Proceedings of the 3rd IASTED International Conference
ARTIFICIAL INTELLEGENCE AND APPLICATIONS
September 8-10, 2003, Benalmadena, Spain

GA-BASED GAIT GENERATION OF SONY QUADRUPED ROBOTS

Dragos Golubovic and Huosheng Hu

Department of Computer Science, University of Essex
Colchester CO4 3SQ, United Kingdom
Email: dgolub@essex.ac.uk, hhu@essex.ac.uk

Abstract

This paper presents a GA (Genetic Algorithm) approach to the development of locomotion gait for Sony quadruped robots. The selection of GA parameters such as the population size and recombination methods is made to be flexible and strive towards optimal performance autonomously. An interactive software environment with an overhead CCD camera is used to evaluate the performance of the generated gaits. The experimental results are given to show that the stable and fast gaits have been achieved.

Keyword: Evolutionary Programming, Quadruped Robots, Genetic Algorithms, Gait Generation

1. Introduction

The development of mobile robots with life-like appearance, behaviours and intelligence [1] is an extremely challenging task in robotics and AI fields. Until recently the success of the Honda Humanoid robots P2, P3 and ASIMO has demonstrated its technical feasibility [2]. At the same time, Sony quadruped walking robots, AIBO, resemble the basic behaviour of dogs or cats [3]. These robotic pets have been equipped with all the necessary hardware such as the brain, sensors and actuators, providing us with new opportunities to address robot learning [4].

To run a genetic algorithm for optimising behaviours of these AIBO robots requires setting a number of suitable parameter values [5,6], which is a non-trivial task. If poor parameter values are used, the performance of a genetic algorithm can be severely impacted. We proposed a technique for setting the probabilities of applying genetic operators during the course of a run. The technique involves adapting the operator probabilities based on their observed performance as the run takes place. We are looking for good values for all parameters, but measuring how good such values are is a challenge task and is addressed in this paper.

Two useful GA strategies were employed to find good operator probabilities. One was proposed in DeJong's thesis work [7] and the other was described by Grefenstette [5]. Both of them focused on how to determine robust parameter settings for population size, operator probabilities, and evaluation normalisation techniques. The problem of adapting genetic algorithm parameters in general was simplified such that the only chromosomal representation technique was used, and two operators were considered - mutation and crossover. We have explored these strategies in our research work to generate locomotive gait for Sony quadruped robots.

The rest of the paper is organised as follows. Section 2 presents the gait model and the choice of parameters for gait generation. In section 3, the genetic algorithm has been used to improve the performance of the gaits being generated. Experimental results are presented in section 4 to show the feasibility of the system. Finally, conclusions and future work are briefly outlined in section 5.

2. Generation of wheel-like motion

Each AIBO Robot has 20 DOF (degrees of freedom) in total: 3 on its head, 3 on each of 4 legs, 2 on its tail, 1 on its mouth and each ear. It has a CCD camera, stereo microphones, an infrared range sensor, few touch sensors, a gyroscope and two accelerometers. Its movement is controlled by a locomotion module that controls the robot's gait with a user-specified set of real-valued parameters [6]. At every 20ms it updates the joint angles of the legs to the next position in their swing phase, including detections and recovering from the robot falling over.

A trajectory refers to both the path of the movement of the tip of a limb (paw), and the velocity along the path, i.e. both spatial and temporal aspects. The spatial aspect is the sequence of the locations of the endpoint from the start of the movement to the goal, and the temporal aspect is the time dependence along the path. The essential conditions for stable dynamic walking on irregular terrain as well as on the flat ground can be itemized with physical terms: (i) the swinging legs should be able to move forward during the former period of the swinging phase; (ii) the swinging legs must be landed reliably on the ground during the latter period of the swinging phase; (iii) the angular velocity of the supporting legs during their pitching motion around the contact points at the moment of landing or leaving should be kept constant,

and (iv) the phase differences between the legs should be maintained in spite of delay of motion of a leg receiving disturbance from irregular terrain.

For the creation of wheel-like leg motion [8], we used six parameters for front and six parameters for rear legs. These twelve parameters determine the spatial aspect of a robot trajectory. The 13th parameter is bound to temporary aspect and determines the speed of paw movement. Six posture parameters used for designing the gait trajectory, which are graphically shown in Figure 1. If posture parameters for the front and rear legs are not identical, the top plane of the body will make the angle with the ground rather than horizontally. Table 1 shows 6 posture parameters used for gait generation.

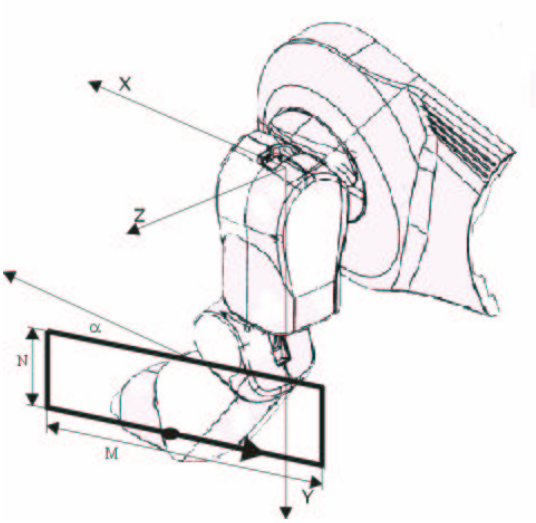

Fig. 1. Paw trajectory

Table 1. Control Parameters for Gait Generation

	Parameter	Max value (mm)	Min value (mm)
Paw motion length	m	60	20
Paw motion width	n	50	10
X coordinate of COR	Xo	70	-20
Y coordinate of COR	Yo	140	50
Z coordinate of COR	Zo	40	-10
Paw rotation angle	a	90	-90
Paw moving speed	s	600	300

In total, there are 13 real-valued parameters are used to determine a gait for the locomotion module. Table 1 lists some of these parameters, which are also the genes for individuals evolved by the genetic algorithm. These parameters specify the position and orientation of the body, the swing path and the swinging rate of the robot legs, the amplitude of oscillation of the body's location and orientation.

3. Genetic Algorithms

Genetic algorithms are stochastic search methods that mimic the metaphor of natural biological evolution. The basic operations first performed by the proposed algorithm are the generation and evaluation of an initial population. Subsequently, a main loop of operations is done, including selecting the best individuals, crossing them and applying mutation is done. The solution is usually the best string that is present in the final population.

3.1. Implementation Framework

Our algorithm operates on a population of potential solutions applying the survival principle of the fittest to produce better and better approximations to a solution. At each generation, a new set of approximations is created by the process of selecting individuals according to their level of fitness in the problem domain and breeding them together using operators borrowed from natural genetics. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation.

We model natural processes, such as selection, recombination, mutation, migration, locality and neighbourhood. Figure 2 shows the structure of a genetic algorithm we implemented. Genetic algorithms work on populations of individuals instead of single solutions. In this way the search is performed in a parallel manner.

Selection -- It determines which individuals are chosen for mating (recombination) and how many offspring each selected individual produces. In our design, roulette-wheel selection is used, which is stochastic sampling with replacement and selection of the fittest technique. More specifically, the individuals are mapped to contiguous segments of a line, such that each individual's segment is equal in size to its fitness. A random number is generated and the individual whose segment spans the random number is selected. The process is repeated until the desired number of individuals is obtained (called mating population). This technique is analogous to a roulette wheel with each slice proportional in size to the fitness.

Recombination -- It produces new individuals in combining the information contained in the parents. After recombination the new individuals of a population are created. We used 4 recombination techniques in parallel.

- Guaranteed-Uniform-Crossover -- Two children are created from two parents randomly, gene by gene.
- Guaranteed-Average -- We created one child from two parents by averaging their fields, and choose randomly which integer to use if the average lies between two integer values. Child is used only if it differs from both parents.

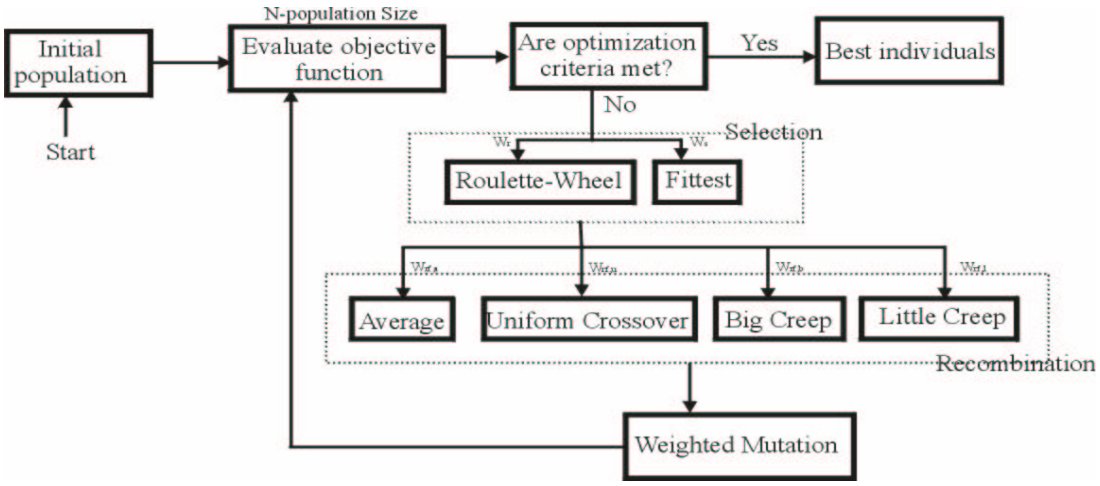

Fig. 2. Structure of genetic algorithm with a single population

- Guaranteed-Big-Creep -- We created one child from one parent by passing down the parent's fields with 20% probability. The amount and the direction of the creep are selected randomly. Child has been used if it differs from both parents.
- Guaranteed-Little-Creep -- We create one child from one parent by passing down the parent's fields with 10% probability. The direction of creep is selected randomly. We limited creep with maximum and minimal values.

Mutation -- After recombination every offspring undergoes mutation. Offspring variables are mutated by small perturbations (size of the mutation step), with low probability. The probability of mutating a variable is set to be inversely proportional to the number of variables (dimensions). The more dimensions one individual has as smaller are the mutation probability. The size of the mutation step is usually difficult to choose. The optimal step size depends on the problem considered and may even vary during the optimisation process.

3.2 The Adaptation Mechanism

A hybrid approach that adapts the probability of genetic operators described above is adopted in reproduction based on the performance of the operator's offspring. The first intuition underlying the adaptation mechanism is that it should alter the probability of applying an operator in proportion to the observed performance of the individuals created by that operator in the course of a run.

Whenever a new population is generated using selection operator Si and recombination operator Rj a pointer is created responsible for keeping statistical data of the current population. After evaluation of all generation members and their fitness functions we evaluate the performance of whole generation and store calculated values in the list. Each selection and recombination operator combination is associated with average fitness DFij, the fitness of the best member produced with these operators Mij and the weight of current operators Wij, which determines probability of appearance for those operators.

At this point we determine whether the new members were better than the current best member of the population produced with these operators. If so Mij is updated. The average fitness DFij is also updated. The average fitness and the best member of the current operators are then compared to average fitness and best members of all the other operator combinations. According to this comparison weights are modified according to Equations (1) and (2) below. All other weights are modified accordingly so sum of all weights should equal 1.

$$W_{ij} = W_{ij}' + \frac{1}{2}\left(\Delta F_{ij} - \frac{\sum_{m\neq i}^{s}\sum_{n\neq j}^{r}\Delta F_{mn}}{s*r-1}\right) + \frac{1}{2}\left(\Delta M_{ij} - \frac{\sum_{m\neq i}^{s}\sum_{n\neq j}^{r}\Delta M_{mn}}{s*r-1}\right) \quad (1)$$

$$\sum_{m\neq i}^{s}\sum_{n\neq j}^{r}\left(W_{mn} = W_{mn}' - \frac{W_{ij} - W_{ij}'}{s*r-1}\right) \quad (2)$$

where Wi,j is the weight of i-th selection and j-th recombination operator; Wi,j' is the weight of i-th selection and j-th recombination operator from the previous generation; DFij is the average fitness of i-th selection and j-th recombination operator; s is the number of selection operators (2); r is the number of recombination operators (4); Mi,j is the fitness of the best member produced with i-th selection and j-th recombination operator.

The mechanism is that purely local measures of performance are not sufficient. One must reward an operator that produces a good child, but one must also reward an operator that set stage for this production. This intuition is grounded in the sort of situation that often

happens toward the end of a run, when one's population is mostly converged. That is the reason we are using both average fitness and best fitness of the applied operators for modifying weight values.

The relative merits of crossover, mutation, and other genetic operators have long been debated in the literature of genetic algorithms. The traditional view is that crossover is primarily responsible for improvements in fitness, and that mutation serves a secondary role of reintroducing alleles that have been lost from the population. Despite this view we believe that we shouldn't make a choice between crossover and mutation but rather the balance among crossover, mutation, and selection that is important. The correct balance also depends on details of the fitness function and the encoding.

$$MV = V \pm \gamma * D * \delta \qquad (3)$$

where D is domain of variable; (search interval), V is numerical value of a gene before mutation, MV is numerical value of the mutated gene, δ is the value between 0 and 1 with probability 1/m (m - number of genes), γ is the value between 0.1and 0.9 effectively determining the extent of mutation

While selection and crossover are producing good results in terms of increasing fitness of the best member and the average fitness mutation doesn't play significant part, but when these values show signs of stalling increased value of mutation parameter γ is used in order to refresh gene set of reproductive members as shown in Equation (3). Furthermore, crossover and mutation vary in relative usefulness over the course of a run.

3.3 Evaluation Mechanism

Initially we created two members of a population. The first task of the controller is to choose operators for the creation of the entire population. Operators are weighted so that the operators with better results in the past have more chance to be chosen again. After generating entire population evaluation mechanism is up and running. The controller picks up the next member of the population and start evaluation process. Before executing the gait determined by population member genes snapshot should be taken from the CCD camera mounted 2m above the pitch [9], see Figure 3.

If the optimisation criteria are not met the creation of a new generation starts. Individuals are selected according to their fitness for the production of offspring. Parents are recombined to produce offspring. All offspring will be

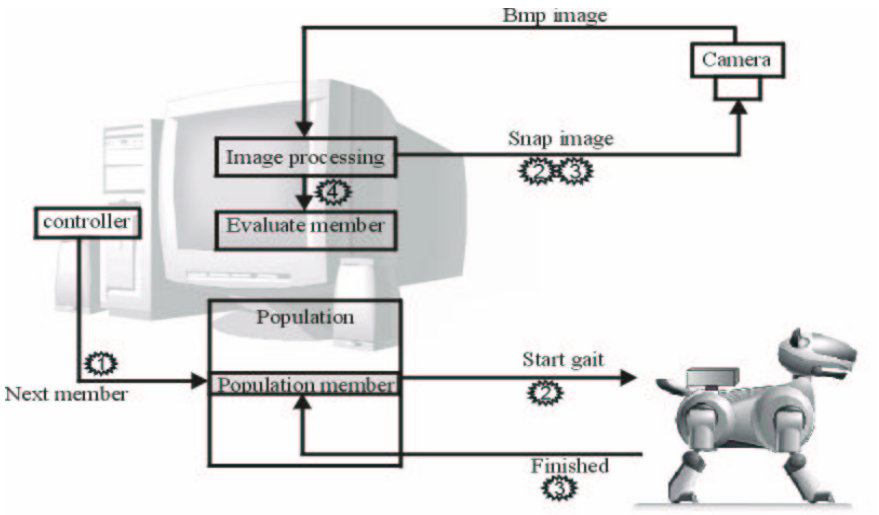

Fig. 3. Evaluation process of generation member

mutated with a certain probability. The fitness of the offspring is then computed. The offspring are inserted into the population replacing the parents, producing a new generation. This cycle is performed until the optimisation criteria are reached.

3.4 Fitness Function and Optimisation

The construction of an appropriate fitness function is very important for the correct work of the GA. This function represents the problem environment, that is, it decides how well the string solves the problem. There are two primary optimisation criteria for gait generation we have been concerned with. Our gait should be fast and stable at the same time. To measure stability during population member run we used readings from gyro-sensor. At the end of run, the variance of gyro readings is calculated from equations (4) and (5) below. The minimisation of the variance and maximisation of the achieved speed are targeted.

$$EX = \sum_{i=1}^{S} X_i \Big/ S \quad , \quad EY = \sum_{i=1}^{S} Y_i \Big/ S \quad , \quad EZ = \sum_{i=1}^{S} Z_i \Big/ S \qquad (4)$$

$$G = \sqrt{[\sum_{i=1}^{s} (X_i - EX)^2 + \sum_{i=1}^{s} (Y_i - EY)^2 + \sum_{i=1}^{s} (Z_i - EZ)^2] \Big/ s} \qquad (5)$$

where s is the number of samples from the gyro.

AIBO robot's gyro sensor returns 3 double values ranging from -9 to 9. During execution we sample readings from the gyro sensor. Our aim is to get the gait with minimal oscillations but it also should be a fast gait. Stability is important because the robot should track the ball well even when he is moving. The number of samples taken from the gyro during the robots run with one gait varies between 20 and 30.

4. Experimental Results

The evaluation takes place inside of AIBO robot's football pitch. Each generation member produces a gait that runs for 5 steps. Time necessary to finish one evaluation varies because these 5 steps won't be executed always for the same time. The 13th gene specifies the speed. If the robot falls over, it is fully capable of getting up and continuing its execution. After executing one member evaluation takes place and the speed and stability parameters are produced which qualitatively determines fitness values. The overhead camera records offset from the starting position and the changes when the robot to return to the starting position. In order to do that, the robot, e.g. forwards and backwards, executes an appropriate number of steps.

When we used a population size of 20 and run it for 50 generations, most of the population members didn't perform very well at the beginning. Some of them didn't walk in the straight line or they even walked backward. Some of them were also causing the robot to fall. By the end of the experiment, the performance of the members significantly improved, as shown in Figure 4. The best member manages to walk in the straight line with good stability and the speed of 7m per minute.

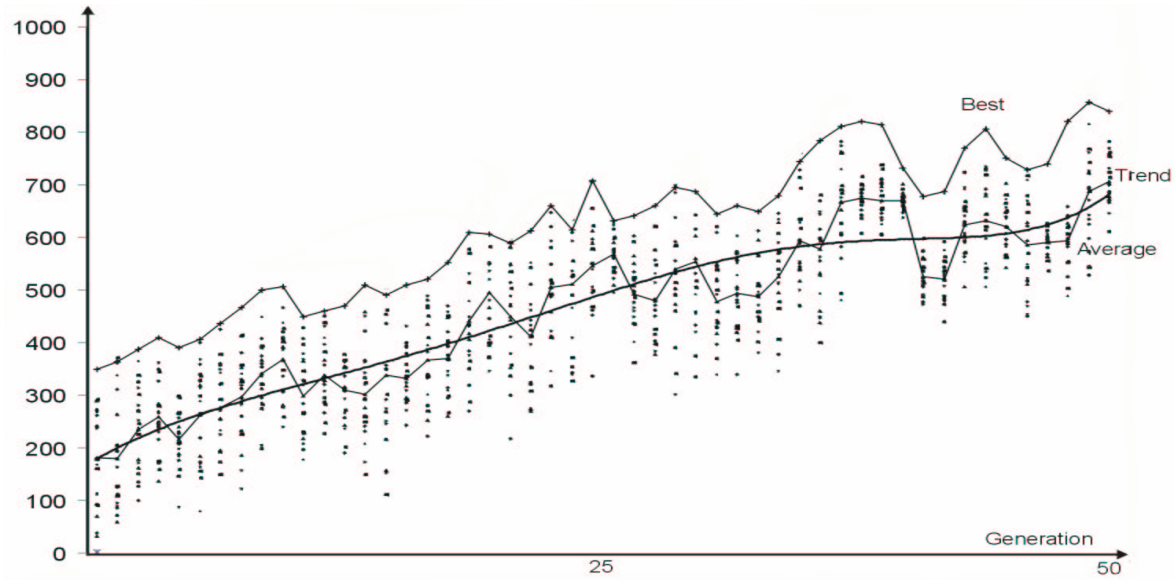

Fig. 4. The results of gait generation based on GA

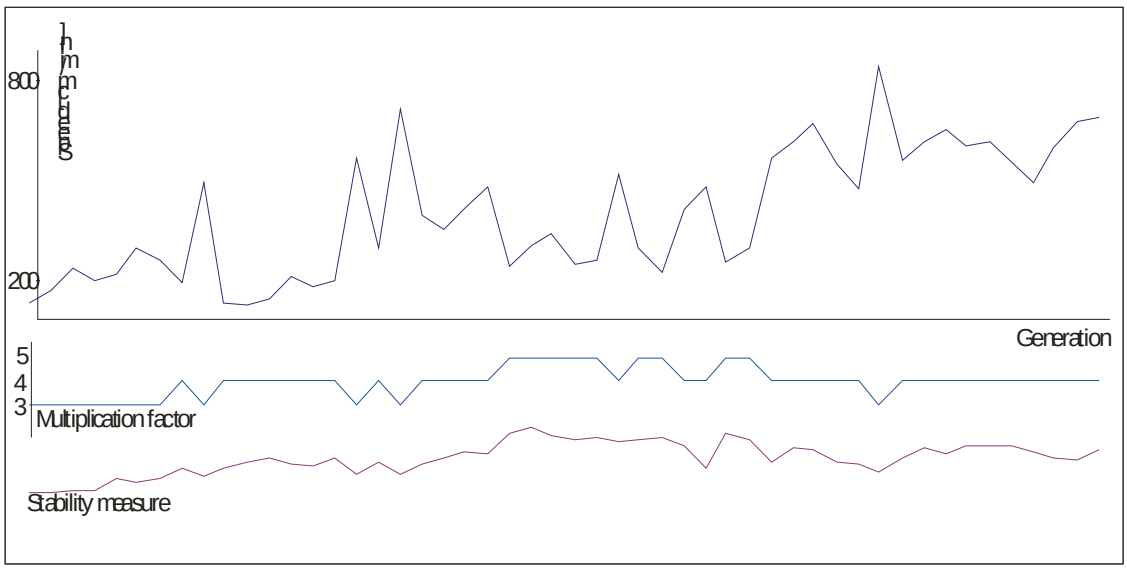

Fig. 5. The speed and stability results (separately)

We already mentioned that we are dealing with contradictory demands in terms of speed and stability. It is interesting to see the improvement of speed and stability separately over a sequence of generations and the affects it has on the overall fitness function. In this case we are observing speed and stability results of the member of population with the greatest fitness value, as shown in Figure 5

In the first 20 generations stability measure tend to have constant growth but it has negative affect on the speed increase. Still at that time speed has more than doubled its value from the start of the experiment. In the 24th generation stability reaches its maximum showing no increase and even tending to fall towards the end of the experiment. On the other hand small decrease in stability values has positive effect on the increase of speed that manages in the next 20 generations to double its value.

5. Conclusions and Future Work

This paper presents a hybrid learning system for Sony robots to learn good walking behaviours with little or no interaction with the designers. Once the learning method is put into place, the module can learn through its interaction with the world. The mutating and combination behaviours of genetic algorithms allow the development of a useful behaviour over time.

The proposed mechanism has several features. It allows rapid parameterisation of operator probabilities across the range of potential genetic algorithms and operator set. It is tailored to a steady state reproduction scheme. It would not be literally applicable to problems with noisy evaluation functions. The most significant parameter impacting solution of the test problem was not the relative probabilities of the operators. Rather, it was a parameter having to do with the selection of parents. This parameter, too, can be adapted usefully over the course of a genetic algorithm run. We will address these issues in the next stage of research.

References

[1] M. Brady and H. Hu, The Mind of a Robot, Philosophical Trans.: Physical Sciences and Engineering, the Royal Society, Vol. 349, No. 1691, London, 1994, pp. 15-28.

[2] M. Asada and H. I. Christensen, "Robotics in the home, office and playing field", in Proceedings of IJCAI, Stockholm, 30 July - 5 August 1999, pp. 1385-1392.

[3] M. Fujita, AIBO: Toward the Era of Digital Creatures, International Journal of Robotics Research, Vol. 20, No. 10, October 2001, pp. 781-794.

[4] H. Hu and D. Gu, Reactive Behaviours and Agent Architecture for Sony Legged Robots to Play Football, Int. Journal of Industrial Robot, Vol. 28, No. 1, Jan. 2001, pp. 45-53.

[5] J. Grefenstette, Optimisation of Control Parameters for Genetic Algorithms, IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-16, No. 1, January/February, 1986.

[6] D. Gu and H. Hu, Reinforcement Learning of Fuzzy Logic Controller for Quadruped Walking Robots, Proc. 15th IFAC World Congress, Barcelona, Spain, July 21-26, 2002.

[7] K. DeJong, An Analysis of the Behaviour of a Class of Genetic Adaptive Systems, PhD Dissertation, University of Michigan, 1975.

[8] M. Maza, J. Fontaine, M. Armada, P. Gonzalez, Wheel+Legs: A New Solution For Traction Enhancement Without Additive Soil Compaction, IEEE Robotics and Automation Magazine, June 1998, pp. 26-32.

[9] D. Golubovic and H. Hu, An interactive software environment for gait generation and control design of Sony legged robots, Proceedings of International Symposium on RoboCup, Fukuoka, June 2002.