

# Evolving morphologies and gaits of physically realistic simulated robots

Milton Roberto Heinen  
UFRGS – Informatics Institute  
Porto Alegre, RS, Brazil 91501-970  
mrheinen@inf.ufrgs.br

Fernando Santos Osório  
ICMC – USP  
Sao Carlos, SP, Brazil 13560-970  
fosorio@icmc.usp.br

## ABSTRACT

This paper describes our research and experiments with autonomous robots, in which were used genetic algorithms to evolve stable gaits of simulated legged robots in a physically based simulation environment. In our approach, gaits are defined using two different methods: a finite state machine based on the joint angles of the robot legs; and an Elman's recurrent neural network. The parameters for both methods are optimized using genetic algorithms, and the proposed model also allows the evolution of the robot body morphology. Several experiments are described, and the obtained results show that it is possible to generate stable gaits and efficient morphologies using machine learning techniques.

## 1. INTRODUCTION

The autonomous mobile robots have been attracting the attention of a great number of researchers, due to the challenge that this research domain proposes: making these systems capable of intelligent reasoning and able to interact with the environment in which they are inserted in, through sensor perception (infrared, sonar, bumpers, gyro, etc) and motor action planning and execution [4]. At the present time, most of the mobile robots use wheels for locomotion, which makes this task easy to control and efficient in terms of energy consumption, but they have some important disadvantages since they have problems moving across irregular surfaces and crossing borders and edges, like stairs. So, in order to make mobile robots better adapted to human environments and to irregular surfaces, they must be able to walk or have a similar locomotion mechanism used by humans and animals, i.e., they should have legs [1, 4].

However, the development of legged robots capable of moving on irregular surfaces is a quite difficult task, which needs the configuration of many gait parameters. The manual configuration of these parameters demands a lot of effort and time consuming of a human specialist, and the obtained results are usually suboptimal and specific for one robot architecture. Thus, it would be useful to generate robot

gait configurations in an automatic manner, using machine learning [18] techniques like genetic algorithms (GA) [6] and artificial neural networks (ANN) [8].

In some previous work, we made a comparative study between robots with four (tetrapod) and six (hexapod) legs [13], and also about the use and the influence of different fitness functions [9, 10] used in robot control evolution. This paper shows a comparative study between the following legged robot control strategies: (i) Control based on FSM (finite state machine); (ii) Control based on ANN. In both strategies the parameters were optimized using GAs. Besides, this paper includes the evolution of the robot morphology at the same time that the evolution of the control parameters. This paper is structured as follows: Section 2 describes some related work in control of legged robots; Section 3 describes the proposed model, called LegGen; Section 4 describes the accomplished experiments and the obtained results; and Section 5 provides some final remarks.

## 2. RELATED WORK

Control of locomotion in legged robots is a challenging multidimensional control problem [1, 4]. It requires the specification and coordination of motions in all robot legs while considering factors such as stability and surface friction [15]. This is a research area which has obvious ties with the control of animal locomotion, and it is a suitable task to use to explore this issue [21]. It has been a research area for a considerable period of time, from the first truly independent legged robots like Phony Pony [17], where each joint was controlled by a simple finite state machine, to the algorithmic control of bipeds and quadrupeds by Raibert [20].

Lewis [16] evolved controllers for a hexapod robot which learned to walk inspired on insect-like gaits. After a staged evolution, its behavior was shaped towards the final goal of walking. Bongard [2] evolved the parameters of a dynamic neural network to control various types of simulated robots. Busch [3] used genetic programming to evolve the control parameters of several robot types. Jacob [14], on the other hand, used reinforcement learning to control a simulated tetrapod robot. Reeve [21] evolved the parameters of various neural network models using genetic algorithms.

In most of these approaches described above, the fitness function used was the distance traveled by the robot in a pre-defined amount of time. Although this fitness function has been largely used, it may hinder the evolution of more stable gaits [7]. In our approach, we use in the fitness function, beyond distance traveled, sensory information (gyroscope and bumpers) to allow stable and fast gaits [10].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'09 March 8-12, 2009, Honolulu, Hawaii, U.S.A.  
Copyright 2009 ACM 978-1-60558-166-8/09/03 ...\$5.00.

### 3. PROPOSED MODEL

The LegGen simulator<sup>1</sup> [10, 12, 13] was developed to accomplish the gait control of simulated legged robots in an automatic way. This simulator is composed of several modules, showed in Figure 1. The *Robotnik* module is responsible for the robot and virtual environment creation using a physics simulation library called ODE<sup>2</sup> (Open Dynamics Engine). The *Evolution* module is responsible for the evolution of the control parameters using genetic algorithms. The *Sensorial* module is responsible for sensory information reading during simulation and fitness calculation for each individual. The *Viewer* module is responsible for the visualization of results in a three-dimensional graphic environment. The *Controller* module is responsible for the robot joint control, which is accomplished using two strategies: (i) a finite state machine (FSM); (ii) an artificial neural network (ANN). The following sections describe these control strategies.

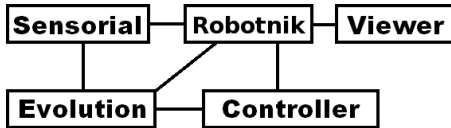


Figure 1: The LegGen modules

#### 3.1 Finite state machine control

In the LegGen simulator, the gait control is generated using a finite state machine (FSM), in which is defined for each state and for each robot joint their final expected angles configuration [2]. In this way, the controller needs to continually read the joint angles state, in order to check if the joint motor accomplished the task. Real robots do this using sensors (encoders) to control the actual angle attained by the joints [1, 4]. So, in this approach the gait control is accomplished in the following way: initially the controller verify if the joints have already reached the expected angles. The joints that do not have reached them are moved (activate motors), and when all the joints have reached their respective angles, the FSM passes to the following state.

To synchronize the movements, it is important that all joints could reach their respective angles at almost the same time. This is possible with the application of a specific joint angular velocity for each joint, calculated by the equation:

$$V_{ij} = Vr_i(\alpha_{ij} - \alpha_{ij-1}) \quad (1)$$

where  $V_{ij}$  is the velocity applied to the motor joint  $i$  in the  $j$  state,  $\alpha_{ij}$  is the joint angle  $i$  in the  $j$  state,  $\alpha_{ij-1}$  is the joint angle  $i$  in  $j - 1$  state, and  $Vr_i$  is the reference velocity of the  $i$  state, used to control the set velocity. The reference velocity  $Vr$  is one parameter of the gait control that is also optimized by the genetic algorithm. The other parameters are the joint angles for each state. To reduce the search space, the GA only generates values between the maximum and minimum accepted values for each specific parameter.

#### 3.2 Neural control

Besides finite state machines, LegGen can use artificial neural networks (ANN) [8] to control the robot joints. This approach has some important and specific limitations: it is

<sup>1</sup>LegGen – <http://www.inf.ufrgs.br/~mrheinen/leggen>

<sup>2</sup>Open Dynamics Engine (ODE) – <http://www.ode.org>

quite difficult to have a priori information about the generation of the control parameters [11, 12]. Since we do not have available the exact and correct sequence of values that should be sent to control the actuators, then it is usually not possible to apply traditional supervised learning algorithms, like back-propagation and other similar ones. This was the main reason to adopt GA to evolve the synaptic weights.

GA can adjust synaptic weights with the advantage that they do not need any local information or local error measure in order to adapt the weights, and so we do not need a training dataset (supervised learning). The weights can be coded into the chromosomes and evolved, using a fitness function to evaluate the robot performance controlled by this evolved ANN. On the other hand, ANN have some advantages when used to control robot legs: they are more robust to noise, i.e., continue to perform well even when faced to unseen situations; and they usually can obtain a good generalized behavior.

The ANN inputs are the present robot joint angles values (angles at time  $t$ , normalized in the range from  $-1$  ( $\alpha_{min}$ ) to  $+1$  ( $\alpha_{max}$ ). In the ANN outputs are obtained the joint angles in the next time step  $t + 1$ , also normalized in the range  $[-1; +1]$ . After some preliminary tests, we choose the Elman model of recurrent ANN, which was very satisfactory when applied in this problem where we need to predict a temporal behavior (sequencing joint angles). The Elman networks are MLP nets with feedback connections from and back to the hidden layer. These connections allow the Elman nets to learn temporal sequences of patterns and then, from the joint angles patterns in time  $t$ , they can generate the next joint angles pattern in their outputs. We adopted the hyperbolic tangent function as the ANN activation function, and also the synaptic weights were limited ranging from  $-1$  to  $+1$ , which simplify the GA weights optimization. This ANN model and parameters setup was empirically tested and showed to be well suited to the problem in question.

#### 3.3 Evolution

In our model, the control parameters are evolved using genetic algorithms. The GA implementation used in our system was based on the GALib software library<sup>3</sup>, developed by Matthew Wall of Massachusetts Institute of Technology (MIT). In the LegGen simulator, a GA as described in [6] was used, and a floating point type genome was adopted. In order to reduce the search space, alleles were used to limit generated values only to possible values for each parameter. Table 1 shows the parameter values used by GA.

Table 1: Parameters of the LegGen simulator

Par-ID	Parameter	Value
1	One point crossover	0.80
2	Mutation rate	0.08
3	Population size	350
4	Number of generations	700

The fitness evaluation uses the following sensory information that must be calculated: (a) the distance  $D = x_1 - x_0$  covered by the robot in the  $x$  axis, where  $x_0$  is the  $x$  start position and  $x_1$  is the end  $x$  position; (b) the instability measure  $G$ , calculated using the robot position variations in the  $x$ ,  $y$  and  $z$  axis. These variations are collected during the

<sup>3</sup>GALib – <http://www.lancet.mit.edu/ga/>

physical simulation, simulating a gyroscope sensor, which is a sensor present in some modern robots [4]. The instability measure  $G$  (Gyro) is then calculated by [7]:

$$G = \sqrt{\frac{\sum_{i=1}^N (x_i - \bar{x}_x)^2 + \sum_{i=1}^N (y_i - \bar{x}_y)^2 + \sum_{i=1}^N (z_i - \bar{x}_z)^2}{N}} \quad (2)$$

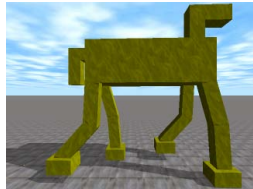
where  $N$  is the number of sample readings,  $x_i, y_i$  and  $z_i$  are the data collected by the simulated gyroscope in the time  $i$ , and  $\bar{x}_x, \bar{x}_y$  and  $\bar{x}_z$  are the gyroscope reading means:

$$\bar{x}_x = \frac{\sum_{i=1}^N x_i}{N}, \quad \bar{x}_y = \frac{\sum_{i=1}^N y_i}{N}, \quad \bar{x}_z = \frac{\sum_{i=1}^N z_i}{N} \quad (3)$$

After finishing the sensory information processing, the fitness function  $F = D/(1 + G)$  is then calculated. Analyzing this fitness function, we see that the individual better qualified will be the one that has the best relationship between velocity and stability, so the best solutions are those that move fast, but without losing the stability.

### 3.4 Modeled robot

The robot model used in great part of the experiments is shown in Figure 2. Its dimensions are approximately the dimensions of a medium sized dog. The joint restrictions used in the simulated robot are similar to its biological equivalent, with the following values: Hip=[-60°;15°]; Knee=[0°;120°]; Ankle=[-90°;30°]. All legs have these same joint restrictions.



Dimensions			
Part	x	y	z
Body	45.0cm	15.0cm	25.0cm
Thigh	5.0cm	15.0cm	5.0cm
Shin	5.0cm	15.0cm	5.0cm
Paw	8.0cm	5.0cm	9.0cm

Figure 2: Modeled robot

### 3.5 Morphology evolution

According to Pfeifer [19], in the nature the evolution of the control (nervous system) does not occurs independently of the body morphology evolution. Instead, this is a process that happens at same time. This strategy is used a lot in the artificial life area [5, 22]. In the previous section, the robot model used in our previous work was described. This robot was modeled in an empirical way, inspired in four leg animals, but with some simplifications. But when the morphology and the control parameters are evolved at same time, this makes it possible to discover new robot models, without a biological equivalent, but equally or more efficient [19]. Thus, LegGen was extended to allow the evolution of the robot morphology at the same time that the evolution of the control parameters. To make this possible, new genes were included in the GA, which encodes the robot segments using three floating point values ( $x, y$  and  $z$  dimensions).

## 4. RESULTS

This section describes two set of experiments accomplished with the proposed model to: (i) compare the control based on FSM and the control based on ANN (Subsection 4.1; (ii) compare the evolution of just the control parameters and the

evolution of the robot morphology and control parameters at the same time (Subsection 4.2. The following sections describe these experiments.

### 4.1 FSM control x ANN control

This subsection describes the experiments accomplished in order to evaluate the robot behavior in both control strategies (FSM and ANN), as described in the previous sections. For each control strategy, we executed 10 different experiments, which are presented here. Table 2 shows the obtained results, where we can see each control strategy (FSM and ANN) and the values of the fitness, distance and gyro instability measure respectively ( $F, D, G$ ) indicated for each experiment (E) in both strategies. The last two rows in Table 2 show the mean ( $\mu$ ) and the standard deviation ( $\sigma$ ) computed over these 10 experiments.

Table 2: Evaluation of the control strategies

E	FSM			ANN		
	F	D	G	F	D	G
1	14.04	32.17	0.128	16.27	29.19	0.079
2	14.28	32.38	0.126	16.63	28.31	0.070
3	13.18	30.33	0.129	16.99	27.85	0.063
4	15.87	26.81	0.069	16.68	27.91	0.067
5	16.64	36.60	0.120	16.16	28.20	0.074
6	16.48	27.69	0.068	15.97	31.13	0.093
7	14.88	31.69	0.112	17.33	29.63	0.070
8	13.77	29.02	0.110	16.65	29.04	0.074
9	15.33	34.41	0.124	16.29	30.15	0.085
10	15.80	37.01	0.134	16.23	29.81	0.083
$\mu$	<b>15.03</b>	<b>31.81</b>	<b>0.112</b>	<b>16.52</b>	<b>29.12</b>	<b>0.076</b>
$\sigma$	<b>1.19</b>	<b>3.48</b>	<b>0.024</b>	<b>0.42</b>	<b>1.08</b>	<b>0.009</b>

In the experiments using the FSM, we fixed the number of states to four. In the experiments using the neural network we adopted a network with three neurons in the hidden layer. These parameters were defined after a careful preliminary study based on experiments. Figure 3 shows the boxplot graph and the 95% confidence interval (CI) of the fitness values obtained in Table 2 experiments.

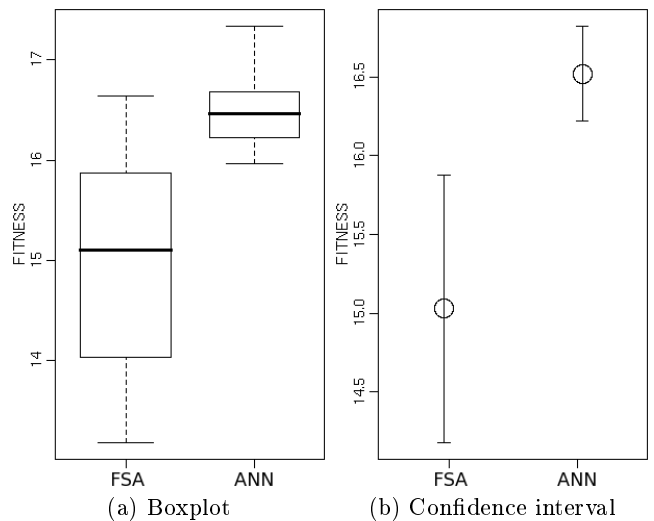


Figure 3: Evaluation of the control strategies

According to Figure 3 we can affirm that the results obtained by the ANN are clearly superior to those obtained using the FSM, since the confidence intervals are not superposed. Besides, the results obtained using the FSM are more unstable with a large variability. Figure 4 compares the fitness improvement of the population during the evolution (best and mean fitness) obtained for each control strategy. The experiments showed in this figure are those that achieved the best results in our simulations.

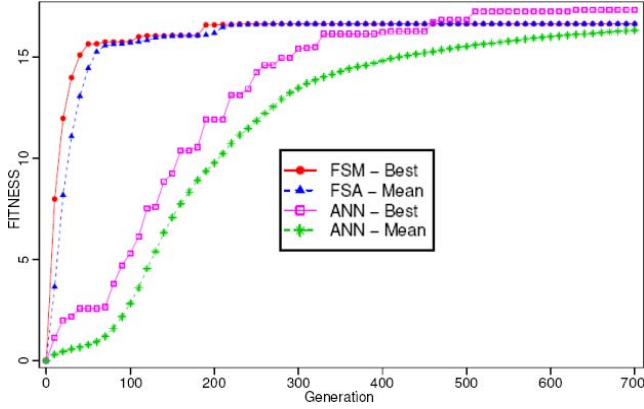


Figure 4: Evolution progress

It is clear that the evolution of the neural control parameters need more generations (epochs) in order to achieve good results. This is due to the ANN search space (44 synaptic weights) larger than the FSM search space (3 parameters). Figure 5 shows a robot controlled by a FSM and Figure 6 shows a robot controlled by an ANN.

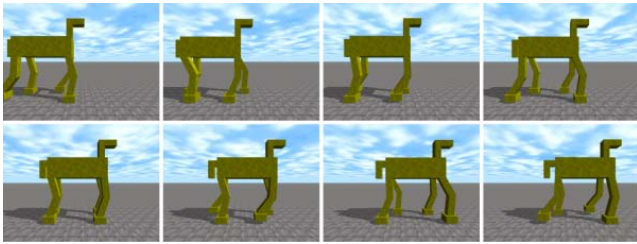


Figure 5: FSM robot control

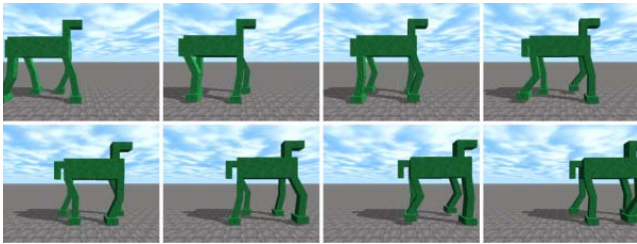


Figure 6: ANN robot control

## 4.2 Morphology evolution x just control

This section describes the experiments accomplished in order to evaluate the morphology evolution importance. We

executed 10 different experiments with the ANN controller (i) evolving just the control parameters and (ii) evolving the robot morphology and control parameters at the same time. Table 3 shows the results obtained in these experiments. The first column (**E**) describes the individual experiment index. The next columns show the values of the fitness function ( $F$ ), distance ( $D$ ) and gyro instability measure ( $G$ ), respectively. The last two rows in Table 3 show the mean ( $\mu$ ) and the standard deviation ( $\sigma$ ) indicated over the 10 experiments. We spent a total of 149.22 hours processing

Table 3: Importance of morphology evolution

<b>E</b>	Just control			Morphology and control		
	$F$	$D$	$G$	$F$	$D$	$G$
1	16.26	29.19	0.079	18.80	38.03	0.101
2	16.64	28.31	0.070	17.90	32.96	0.075
3	16.99	27.85	0.063	19.84	39.52	0.099
4	16.68	27.92	0.067	17.80	37.86	0.112
5	16.16	28.20	0.074	20.09	27.41	0.031
6	15.96	31.13	0.093	15.90	32.80	0.105
7	17.34	29.63	0.070	18.87	41.13	0.117
8	16.65	29.04	0.074	18.50	36.22	0.095
9	16.29	30.15	0.085	19.08	39.16	0.105
10	16.23	29.81	0.083	15.57	37.40	0.140
$\mu$	<b>16.52</b>	<b>29.12</b>	<b>0.076</b>	<b>18.24</b>	<b>36.25</b>	<b>0.098</b>
$\sigma$	<b>0.42</b>	<b>1.08</b>	<b>0.009</b>	<b>1.50</b>	<b>4.10</b>	<b>0.029</b>

the final experiments of Table 3. Figure 7 shows the box plot graph and the confidence interval (CI) of 95%, related to the fitness values obtained in Table 3 experiments.

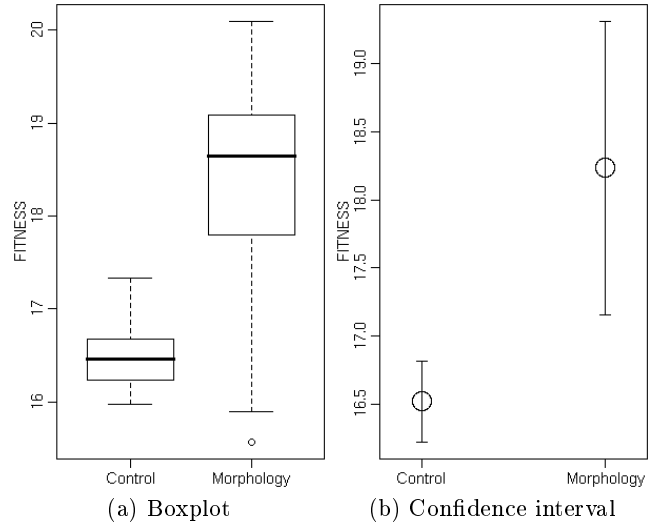


Figure 7: Importance of morphology evolution

According to Figure 7, the results obtained by the morphology and control evolution are clearly superior to those obtained using just the control evolution, since the confidence intervals are not superposed. Figure 8 shows a walking accomplished by an evolved robot. Figure 9 shows the morphologies evolved in Table 3 experiments. Observing Figure 9, it is noticed that the large state space allows the evolution of different efficient solutions, in a similar manner that was occurred in the natural evolution.

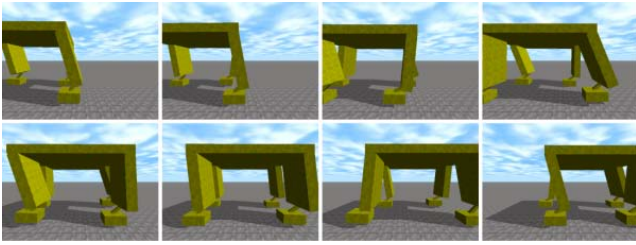


Figure 8: Robot evolved in the 6th experiment

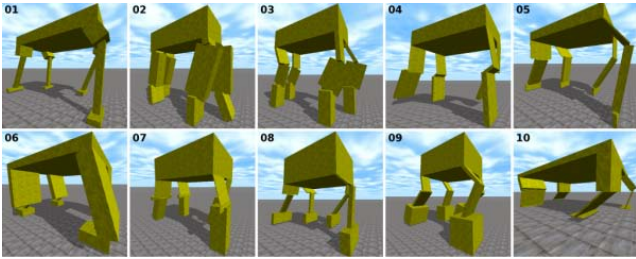


Figure 9: Robot morphologies evolved

## 5. CONCLUSIONS AND PERSPECTIVES

The main goal of this paper was to describe our research and experiments with autonomous robots, in which were evolved stable gaits of simulated legged robots in a physically based simulation environment. These experiments were accomplished using the LegGen simulator, which was developed in order to study the automatic configuration of parameters used to control the gait of legged robots. In the LegGen simulator, the robot joints are controlled using two different strategies: (i) GA evolved a finite state machine and (ii) GA evolved an artificial neural network. Several experiments were achieved, comparing both approaches and demonstrating (with a valid statistical analysis) that the neural controller is superior to the FSM controller, obtaining a better performance (more stable, better displacement). Besides, the experiments demonstrate that morphology evolution is better than evolution of control parameters only.

## Acknowledgments

This work was supported by Fapergs, CAPES and CNPq.

## 6. REFERENCES

- [1] G. A. Bekey. *Autonomous Robots: From Biological Inspiration to Implementation and Control*. MIT Press, Cambridge, MA, 2005.
- [2] J. C. Bongard and R. Pfeifer. A method for isolating morphological effects on evolved behaviour. In *Proc. 7th Int. Conf. Simulation of Adaptive Behav. (SAB)*, pages 305–311, Edinburgh, UK, Aug. 2002. MIT Press.
- [3] J. Busch, J. Ziegler, C. Aue, A. Ross, D. Sawitzki, and W. Banzhaf. Automatic generation of control programs for walking robots using genetic programming. In *Proc. 5th European Conf. Genetic Programming (EuroGP)*, volume 2278 of *LNCS*, pages 258–267, Kinsale, Ireland, Apr. 2002. Springer-Verlag.
- [4] G. Dudek and M. Jenkin. *Computational Principles of Mobile Robotics*. Cambridge Univ. Press, UK, 2000.
- [5] P. Eggenberger. Evolving morphologies of simulated 3d organisms based on differential gene expression. In *Proc. 4th European Conf. Artificial Life (ECAL'97)*, pages 205–213, Cambridge, MA, 1997. MIT Press.
- [6] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.
- [7] D. Golubovic and H. Hu. Ga-based gait generation of sony quadruped robots. In *Proc. 3th IASTED Int. Conf. Artificial Intelligence and Applications (AIA)*, Benalmadena, Spain, Sept. 2003.
- [8] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, Upper Saddle River, 1999.
- [9] M. R. Heinen and F. Osório. Gait control generation for physically based simulated robots using genetic algorithms. In *Proc. 18th Brazilian Symposium on AI (SBIA)*, LNCS, Ribeirão Preto - SP, Brazil, Oct. 2006.
- [10] M. R. Heinen and F. S. Osório. Applying genetic algorithms to control gait of physically based simulated robots. In *Proc. IEEE Congr. Evolutionary Computation (CEC)*, Vancouver, Canada, July 2006.
- [11] M. R. Heinen and F. S. Osório. Neural networks applied to gait control of physically based simulated robots. In *Proc. 9th Brazilian Neural Networks Symposium (SBRN)*, Ribeirão Preto, SP, Brazil, 2006.
- [12] M. R. Heinen and F. S. Osório. Applying genetic algorithms to control gait of simulated robots. In *Proc. IEEE Electronics, Robotics and Automotive Mechanics Conf. (CERMA) 2007*, pages 500–505, Cuernavaca, Morelos, Mexico, Sept. 2007.
- [13] M. R. Heinen and F. S. Osório. Evolving gait control of physically based simulated robots. *Revista de Informática Teórica e Aplicada (RITA)*, XVI(1):119–134, 2007.
- [14] D. Jacob, D. Polani, and C. L. Nehaniv. Legs than can walk: Embodiment-based modular reinforcement learning applied. In *Proc. IEEE Int. Symposium on Computational Intelligence in Robotics and Autom. (CIRA)*, pages 365–372, Espoo, Finland, June 2005.
- [15] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 2619–2624, New Orleans, LA, Apr. 2004.
- [16] M. A. Lewis, A. H. Fagg, and A. Solidum. Genetic programming approach to the construction of a neural network for control of a walking robot. In *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 2618–2623, Nice, France, 1992.
- [17] R. B. McGhee. Robot locomotion. *Neural Control of Locomotion*, pages 237–264, 1976.
- [18] T. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.
- [19] R. Pfeifer and C. Scheier. *Understanding Intelligence*. MIT Press, Cambridge, MA, 1999.
- [20] M. H. Raibert. *Legged Robots That Balance*. MIT Press, Cambridge, MA, 1986.
- [21] R. Reeve and J. Hallam. An analysis of neural models for walking control. *IEEE Trans. Neural Networks*, 16(3):733–742, May 2005.
- [22] K. Sims. Evolving virtual creatures. *Computer Graphics*, 28:15–24, July 1994.