply the techniques that were successful with the ServoBot to the Lynxmotion Hexapod II Robot. Although physical differences between the two robots required changes to the robot simulations, genetic algorithms and the two-phase successfully learned effective gaits.

Keywords: robotics, genetic algorithms, gait, learning, multi-legged robot, cyclic, control

1. INTRODUCTION

Autonomous robots can perform tasks that would be dangerous for humans. Examples of these tasks are mine-sweeping, espionage, and exploration. Because much of the terrain that might need to be covered is uneven, legged robots may be more appropriate than wheeled. An effective method to move a legged robot forward is required to make the robot usable for the aforementioned tasks. The generation of effective gaits for these robots is an issue and learning them saves initial programming and allows for adaptation to changes in robot capabilities. A variety of methods have been used to generate gaits for robots. Brooks used subsumption architectures to generate gaits [1]. Lewis, Fagg, and Bekey used neural net works [5]. Reinforcement learning was used by Earon, Barfoot, and D'Eleuterio [2]. Gallagher and Beer used genetic algorithms to develop a neural net to control a simulated cockroach [3].

In previous work, Parker evolved instructions for a control program that could be loaded directly into BASIC Stamp II controllers to produce gaits for a six-legged robot called a ServoBot [8]. This method was a two-phase process. The first phase was to use cyclic genetic algorithms to evolve programs, based on the individual physical characteristics of the leg, that produced cyclic movement. These leg cycles were optimized to produce thrust to propel the robot forward. A set of these leg cycle programs was developed for each leg. The second phase in the process used genetic algorithms to coordinate the leg cycles produced in the first phase. These were optimized to produce an effective gait for the robot. The coordination program determined which leg cycles to use for each leg and instructed the leg program when to begin moving and when to stop the current leg cycle and begin it again.

Applying the same two-phase gait generation process and the same leg learning and gait learning algorithms to the Lynxmotion Hexapod II Robot, another six-legged robot, did not produce effective gaits for the Hexapod II robot. This inadequate performance was the result of the physical differences between the ServoBot and the Hexapod II Robot. Changes were made to the robot model being used for evolution and subsequent tests resulted in successful generation of gaits for the Hexapod II robot. The same two-phase gait generation process worked well on both robot types once the models were changed to accommodate the robot differences.

2. THE ROBOT

The robot used in this study is the Hexapod II Robot (Figure 1), sold by Lynxmotion, Inc.. The Hexapod II Robot is a six-legged robot with two degrees of freedom per leg. Its legs are controlled by 7 Basic Stamp IIs, sold by Parallax, Inc., and 12 servomotors.

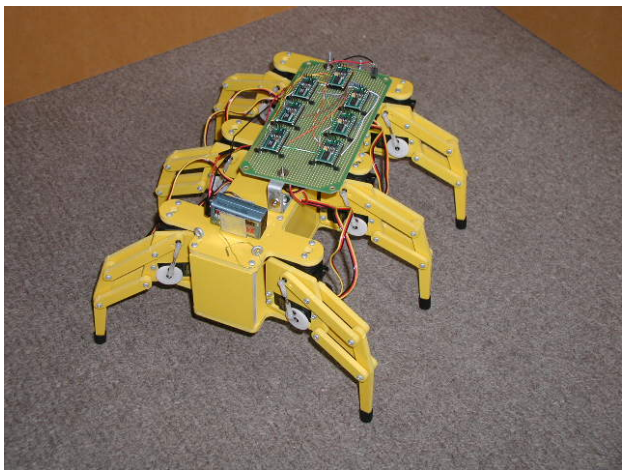

Figure 1: The Hexapod II robot configured with 7 BASIC Stamp IIs.

Six servomotors are directly attached to the legs of the robot, one per leg, to provide horizontal motion for each leg. Six more servos are attached with push/pull rods to the legs, one per leg, to provide vertical motion for each leg. Pulses sent to the servos cause them to move from their resting position to an angular position. Because the servos are attached to the legs of the robot, when the servos move, the legs of the robot move. A pulse must be sent to each servo every 25 ms to maintain positional control. The length (duration) of the pulse dictates the angular position that the servo is to attain. Servo motors can be positioned between 0 and 180 degrees and since the robot's body results in the same restriction the angle of the attached servos that control the horizontal motion of the legs is between 0 degrees and 180 degrees. Pulses between 20 and 2400 microseconds are sufficient to move these servos to both the maximum and minimum angles that they can attain, allowing the attached leg to move from one extreme horizontal position to the other. A push/pull rod is attached to each leg and the servo controlling the vertical movement of the leg. The rod restricts the vertical motion of the leg and the servo controlling the vertical motion of the leg to a tighter range. Pulses between 20 and 2400 microseconds are sufficient to move these servos to both their maximum and minimum attainable angles, allowing the attached leg to move from one extreme vertical position to the other. Each servo is distinct, so the pulse required by one to move a leg to a certain position is different than the pulses required by the other servos to move the other legs to the same position. Each servo also has a maximum number of degrees that it can change in a single pulse command (every 25 ms). If the servo is instructed to turn further than it can in a single pulse, it will turn this maximum number of degrees and not attain the directed position during that pulse.

The servomotors that control the horizontal movement of the legs of the ServoBot used in previous studies are attached to the legs with a push/pull rod, instead of a direct connection. When these servos move, the spot on the leg where the push/pull rod is attached always remains a constant distance from where the rod is attached to the servo. This causes the leg to have a range of motion that is limited by the linkage of the push/pull rod, and is thus much less than the 180 degree range of motion that the legs of the Hexapod II Robot have. The ranges of horizontal motion of the ServoBot legs are between 45 and 90 degrees. This had significant implications in the study. The servos that control the vertical movement of the legs are attached with push/pull rods, as with the Hexapod II Robot, so there is no difference between how they function on the two types of robots.

Six of the Basic Stamp IIs are used to control the six different legs. Each leg stamp sends (at 25 ms intervals) a pulse to each of the servos on the leg that it controls. These pulses are referred to as activations and a sequence of them are required to produce constant motion. A cycle of these activations is needed to produce the leg cycles needed for a gait. The stamps can each store a different cycle of activations to be sent sequentially to the servos they control. The seventh BASIC stamp II is used to coordinate the legs. It functions as a set of timers, instructing each of the other six stamps when to begin sending pulses to the servos and signaling each stamp when to restart their leg cycles. These programs can be downloaded through a modified serial cable onto the stamps.

3. INCREMENTALLY EVOLVING GAITS

The first phase of gait evolution involves generating instructions that move the legs of the robot and produce thrust to propel the robot forward. While a set of instructions will move the leg for a period of time, in order for the leg to keep moving, it must repeat the instructions continuously [4]. Therefore, the instructions need to be suitable to be repeated. The instructions are also analogous to tasks, as opposed to traits. For these reasons, Cyclic Genetic Algorithms (CGAs) are used for the first phase of the process of evolving gaits. CGAs operate like ordinary genetic algorithms, using a form of selection, crossover, and mutation, but they differ in that the genes

in the chromosomes represent tasks instead of traits. These tasks are completed in sequential order with the possibility that some portion of the chromosome can have tasks that are continually repeated. Crossover in the repeat (cyclic) section is two-point crossover. Two numbers representing gene positions are randomly selected and the new chromosome is formed by replacing the genes at the positions between those numbers in the first chromosome with the genes between those numbers in the second chromosome. The genes of the chromosome represent the instructions for the movement of the robot's legs. The leg cycle chromosomes, displayed on the left in Figure 2, consist of eight genes, where each gene is an ordered triple of integers (r_i, h_i, v_i). The number of activations that a task is to take is represented by r_i, h_i is the activation (length of a pulse) to be sent to the servo controlling the horizontal movement of a leg, and v_i is the activation (length of a pulse) to be sent to the servo controlling the vertical movement of a leg.

$((r_0, h_0, v_0)$	$(n$
(r_1, h_1, v_1)	(c_0, s_0)
(r_2, h_2, v_2)	(c_1, s_1)
(r_3, h_3, v_3)	(c_2, s_2)
(r_4, h_4, v_4)	(c_3, s_3)
(r_5, h_5, v_5)	(c_4, s_4)
(r_6, h_6, v_6)	$(c_5, s_5))$
$(r_7, h_7, v_7))$	

Figure 2: Leg cycle chromosome on the left, coordination chromosome on the right.

The second phase of gait generation is coordinating the leg cycles produced in the first phase to make the robot walk. The coordination of the legs is not a task that is repeated cyclically, like the instructions used to generate leg cycles are. The genes for the coordination chromosomes are analogous to traits, because they indicate what parameters are to be used, as opposed to tasks to be completed in a set amount of time. Therefore, regular genetic algorithms are used for the coordination of the leg cycles. The coordination chromosome, displayed on the right in Figure 2, consists of 7 genes, where the first gene is an integer n which represents the number of activations in the gait cycle, and the remaining six genes are ordered pairs of numbers (c_i, s_i) as displayed in the figure. The time that a leg will begin moving is represented by c_i and s_i indicates which of the set of generated leg cycles for that leg is used [8].

4. ROBOT DIFFERENCES FROM PREVIOUS STUDIES

In previous work, Parker used Cyclic Genetic Algorithms to produce gaits for hexapod robots. The robot used was the ServoBot, a six-legged robot that has two degrees of motion for each leg [6]. When these studies were done, pulses between 20 and 2400 microseconds were applied to the servomotors to move the robot's legs [8]. This range of pulses was sufficient to move the legs from the full forward position to the full back position and the full up position to the full down position. Measurements of the horizontal distance that the leg was from the full forward position and the vertical distance from the full down position were taken for various pulses between 20 and 2400. The distance the leg was from the full forward position was then calculated for the remaining values between 20 and 2400, using a linear method, and the vertical distance from the full down position was calculated in the same manner. For example, if a pulse of 200 puts the leg 10 mm from the full forward position of the leg and a pulse of 400 puts the leg 20 mm from the full forward position of the leg, the model will assume that a pulse of 300 will put the leg 15 mm from the full forward position of the leg. This worked well for determining the vertical position. Because the legs of ServoBots have a range of horizontal motion significantly less than 180 degrees, this was also reasonably accurate for determining the horizontal position. Due to the push/pull rod linkage and the resultant restrictions on leg movement (maximum horizontal movement range of approximately 60 degrees), the leg operated in an area where the angular movement of the servo corresponded directly to the linear thrust provided by the leg. This also allowed for a simple method of determining a leg's maximum rate of movement. Although it was actually a maximum rate that the servo could change its angle, it could be measured on the robot as the maximum rate of the linear change in the leg.

The legs of the Hexapod II Robot used in this study, however, may move 180 degrees horizontally because the servos moving them are attached directly instead of with push/pull rods like the ServoBot. The angle from the full forward position was measured for pulses of 2, 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000, 2200, and 2400. The angle of the leg from the full forward position can be approximated with the same linear method used for approximating the distance from the full forward position of the leg for the ServoBot. However, the distance from the leg's full forward position cannot be accurately approximated in the same linear manner because the sine and cosine functions, which need to be used to accurately calculate the position of the leg, are not linear. Instead, the distance from the full forward position of the leg must be calculated using the angles that were approximated. Because of this, the model of the legs used to learn efficient and effective leg cycles and the model of the robot used to learn gaits had to be modified to take into account the angles and calculate the distance of the leg from the full forward position based on the angle of the leg. By the same line of reasoning, the maximum distance that a leg can move horizontally during one

activation, which is a crucial part of the model, could not be accurately determined using the previous model. This measurement, if calculated inaccurately, would likely cause the simulation to generate gaits that were not as effective as they could be. Instead, the maximum angle that a leg can move was calculated and used in the model. The method for determining the vertical position did not need to be modified, because there is no difference in the way that the legs move vertically between the two types of robots.

5. METHOD

The work of Parker was reproduced, and three simulations of how various parts of a ServoBot would work were developed [8]. The first simulation developed leg cycles to produce thrust to move the robot forward. For each leg, a population of 64 random leg cycle chromosomes, which represented the instruction a leg would receive, was generated. Each of these chromosomes was evaluated and given a score based on the thrust that the leg would produce by repeating the cycle of instructions in the chromosome five times. The total thrust was divided by the number of activations used to produce it. After each of the chromosomes was evaluated, the best one was placed unmodified into the next generation. The other 63 chromosomes that would make up the rest of the population for the next generation were determined by stochastically selecting pairs of chromosomes based on their fitness and evolving them using the crossover methods described previously, with each bit in the new chromosomes having a 1/300 chance of mutation. This simulation was run for 500 generations, with the best chromosomes being saved to a file. After running the simulation on five distinct starting populations, the best leg cycle chromosomes were analyzed to determine an optimal number of activations needed to produce a leg cycle for each leg. This number was then used to create a set of leg cycles with different cycle lengths (numbers of activations to produce the leg cycle) for each leg. The range of the cycle lengths of these produced gait cycles was from the optimum minus 10 to the optimum plus 10.

The second simulation produced leg cycles with numbers of activations equal to each number in the desired range. A population of 64 random leg cycle chromosomes was generated, again representing the instructions a leg would receive, for each leg. The chromosomes were evaluated almost the same way they were in the first simulation, but with encouragement given for chromosomes to have a number of activations equal to the average number in the desired range from the first simulation. The simulation was run for 500 generations. If the fittest chromosome at the end of the 500 generations did not contain a number of activations in the desired range from the first simulation, then that phase of the second simulation was run again until the fittest chromosome contained a number of activations in the desired range. The fittest chromosome and the population of the 500^{th} generation were saved, and the number of activations was considered to be the desired length. In the next phase of the second simulation, one was subtracted from the desired length. The simulation took the existing population and ran again for 200 generations, encouraging the chromosomes to contain a number of activations equal to the desired length. The simulation continued running, each time subtracting one from the desired length and running for 200 generations, beginning with the current population and encouraging the chromosomes to have a number of activations equal to the desired length. When the desired length fell below the lower number in the range of activations, the saved population from the first phase was restored, as was the desired length. The simulation then added one to the desired length, took the existing population and ran again for 200 generations, encouraging the chromosomes to contain a number of activations equal to the desired length. The simulation continued running, each time adding one from the desired length and running for 200 generations, beginning with the current population and encouraging the chromosomes to have a number of activations equal to the desired length. When the desired length reached the upper limit of the range, the fittest chromosome for each desired length was written to a file. This resulted in a set of varying length leg cycles for each leg.

The third simulation coordinated the leg cycles produced by the second simulation in order to produce an effective gait for the robot. A random population of 64 coordination chromosomes was produced with each chromosome consisting of a number of activations to perform in producing the gait and six pairs of numbers. These six pairs of numbers, one for each leg, designated which leg cycle should be used from the second simulation and what time that leg should begin moving. The simulation then determined how much forward progress the robot would make for a chromosome, based on the following rules:

1. A timer would start, beginning at 0, and increment by 1 until it reached the number of activations to perform, which equaled the integer specified by the first gene in the coordination chromosome. Then it would reset to 0. This process would be repeated until the timer had increased 500 times.

2. Each leg would start moving when the timer reached the number specified by the c_i value of the gene corresponding to the leg, in the coordination chromosome.

3. Each leg would move one activation per increment of the timer, as instructed by its leg cycle. If the leg had no more instructions to perform before the timer reset to 0, it would stop moving. If the leg still had more instructions

to perform when the timer was reset, it would not do them, but instead start from the beginning of its instructions.

The score for each chromosome was equal to the forward progress the robot would make in 500 activations. This simulation was run for 2000 generations, with the next generation being produced the same way it was in the other two simulations. The fittest chromosome for generations 0, 50, 100, 200, 500, 1000, and 2000 was written to a file.

When the simulations were run using the measurements from the ServoBot, they predicted that the robot would make a good amount of forward progress and walk with a tripod gait. When the gaits generated by the simulations were tested on the ServoBot, they caused the ServoBot to move forward well, and it moved with a desirable tripod gait. When the same simulations were run using the measurements from Hexapod II Robot, they also predicted that the robot would make significant forward progress and walk with a tripod gait. However, when the gaits generated were tested on the robot, they often caused the Hexapod II Robot to turn, and thus make very little forward progress from its starting position. The simulations were then adapted using a new model with angles and trigonometric functions to calculate the horizontal distance a leg was from the full forward position. These simulations will henceforth be referred to as the angular simulations, and the unmodified simulations will be referred to as the normal simulations. The Hexapod II Robot was measured again to determine the angles of the servos when pulses of length 2, 200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000, 2200, and 2400 were applied. The maximum angle that the servos could move during an activation was calculated and the new simulations were then run using the measurements for the Hexapod II Robot. The angular simulations produced gaits that they predicted would be tripod gaits, and they predicted that the robot would make even more forward progress using the gaits generated by it than by the ones generated by the normal simulations.

6. RESULTS

Two sets of leg cycles (each set was made up of a leg cycle for each leg) were evolved using a CGA. One set was learned using the normal simulation and the other was learned using the angular simulation. The leg cycles were then used by a standard GA to evolve gaits. Each of the learning sessions was performed five times. In all cases, the learning curves were similar for the normal and angular simulations and produced what appeared to be proper gaits. Figure 3 shows a comparison of the normal and angular simulation learning curves for several intermediate solutions as the gaits evolved.

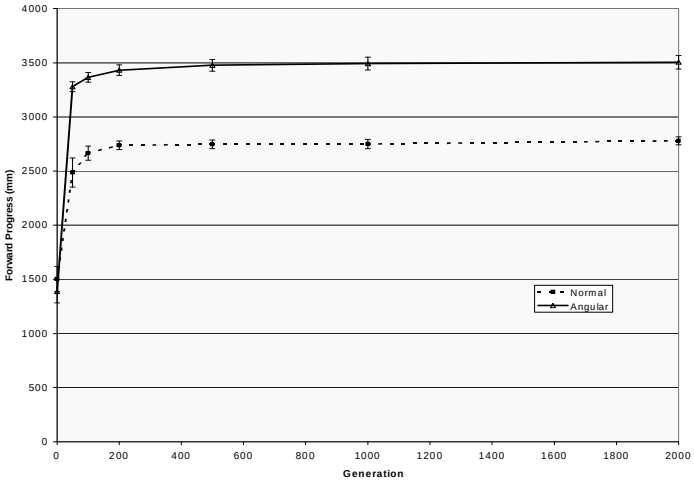

Figure 3: Average (standard error bars are shown) predicted progress for both simulation types.

These same five sets of solutions were then tested on the actual robot. Figure 4 shows the average (with standard error bars) results of these tests. As can be observed, these results are significantly different than the predicted results. Comparing the predicted to the actual for the angular simulation shows similar learning curves, but the actual has less forward progress than predicted. This is probably due to the fact that the simulation was not physically accurate. The servos were not affected by load, the legs were considered off the ground as soon as they were 1 mm up resulting in no negative friction when the legs were repositioning for another thrust, there was no slippage when the legs were on the ground predicting maximum thrust output when they were moving back, and the legs did not run into each other on the extremes. The normal simulation actual results have these same differences, but they have other significant problems. The legs do not move as predicted because the simulation is not accurate. The normal simulation does not accurately take into account what will happen when the legs are closer to their extremes. The result was unpredicted behavior. The output in the lower generations was not significantly affected because all the gaits were bad and any success was mostly random. By the later generations, the evolution is perfecting the gait with an inaccurate simulation. The resulting gaits have drag where it is not expected, which reduces their forward movement plus causes them to veer off course. Both resulted in degraded performance. Observations of the actual robots during these tests revealed that the controller produced using the angular simulation were reasonable gaits that were tripod in nature. Although the gaits generated using the normal simulations tended to be tripod in nature, they had lateral movement caused by legs dragging in the extremes, which produced slow curved tracks over the ground.

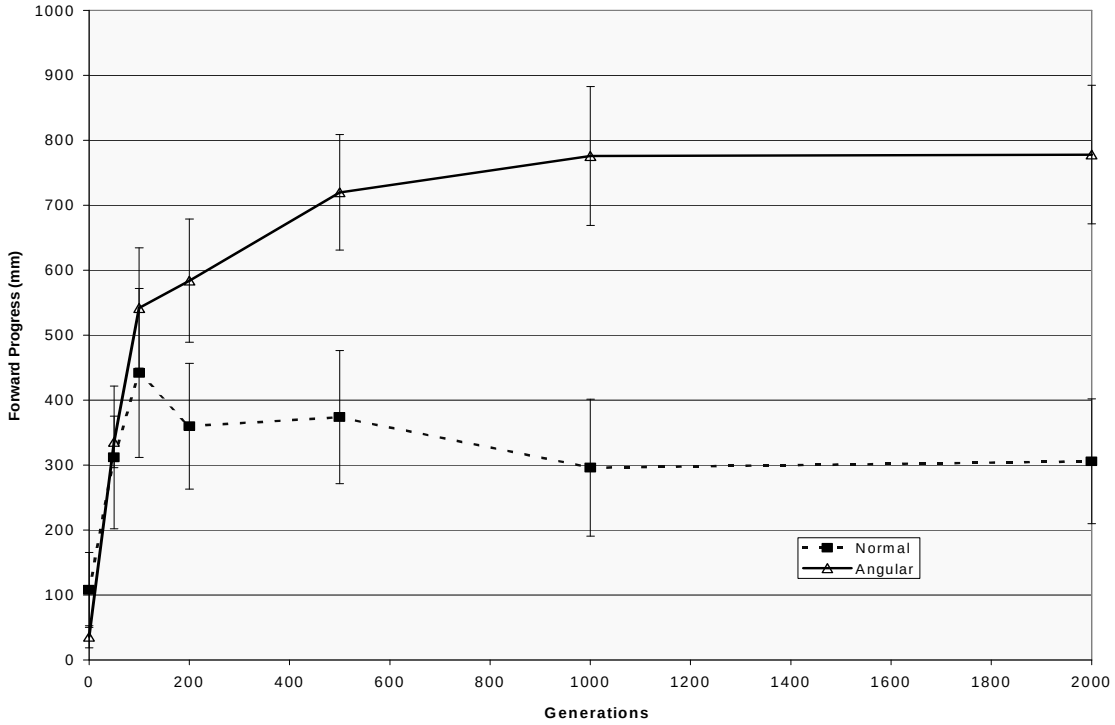

Figure 4: Average actual progress for both simulation types.

7 CONCLUSIONS

Gaits for the Lynxmotion Hexapod II Robot can be learned using a two-phase process of evolving leg cycles with CGAs and evolving gait cycles with normal genetic algorithms. The gaits are significantly more effective, causing the robot to make more forward progress, when the simulations used to generate them calculate the horizontal position of the legs using trigonometric functions and angles rather than just distance. Future work could involve including other physical factors in the simulation to get the actual and predicted results to better coincide. In addition, work could be done to modify the simulation to penalize the robot for legs colliding with each other, a problem not encountered with the ServoBot due to the limited range of horizontal motion of its legs. Another experiment could be to re-measure the ServoBot and generate gaits for it using the angular simulations to see if they produce more effective gaits than the ones produced using normal simulations.

REFERENCES

[1] Brooks, R., A Robot that Walks; Emergent Behaviors from a Carefully Evolved Network, *Neural Computation*, Vol. 1, No. 2, 1989, pp. 253–262.

[2] Earon, E., Barfoot, T. and D'Eleuterio, G, A Step in The Right Direction: Learning Hexapod Gaits Through Reinforcement, International Symposium on Robotics (ISR), 2000.

[3] Gallagher, J. and Beer, R., Application of Evolved Locomotion Controllers to a Hexapod Robot, *Technical Report CES-94-7, Department of Computer Engineering and Science, Case Western Reserve University*, 1994.

[4] Larochelle, K., Dashnaw, S., and Parker, G. B., Gait Evolution for a Hexapod Robot, *Proceedings of the Fourth International Symposium on Soft Computing and Intelligent Systems for Industry*, 2001.

[5] Lewis, M, Fagg, A., and Bekey, G., Genetic Algorithms for Gait Synthesis in a Hexapod Robot, *Recent Trends in Mobile Robots* (Y. Zheng Ed.), World Scientific Press, 1994.

[6] Parker, G. B., 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*, 1997, pp. 141-144.

[7] Parker, G. B., Evolving Leg Cycles to Produce Hexapod Gaits, *Proceedings of the World Automation Congress (WAC2000), Vol. 10, 2000*, pp. 250-255.

[8] Parker, G. B., The Incremental Evolution of Gaits for Hexapod Robots, *Proceedings of the Genetic and Evolutionary Computation Conference*, 2001, pp. 1114-1121.