2nd International Conference on Engineering Optimization
September 6-9, 2010, Lisbon, Portugal

A Global Optimization Stochastic Algorithm for Head Motion Stabilization during Quadruped Robot Locomotion

Lino Costa[1], **Ana Maria A.C. Rocha**[2], **Cristina P. Santos**[3] & **Miguel Oliveira**[4]

Department of Production and Systems, School of Engineering, University of Minho, 4710-057 Braga, Portugal

[1]lac@dps.uminho.pt; [2]arocha@dps.uminho.pt

Industrial Electronics Department, School of Engineering, University of Minho, 4800-058 Guimarães, Portugal

[3]cristina@dei.uminho.pt, [4]mcampos@dei.uminho.pt

Abstract
Visually-guided locomotion is important for autonomous robotics. However, there are several difficulties, for instance, the robot locomotion induces head shaking that constraints stable image acquisition and the possibility to rely on that information to act accordingly. In this work, we propose a combined approach based on a controller architecture that is able to generate locomotion for a quadruped robot and a genetic algorithm to generate head movement stabilization. The movement controllers are biologically inspired in the concept of Central Pattern Generators (CPGs) that are modelled based on nonlinear dynamical systems, coupled Hopf oscillators. This approach allows to explicitly specify parameters such as amplitude, offset and frequency of movement and to smoothly modulate the generated oscillations according to changes in these parameters. Thus, in order to achieve the desired head movement, opposed to the one induced by locomotion, it is necessary to appropriately tune the CPG parameters. Since this is a non-linear and non-convex optimization problem, the tuning of CPG parameters is achieved by using a global optimization method. The genetic algorithm searches for the best set of parameters that generates the head movement in order to reduce the head shaking caused by locomotion. Optimization is done offline according to the head movement induced by the locomotion when no stabilization procedure was performed. In order to evaluate the resulting head movement, a fitness function based on the Euclidian norm is investigated. Moreover, a constraint handling technique based on tournament selection was implemented. Experimental results on a simulated AIBO robot demonstrate that the proposed approach generates head movement that reduces significantly the one induced by locomotion.
Keywords: Stochastic Algorithms, Global Optimization, Robotics, Quadruped Locomotion, Central Pattern Generators, Dynamical systems.

1. Introduction

Robot locomotion is a challenging task that involves several relevant subtasks, not yet completely solved. The head shaking that results from robot locomotion is important because it difficults stable image acquisition and the possibility to rely on that information to act accordingly, for instance, to achieve visually-guided locomotion. This work deals with the motion stabilization system of an ers-7 AIBO quadruped robot, which performs its own head motion according to a feedforward controller. Several similar works have been proposed in literature [7, 12, 15, 17]. However, these methods consider the robot moves according to a scheduled robot motion plan, which imply that space and time constraints on robot motion must be known before hand as well as robot and environment models. As such, control is based on this scheduled plan.

Our focus meets on the development of a head controller able to minimize the head motion induced by locomotion itself. Specifically, we propose a combined approach to generate head movement stabilization on a quadruped robot, using Central Pattern Generators (CPGs) and a genetic algorithm. CPGs are neural networks located in the spine of vertebrates, able to generate coordinated rhythmic movements, namely locomotion. The CPGs generate trajectories for tilt, pan and nod head joints. These CPGs are modelled as coupled oscillators and solved using numeric differentiation. The proposed CPGs, based on Hopf oscillators, allow to explicitly specify parameters such as amplitude, offset and frequency of movement and to smoothly modulate the generated trajectories according to changes in these parameters. Thus, in order to achieve the desired head movement, opposed to the one induced by locomotion, it is

necessary to appropriately tune the CPG parameters. This is achieved using an optimization method by optimizing the CPG parameters and smoothly modulate the generated trajectories according to changes in these parameters. Here, optimization is done off-line according to the head movement induced by the locomotion when no stabilization procedure was performed.

In this paper, a genetic algorithm will be used to determine the best set of parameters that generate the head movement in order to reduce the head shaking caused by locomotion. Note that we are dealing with a nonlinear problem where continuity and convexity conditions are not guarantee. Hence, searching for a global optimum is a difficult task that requires approaches based on stochastic algorithms like evolutionary algorithms, in particular, genetic algorithms. These are search algorithms that mimic the process of natural selection [5]. Thus, unlike conventional algorithms, in general, only the information regarding the objective to optimize is required. Moreover, they are based on a population that evolves over time, possibly in the direction of the optimum. Moreover, several constraints are imposed in this optimization problem. We have already addressed this problem in a preliminary experience using the genetic algorithms [10] and the electromagnetism-like algorithm [11]. Moreover, we noticed taht solving this problem requires a considerable computational effort. Thus, alternative techniques for handling constraints can become the search more efficient. In this work, several experiments were conducted in order to compare the performance of distinct constraint handling techniques. Experimental results on a simulated AIBO robot demonstrate that the proposed approaches generate head movement that does not eliminate but reduce the one induced by locomotion.

The remainder of this paper is organized as follows. Section 2 describes the head stabilization problem. In Section 3, the stochastic algorithm is presented as well as the constraint-handling techniques. Next, the results of the experiments conducted are presented and discussed in Section 4. Finally, the main conclusions and future work are addressed in Section 5.

2. Head Stabilization Problem

In order to implement the head motion required to reduce the camera (head) movement induced by locomotion itself, it is necessary one or several optimal combinations of amplitude, offset and frequency of each head oscillator. This is possible because we can easily modulate these parameters of the generated trajectories according to changes in the CPG parameters and these are represented in an explicit way by our CPGs. Therefore, we have to tune the CPG parameters: amplitude μ, offset O and frequency ω. The multitude of parameter combinations is large, and it is difficult to derive an accurate model for the tested quadruped robot and for the environment. Besides, such a model based approach would also require some post-adaptation of results (because of backlash, friction, etc).

In this study, the search of parameters suitable for the implementation of the required head motion is carried out based on the data from a simulated quadruped robot. We recorded the (X, Y, Z) head coordinates in a world coordinate system (see Fig. 1), when the robot walks during 30 s (seconds) and no head stabilization is performed. We are interested in the opposite of this movement around the (X, Y, Z) coordinates. From now on, this data is referred to as $(X, Y, Z)_{\text{observed}}$.

Figure 1: World coordinate system.

We implemented head stabilization for 30 s (seconds). It is arbitrary and could have been chosen differently. Three head CPGs generate rhythmic motion for the tilt, pan and nod joints. By applying forward kinematics [16], we compute the resulting $(X, Y, Z)_{\text{computed}}$ head coordinates in the world coordinate system. The distance between the observed and computed head coordinates is the evaluated criterion used to explore the parameter space of the CPG model to identify the head movement that minimizes the one induced by locomotion itself. Thus, the constrained optimization problem can be

mathematically formulated has follows:

$$
\begin{aligned}
\min f(x) = \quad & \|(X,Y,Z)_{\text{observed}} - (X,Y,Z)_{\text{computed}}\|_2 \\
\text{subject to} \quad & -y_{\text{tilt}} - 75 + \tfrac{A_{\text{tilt}}}{2} \leq 0 \\
& y_{\text{tilt}} + \tfrac{A_{\text{tilt}}}{2} \leq 0 \\
& -y_{\text{pan}} - 88 + \tfrac{A_{\text{pan}}}{2} \leq 0 \\
& y_{\text{pan}} - 88 + \tfrac{A_{\text{pan}}}{2} \leq 0 \\
& -y_{\text{nod}} - 88 + \tfrac{A_{\text{nod}}}{2} \leq 0 \\
& y_{\text{nod}} - 45 + \tfrac{A_{\text{nod}}}{2} \leq 0 \\
& 0 \leq A_{\text{tilt}} \leq 75 \\
& -75 \leq y_{\text{tilt}} \leq 0 \\
& 1 \leq w_{\text{tilt}} \leq 12 \\
& 0 \leq A_{\text{pan}} \leq 176 \\
& -88 \leq y_{\text{pan}} \leq 88 \\
& 1 \leq w_{\text{pan}} \leq 12 \\
& 0 \leq A_{\text{nod}} \leq 60 \\
& -88 \leq y_{\text{nod}} \leq 45 \\
& 1 \leq w_{\text{nod}} \leq 12
\end{aligned}
\tag{1}
$$

3. Stochastic Algorithm Framework

The stochastic algorithm framework is based on a genetic algorithm that searches the optimal combinations of the CPG parameters. The basic idea is to combine the CPG model for head movement generation with the optimization algorithm. The distance between the observed and computed head coordinates is used as a measure of the quality of the resulting head movement. Two constraint handling techniques based on a repair mechanism or a tournament selection mechanism will be embedded in the genetic algorithm to solve the constrained optimization problem previously defined in Eq. (1).

3.1. Genetic Algorithm

A Genetic Algorithm (GA) is a population based algorithm that uses techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover [5]. Thus, unlike conventional algorithms, GAs start from a population of points P of size s. In spite of the traditional binary representation used by GAs, in our implementation, a real representation is used since we are leading with a continuous problem. Therefore, each point of the population $z^{(l)}$, for $l = 1, \ldots s$, is an n dimensional vector. The initial population is randomly generated. A fitness function is defined in order to compare the points of the population (in terms of objective function value and constraint violation). We implement a stochastic selection that guarantees that better points are more likely to be selected. New points in the search space are generated by the application of genetic operators (crossover and mutation) to the selected points from population. Elitism is implemented by maintaining, during the search, a given number e, of best points in the population.

Crossover combines two points in order to generate new ones. A Simulated Binary Crossover (SBX) [3] that simulates the working principle of single-point crossover operator for binary strings is implemented. Two points, $z^{(1)}$ and $z^{(2)}$, are randomly selected from the pool and, with probability p_c, two new points, $w^{(1)}$ and $w^{(2)}$ are generated according to

$$
\begin{aligned}
w_i^{(1)} &= 0.5\left((1+\beta_i)z_i^{(1)} + (1-\beta_i)z_i^{(2)}\right) \\
w_i^{(2)} &= 0.5\left((1-\beta_i)z_i^{(1)} + (1+\beta_i)z_i^{(2)}\right),
\end{aligned}
\qquad
\beta_i = \begin{cases} (2r_i)^{\frac{1}{\eta_c+1}} & \text{if} \quad r_i \leq 0.5 \\ \left(\frac{1}{2(1-r_i)}\right)^{\frac{1}{\eta_c+1}} & \text{if} \quad r_i > 0.5 \end{cases}
\tag{2}
$$

for $i = 1, \ldots, n$, where $r_i \sim U(0,1)$, (r_i is a random number uniformly distributed in $[0,1]$) and $\eta_c > 0$ is an external parameter of the distribution. This procedure is repeated until the number of generated points equals the number of points in the pool.

A Polynomial Mutation is applied, with a probability p_m, to the points produced by the crossover operator. Mutation introduces diversity in the population since crossover, exclusively, could not assure the exploration of new regions of the search space. This operator guarantees that the probability of creating a new point $t^{(l)}$ closer to the previous one $w^{(l)}$ $(l = 1, \ldots, s)$ is more than the probability of creating one away from it. It can be expressed by:

$$t_i^{(l)} = w_i^{(l)} + (u_i - l_i)\iota_i\,, \qquad \iota_i = \begin{cases} (2r_i)^{\frac{1}{\eta_m+1}} - 1 & \text{if} \quad r_i < 0.5 \\ 1 - (2(1-r_i))^{\frac{1}{\eta_m+1}} & \text{if} \quad r_i \geq 0.5 \end{cases} \tag{3}$$

for $i = 1, \ldots, n$, where $r_i \sim U(0,1)$ and $\eta_m > 0$ is an external parameter of the distribution. The GA proceeds as the following algorithm.

Algorithm 1 *Genetic Algorithm*

1. *Given* $e > 1$, $s > 1$, $0 < p_c < 1$, $\eta_c > 0$, $0 < p_m < 1$, $k_{\max} > 1$, *set* $k = 0$;
2. *Randomly generate* $z^{(l),k} \in \Omega$, *for* $l = 1, \ldots, s$ (Initialization of P);
3. *While the stopping criterion is not met do*
   - 3.1. *Compute* $F(z^{(l),k})$, *for* $l = 1, \ldots, s$
   - 3.2. *Select by tournaments* $s - e$ *points from* P
   - 3.3. *Apply SBX crossover to the* $s - e$ *points, with probability* p_c *using 2*
   - 3.4. *Apply mutation to the* $s - e$ *points with probability* p_m *using 3*
   - 3.5. *Replace the worst* $s - e$ *points of* P
   - 3.6. *Set* $k = k + 1$;
4. *Set* $y = z_{best}^k$;

This procedure stops when $k_{\max}$ generations are performed. Then, the procedure is terminated and the best point in the population is returned. This returned point is the best approximation to the solution of the problem defined by Eq. (1), i.e., the point that minimizes the distance between the observed and computed head coordinates of the robot.

3.2. Constraint handling

As we can see in Eq. 1, the problem has several simple boundary constraints as well as inequality constraints. In order to the first type of constraints, each new generated point is projected component by component in order to satisfy boundary constraints (for all $i, \ldots, n$) as follows:

$$x_i = \begin{cases} l_i & \text{if} \quad x_i < l_i \\ x_i & \text{if} \quad l_i \leq x_i \leq u_i \\ u_i & \text{if} \quad x_i > u_i \end{cases} . \tag{4}$$

In order to handle the inequality constraints, two approaches are implemented: a repairing method and the tournament based constraint method [2]. In the approach based on a repairing mechanism, any infeasible solution is repaired exploring the relations among variables expressed by the inequality constraints, i.e., the values y_{tilt}, y_{pan} and y_{nod} are repaired in order to satisfy the constraints. The application of this repairing mechanism to all infeasible solutions in the population, guarantees that all solutions become feasible. Thus, the fitness function can be defined as $F(x) = f(x)$. We want to remark that the weakness of the repairing methods lies in their problem dependence. Different repair procedures have to be designed for each particular problem. There are no standard heuristics for accomplishing this. Sometimes repairing infeasible solutions might become as a complex task as solving the original problem.

The second approach is based on the tournament based constraint method proposed by Deb [2] that is based on a penalty function which does not require any penalty parameter. For comparison purposes, tournament selection is exploited to make sure that:

1. when two feasible solutions are compared, the one with better objective function value is chosen;
2. when one feasible and one infeasible solutions are compared, the feasible solution is chosen;

3. when two infeasible solutions are compared the one with smaller constraint violation is chosen.

Therefore, the fitness function, in which infeasible solutions are compared based on only their constraint violation, can be expressed by:

$$F(x) = \begin{cases} f(x) & \text{if} \quad g_j(x) \leq 0, \forall j \\ f_{\max} + \sum_{j=1}^{m} |\max(0, g_j(x))| & \text{otherwise} \end{cases} \tag{5}$$

where $f_{\max}$ is the objective function value of the worst feasible solution in the population and $g_j(x)$ are the m inequality constraints to be satisfied ($g_j(x) \leq 0$). This approach can be only applicable to population-based search methods such as GAs. Thus, the fitness of an infeasible solution not only depends on the amount of constraint violation, but also on the population of solutions at hand. An important advantage of this approach is that it is not required to compute the objective function value for infeasible solutions.

4. Numerical Experiments

In this section the results of our numerical experiences are reported. Here we aim to compare the performance of the two constraint handling techniques: the repairing mechanism and the tournament based constraint method, embedded in the genetic algorithm to solve Eq. (1). The algorithm is coded in Matlab 7, and the results were obtained in a Intel Pentium 4 computer running Windows XP SP3, CPU 3.3Ghz with 1 GB of RAM.

Table 1 lists the genetic algorithm parameters used in the optimization process.

Table 1: Genetic Algorithm parameters.

$k_{\max}$	s	e	p_c	η_c	p_m	η_m
300	100	10	0.9	20	1/9	20

When stochastic methods are used to solve problems, the impact of the random number seeds has to be taken into consideration and each optimization process should be run a certain number of times. In this experience we set it to 10. In Table 2, we report the computational times (from column 2-4) and best solutions found (last three columns) in our numerical study. Here, the first column refers to the constraint-handling technique used, followed by the best time, average time and the standard deviation time spent (in hours), over the 10 runs. Next three columns list the best, average and standard deviation of the solutions found (in terms of fitness function) after all the 10 runs, respectively. Note that, in all experiments, the evaluation time for head movement generation is 30 s.

Table 2: Computational times and best solution found.

Constraint-handling Technique	Best time (hours)	Average time (hours)	Std. Dev. time (hours)	Best fitness	Average fitness	Std. Dev. fitness
Repairing	9.2078	9.9606	0.7166	3978.6	5232.3	762.6713
Tournament	4.9259	5.4823	0.2901	3983.8	4877.0	839.3602

Comparing the performance of the two mechanisms, by analyzing the Table 2, we conclude the two mechanisms to handle constraints are competitive. In terms of best fitness found we see that the approach using repairing solutions is the one who has least value. However the average fitness value found over the 10 runs is better. We remark that the metric in terms of average values is indeed the most important when comparing stochastic algorithms, since it reports the central tendency of the results over the runs. Clearly, the tournament selection method is the least time consuming algorithm. This is because, this algorithm deals not only with feasible but with infeasible points, and in this latter case the forward kinematics is not activated. We also remark that the tournament method had 27970 as average number of function evaluations while the repairing method involved 30000 function evaluations in each run. This difference is justified for not being necessary to compute the objective function value for infeasible solutions.

Table 3 shows tuned CPG parameters representing the best point obtained, over the 10 runs, using each constraint handling mechanism.

Table 3: Best point CPG parameters

	Repairing	**Tournament**
A_{tilt}	1.8286	1.9104
y_{tilt}	-0.914	-0.955
w_{tilt}	4.1124	4.1099
A_{pan}	8.0535	8.0709
y_{pan}	0.0930	0.1150
w_{pan}	2.1307	2.1304
A_{nod}	0.4825	0.0816
y_{nod}	-1.1543	-1.0832
w_{nod}	3.9974	7.5382

For the repairing mechanism, Fig. 2 shows the evolution of the average fitness along the 300 generations for 10 runs (the best run is in bold).

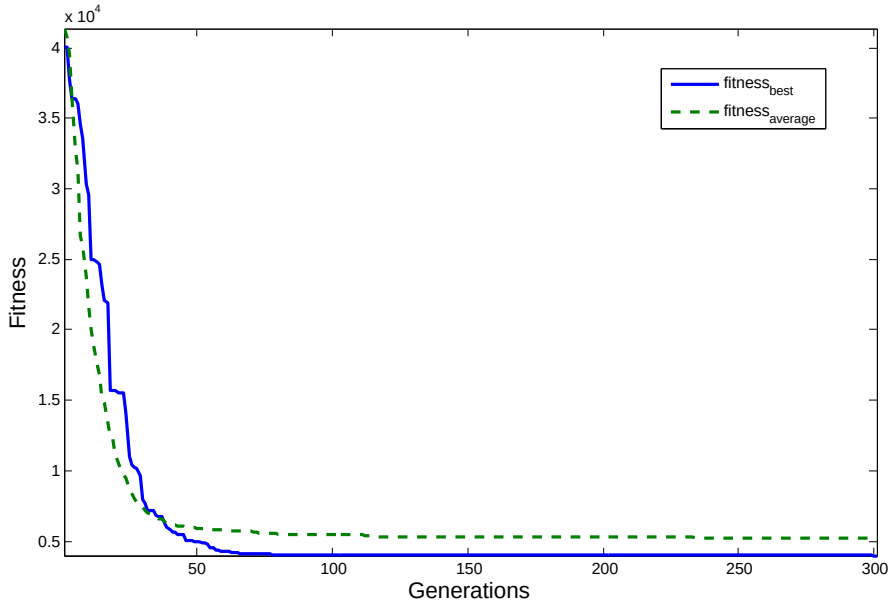

Figure 2: Fitness evolution, with repairing mechanism, along 300 generations.

For the tournament mechanism, Fig. 3 shows the evolution of the average fitness along the 300 generations for 10 runs (the best run is in bold).

Fig. 4 depicts the time courses of the (X, Y, Z) calculated (solid line) and observed (dotted line) head movement according to the CPG parameters of the best solution of 300th generation, when employing the tournament mechanism. This data was mathematical treated such as to keep only the oscillations in the movement and remove the drift that the robot has in the X coordinate and also the forward movement in the Z coordinate. We conclude that the generated movements are quite similar in the X coordinate. The calculated movement is quite different in the Z coordinate. This results from the fact that the pan joint controls movement in the X coordinate, while both the tilt and nod joints control the Y and Z coordinates. If we are able to mimic movement in the Y coordinate, in the Z coordinate this mimic is therefore difficult to achieve.

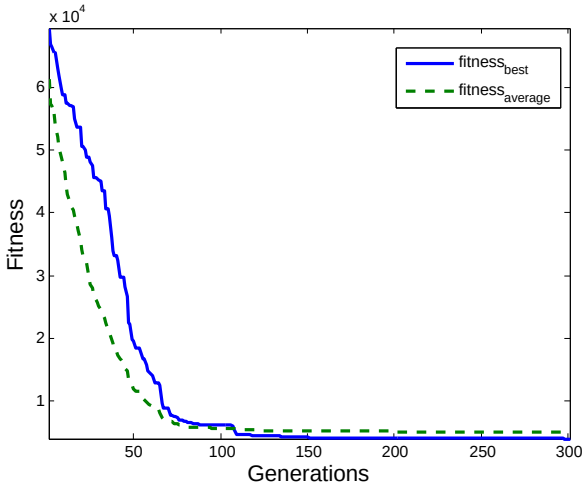

Figure 3: Fitness evolution, with tournament mechanism, along 300 generations.

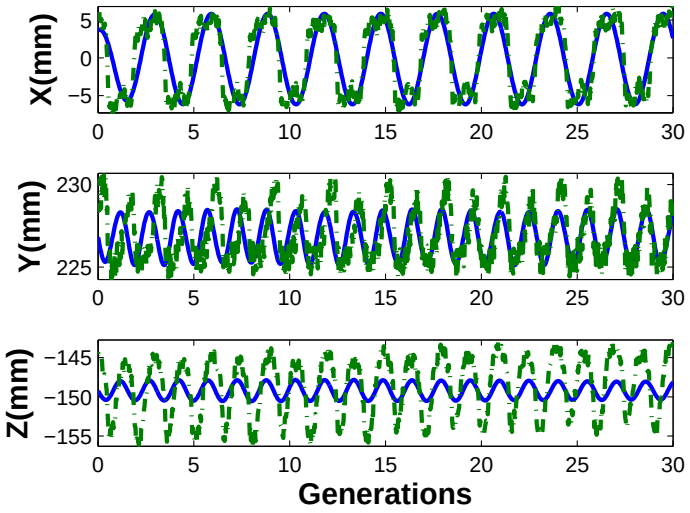

Figure 4: (X, Y, Z) calculated (solid line) and observed (dotted line) head movement according to the CPG parameters of the 300th generation best point.

4. Conclusions

In this article, we focus on the development of a head controller able to minimize the head motion induced by locomotion itself, on a quadruped robot. Our aim was to build a system able to eliminate or reduce the head motion of a robot that walks in the environment. For that, we set a dynamical controller generating trajectories for the head joints such that the final head movement is opposite to the one induced by locomotion. Specifically, we propose a combined approach to generate head movement stabilization, using CPG model for head movement generation and a genetic algorithm that searches the optimal combinations of the CPG parameters. Two constraint handling techniques based on a repair mechanism or a tournament selection mechanism were embedded in the genetic algorithm to solve the constrained optimization problem. Experimental results on a simulated AIBO robot demonstrate that the proposed approaches generate head movement that does not eliminate but reduce the one induced by locomotion. Furthermore, the comparing results show that both techniques are competitive in terms of

fitness value found, despite the computational cost for the repairing method. Nevertheless, the repairing methods are problem dependent. In this study, the search of parameters suitable for the implementation of the required head motion was carried out based on the data from a simulated quadruped robot. In the future we intend to implement these ideas in the real quadruped robot.

References

[1] L. Castro, C.P. Santos, M. Oliveira and A. Ijspeert, Postural Control on a Quadruped Robot Using Lateral Titl: a Dynamical System Approach, *European Robotics Symposium EUROS 2008*, Prague, 2008.

[2] K. Deb, An Efficient Constraint Handling Method for Genetic Algorithms, *Computer Methods in Applied Mechanics and Engineering*, 311–338, 1998.

[3] K. Deb, Agrawal, R.B.: Simulated binary crossover for continuous search space, *Complex Systems*, 9 (2), 115–149, 1995.

[4] S. Degallier, C. P. Santos, L Righetti and A. Ijspeert, Movement Generation using Dynamical Systems: a Drumming Humanoid Robot, *Humanoids06 IEEE-RAS International Conference on Humanoids Robots*, Genova, Italy, December 4–6, 2006.

[5] D. Goldberg, Genetic Algorithms in Search, *Optimization*, and Machine Learning, Addison-Wesley, (1989).

[6] S. Grillner, Neurobiological bases of rhythmic motor acts in vertebrates, *Science*, 228 (4696), 143–149, 1985.

[7] R. Kaushik, M. Marcinkiewicz, J. Xiao, S. Parsons, and T. Raphan, Implementation of Bio-Inspired Vestibulo-Ocular Reflex in a Quadrupedal Robot,*2007 IEEE International Conference on Robotics and Automation (ICRA 2007)*, Roma, Italy, 10-14 April, 4861–4866, 2007.

[8] O. Michel, Webots: Professional mobile robot simulation, *International Journal of Advanced Robotic Systems*, 1 (1), 39–42, 2004.

[9] J. Pratt, C. Chew, A. Torres, P. Dilworth and G. Pratt, Virtual Model Control: An intuitive approach for bipedal locomotion, *The Int. J. of Robotics Research*, 20 (2), 129–143, 2001.

[10] C. Santos, M. Oliveira, A.M.A.C. Rocha and L. Costa, Head Motion Stabilization During Quadruped Robot Locomotion: Combining Dynamical Systems and a Genetic Algorithm, *IEEE International Conference on Robotics and Automation (ICRA 2009)*, May 12-17, Kobe, Japan 2009.

[11] C.P. Santos, M. Oliveira, V. Matos, A.M. A.C. Rocha and L. Costa, Combining Central Pattern Generators with the Electromagnetism-like Algorithm for Head Motion Stabilization during Quadruped Robot Locomotion, *2nd International Workshop on Evolutionary and Reinforcement Learning for Autonomous Robot Systems (ERLARS 2009)*, St. Louis, Missouri, USA, October 15, 2009.

[12] Y. Son, T. Kamano, T. Yasuno, T. Suzuki and H. Harada, Generation of Adaptive Gait Patterns for Quadruped Robot with CPG Network Including Motor Dynamic Model, *Electrical Engineering in Japan*, 155 (1), 2006.

[13] A. Sproewitz, R. Moeckel, J. Maye, M. Asadpour and A.J. Ijspeert, Adaptive locomotion control in modular robotics. *Workshop on Self-Reconfigurable Robots/Systems and Applications IROS07*, 81–84, November 2007.

[14] L. Righetti and A.J. Ijspeert, Pattern generators with sensory feedback for the control of quadruped locomotion, *IEEE International Conference on Robotics and Automation ICRA 2008*, California, 2008.

[15] S. Takizawa, S. Ushida, T. Okatani, K. Deguchi, 2DOF Motion Stabilization of Biped Robot by Gaze Control Strategy. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, 3809–3814, August 2005.

[16] E. Tira-Thompson, Tekkotsu, A rapid development framework for robotics, Masters thesis, *Masters Thesis*, Carnegie Mellon University, 2004.

[17] H. Yamada, M. Mori, S. Hirose, Stabilization of the head of an undulating snake-like robot, *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, San Diego, CA, USA, Oct 29 - Nov 2, 3566–3571, 2007.