Evolution of Biped Walking Using Neural Oscillators Controller and Harmony Search Algorithm Optimizer

Ebrahim Yazdi, Abolfazl Torogh Haghighat

Abstract—In this paper, a simple Neural controller has been used to achieve stable walking in a NAO biped robot, with 22 degrees of freedom that implemented in a virtual physics-based simulation environment of Robocup soccer simulation environment. The algorithm uses a Matsuoka base neural oscillator to generate control signal for the biped robot. To find the best angular trajectory and optimize network parameters, a new population-based search algorithm, called the Harmony Search (HS) algorithm, has been used. The algorithm conceptualized a group of musicians together trying to search for better state of harmony. Simulation results demonstrate that the modification of the step period and the walking motion due to the sensory feedback signals improves the stability of the walking motion.

Index Terms—Bipedal Walking, Harmony Search Algorithm, Neural Oscillator, Sensory Feedback Signal.

1 INTRODUCTION

The infrastructure of our society is designed for humans. For example, the sizes of doors and the heights of steps on stairs are determined by considering the heights of people and the lengths of their legs. Therefore, we can apply robots for our society without extra investment in the infrastructure if the robots have the human shape [1]. In researches about bipedal robots, walking is one of the main challenges. There are two major approaches, model-based and model-free, in bipedal walking researches. In model-based approaches, controller of robot is dependent on model of robot and from one robot to another every thing in controller should be changed. Two well known methods in this approach are "Zero Moment Point" [2, 3] (ZMP) and "Inverted Pendulum" [4]. The animated end results of these efforts closely resemble natural motion patterns, but may nevertheless fail to convince the human eye in specifically designed "motion Turing tests" [5]. This lack of realism is a direct consequence of the controller architecture employed; a state machine does not readily produce the fluctuations typical of real locomotion. More significantly, it cannot easily be extended to integrate sensory input. Furthermore, the creation of state machines can be a cumbersome process, as states have to be identified, implemented and fine-tuned by hand for each type of gait to be modeled [5]. In model-free approach, controller of robot is independent of its model. Model-free approach has two portions. A portion for control of the robot and a portion for find the best values for variables of controller. Genetic Algorithm Optimized Fourier Series Formulation (GAOFSF) [6] as a model free approach is imitated of Human’s walk.

GAOFSF was used in 2006 for the first time for gait generation in bipedal locomotion [7]. GAOFSF uses TFS (Truncated Fourier Series; modified definite Fourier serie equation) to control the robot and uses from genetic algorithms for optimizing the parameters of TFS. Central Pattern Generator (CPG) as a model-Free approach uses from a set of neural oscillators for controlling and uses genetic algorithm as a weight optimizer. Many studies have been carried out in order to elucidate the role for CPG in locomotion using quadruped robots [8], [9], biped robots [10], [11], [12], and simulated salamander [13].

In this paper, we investigate experimentally sensory feedback neural network controller for biped locomotion based on Matsuoka [14] formulate. A controller composed of nonlinear oscillators receives feedback signals from touch sensors. Instantly, feedback sensors send signals to this network, its neural oscillators generate output signals for applying to joint angle trajectories of robot.

In similar works[15,16] GA used as optimizer for finding best weights for neural networks, but in this approach, the Harmony Search (HS) algorithm [17] technique with constraint handling on angles and time is used to find optimum weights of network and train the robot to achieve fast bipedal walking, for the first time. HS is a meta-heuristic algorithm, mimicking the improvisation process of music players [18]. During last several years, the HS algorithm has been vigorously applied to various optimization problems [18, 19], presenting several advantages with respect to traditional optimization techniques such as the following [18]: (a) HS algorithm is simple in concept, few in parameters, easy in implementation, imposes fewer mathematical requirements, and does not require initial value settings of the decision variables. (b)
As the HS algorithm uses stochastic random searches, derivative information is also unnecessary. (c) The HS algorithm generates a new vector, after considering all of the existing vectors, whereas the GA only considers the two parent vectors. These features increase the flexibility of the HS algorithm and produce better solutions.

The paper is organized as follows: Section 2 provides a definition about the simulator, Nao robot and explanation about important joints for walking. In section 3, we provide a formal definition of the Matsuoka formulate and using this to our suggested Neural Network. Section 3, begins with explanations about HS and continues with applying HS as an optimizer to our algorithm. Section 4 gives the performance evaluation of the proposed algorithm compared to GA-Based algorithm. Section 5 concludes the paper.

2 SIMULATOR AND ROBOT MODEL

The target Robot of our study is a 22-DOF (degrees of freedom) NAO robot (Fig.1) with 4.5Kg weight and 57Cm stand height. The robot has two arms with four DOF for each arm, two legs with six DOF for each leg and a head with 2 degrees of freedom.

The simulation performed by Rcssserver3d simulator which is a generic three dimensional simulator based on Spark and Open Dynamics Engine (ODE [20]). Spark is capable of carrying out scientific distributed multi agent calculations as well as various physical simulations ranging from articulated bodies to complex robot environments [21]. The time‐integrated simulation is processed with a resolution of 50 simulation steps per second. Rcssserver3d simulator is a noisy simulator and to overcome inherent noise of the simulator, Resampling algorithm is implied which could lead to robustness in nondeterministic environments.

In this approach, the body trunk of the robot is not actuated. Our experiences show, the leg joints are more effective joints for leg motions, that the joints of hip, knee and ankle which move on the same plane of forward-backward are the major ones (Table 1 shows description of them). Although, other joints are also effective, but in fact, their role smoothes the Robot walking motion. Because of this we prefer to ignore them to decrease learning search space.

3 NEURAL SYSTEM MODEL

3.1 Matsuoka Neural oscillator

Matsuoka oscillator is a kind of relaxation oscillator and is basically composed of a pair of mutually inhibiting neural elements with adaptation effects (i.e. exhaustion after excitation) [14].

The original Matsuoka neural oscillator formulate can be written as follow:
\[ r_1 x_1 = c - x_1 - \beta v_1 - \gamma v_2 - x_2 + \alpha u f_1 \] (1)
\[ r_2 v_1 = [x_1]^+ - v_1 \] (2)
\[ r_1 x_2 = c - x_2 - \beta v_2 - \gamma [x_1]^+ + \alpha u f_2 \] (3)
\[ r_2 v_2 = [x_2]^+ - v_2 \] (4)

Where \( x_1, x_2, v_1, v_2 \) are internal variables; \( r_1, r_2 \) are time constants; \( \beta \) represents the intensity (suppression) of adaptation, \( c \) means the constant input; \( u_1, u_2 \) are the feedback inputs mainly from sensors, the inputs scaled by gain \( \alpha \); \( \gamma \) is the connection coefficient of two exhaustive elements. Given a certain constant input \( c \) to both of the two elements, either of the elements with higher state values initially gets exited more than the other and the more excited one inhibits the other. As a result, the excited element sends the positive feedback to the other element and because of receiving inhabitation from the other element becomes weaker. Then after the excitation of one element reaches the peak, it gradually relaxes due to the adaptivity. This weakens the suppression to the other element. As a result the excitation of the other element grows stronger. So, we have a symmetrical opposite signals to the initial state and this process continues like seesaw game. If \( u_1, u_2 \) are large enough, and close to the oscillator’s natural frequency, the phase difference between input and output is tightly locked due to entrainment. Figure 2 shows this neural oscillator architecture.

### 3.2 THE NEURAL ARCHITECTURE

Human motions are recognized flexible and periodic but more challenging for motion stability issue. Therefore, human-like motion patterns are included into the research objectives. The walking trajectory is divided into several types. Positional trajectory and angular trajectory are two of them. The main idea for the walking posture control used here is based on the reaction angular trajectory. Similar to [22], Foot was kept parallel to the ground by using ankle joint in order to avoid collision. Therefore, ankle trajectory can be calculated by hip and knee trajectories and ankle DOF parameters are eliminated.

According to this fact that human walking is a periodic motion and attention to this point that Matsuoka oscillator can generate periodic signals, our neural oscillator approach generates a core oscillation with the use of the Matsuoka oscillator. One of the elements in the Matsuoka oscillator corresponds to left leg and the other to right leg. The output of the Matsuoka oscillator given to same leg neurons. Besides, We have taken a neuron for each joint angles of left hip, right hip, left knee, right knee (the activity function of these neurons is PureLin) and the output of each Matsuoka elements. By multiplying by a separate weight would be exerted to the neurons in charge. Multiplying the weight in matsuoka’s signals causes to change the output wave length to each neuron and turn to the quantity required for each joint. Eventually, A bias has been assigned to every Purelin’s neurons. This bias can shift the location of the generated signals in the degree axis of each joint to the negative or positive direction (on the other hand, to add a permanent offset as the same quantity of the input signal of every joint). Attention to this point that researchers are always looking for small the search space, bias and weight in each one of left hip, right hip and left knee-right knee neurons can be chosen the same separately (because the movement of each of these joints is isochronous and they have just half of the period difference at the time of occurrence) that this can cause the search space to be smaller. after exerting achieved results from neural oscillators on the robot joints, left hip and right hip joint angles would be exerted to Matsuoka’s neurons that is responsible for generating their signals respectively.

![Fig. 2. Matsuoka Coupled Neural Oscillator. When a constant input is fed to an adaptive neural element, its responsive output first approaches the constant input, but then after some point the output declines because of the adaptivity of the element.](image-url)

### 4 HARMONY SEARCH ALGORITHM

#### 4.1 Harmony Search Algorithm Definition
Harmony Search (HS) algorithm which is a nature-inspired algorithm, mimicking the improvisation process of music players has been recently developed by Geem et al. [18]. The HS algorithm is simple in concept, few in parameters, and easy in implementation, it has been successfully applied to various benchmarking and real-world problems including travelling salesman problem [19]. The main steps of HS are as follows [19]:

Step 1: Initialize the problem and algorithm parameters.
Step 2: Initialize the harmony memory.
Step 3: Improvise a new harmony.
Step 4: Update the harmony memory.
Step 5: Check the stopping criterion.

These steps are described in the next five subsections.

4.1.1 Initialize the Problem and Algorithm Parameters

As demonstrated in this document, the numbering for sections upper case Arabic numerals, then upper case Arabic numerals, separated by periods. Initial paragraphs after the section title are not indented. Only the initial, introductory paragraph has a drop cap.

In the first step, the optimization problem is specified as follows:

Minimize (or Maximize) $f(x)$ Subject to,

$$x_i \in X_j, i = 1, 2, ..., N$$

Where $f(x)$ is an objective function; $x$ is a solution vector.

4.1.2 Initialize the Harmony Memory

In Step 2, the Harmony Memory (HM) as shown in Equation 7, is crammed with as many randomly generated solution vectors as the size of the HM (i.e., HMS).

$$HM = \begin{bmatrix} 1 & 1 & 1 \\ x_1 & x_2 & ... & x_N \\ 1 & 2 & 2 \\ x_2 & x_2 & ... & x_N \\ \vdots & \vdots & \ddots & \vdots \\ HCM & HCM & HCM \end{bmatrix}$$

(6)

4.1.3 Improvise a New Harmony

A New Harmony vector $x' = (x'_1, x'_2, ..., x'_N)$ is improvised by following three rules: (1) random selection, (2) HM consideration, and (3) pitch adjustment. Generating a new harmony is called ‘improvisation’.

In the memory consideration, the value of the first decision variable ($x'_1$) for the new vector is chosen from any of the values in the specified HM range ($x'_1 \in X_{HM}^{HMS}$). Values of the other decision variables ($x'_2, x'_3, ..., x'_N$) are chosen in the same manner. The HMCR, which varies between 0 and 1, is the rate of choosing one value from the historical values stored in the HM, while (1 - HMCR) is the rate of randomly selecting one value from the possible range of values.

$$x'_i \left\{ \begin{array}{ll} 1 & x'_i \in \{x_1, x_2, \cdots, x_N\} \\ 1 & \text{with probability } \text{HMCR} \\ x'_i \in X_j & \text{with probability } \left(1 - \text{HMCR}\right) \end{array} \right.$$  \hspace{1cm} (7)

Every component obtained by the memory consideration is examined to determine whether it should be pitch-adjusted. This operation uses the PAR parameter, which is the rate of pitch adjustment as follows:

$$\text{pith Adjusting Decision for } x'_i \left\{ \begin{array}{ll} \text{Yes with probability } \text{par} \\ \text{No with probability } (1 - \text{par}) \end{array} \right.$$  \hspace{1cm} (8)
The value of (1 - PAR) sets the rate of doing nothing. If the pitch adjustment decision for \( x_i \) is YES, is \( x_i \) replaced randomly with another value. In Step 3, HM consideration, pitch adjustment or random selection is applied to each variable of the new harmony vector in turn.

In the case of combinational optimization problems with discrete representation or permutation encoding of solutions, the process of applying PAR parameter must be adapted accord to problem objective.

### 4.1.4 Update Harmony Memory

If the new harmony vector \( x = (x_1, x_2, ..., x_n) \) is better than the worst harmony in the HM, judged in terms of the objective function value, the new harmony is included in the HM and the existing worst harmony is excluded from the HM.

### 4.1.5 Check Stopping Criterion

If the stopping criterion (i.e. maximum number of improvisations) is satisfied, computation is terminated. Otherwise, Steps 3 and 4 are repeated.

### 4.2 Harmony Operators According to Neural Oscillators Parameters

Neural oscillators parameters can be used directly with HS algorithm. Each solution corresponds to a TFS motion generator. Each solution in harmony memory, harmony vector, is encoded as an array of 10 elements, six elements for Matsouka oscillator variables and time periods and four other elements for weights and biases, where each element is a real variable.

#### 4.2.1 Initialization

For initialization, HM is filled with as many randomly generated solution vectors as the size of the HM (e.g. HMS). In the complete, each random generated vector is a valid neural oscillator parameter but for applying to robot, the probability that randomly generated neural oscillators are valid is low. So, alternatively, we develop an intelligence table for initialization (table 2). The lengths of intervals have been taken according to maximum and minimum degree of freedom of every joint of the robot. For example for knee joint angle, when \( \tau_1 \) and \( \tau_2 \) are between 0 and 25, the largeness of Matsouka’s signal wavelength would be between 0 and 25, and by multiplying this signal by \( \omega_1 \), the whole search space would be covered, and by choosing \( b_1 \) between -5 and -125 this wavelength can generate signal by each virtual primal offset.

#### 4.2.2 Improvisation Step

New harmony vector is generated based on process of part 3 in previous Section. Each element of harmony vector is selected from HM with probability HMCR and with probability \( (1 - \text{HMCR}) \) is selected from set \{1, 2, ..., 10\}.

In the pitch adjusting process, a value is set to a number in range from lower band to upper band with probability of PAR, or just stays in its original value with probability \( (1 - \text{PAR}) \).

### 4.2.3 Fitness function

#### Table 2

| Symbol | Upper Bound | Lower Bound |
|--------|-------------|-------------|
| \( \tau_1 \) | 25 | 0 |
| \( \tau_2 \) | 25 | 0 |
| \( \alpha \) | 5 | -5 |
| \( \beta \) | 5 | -5 |
| \( \gamma \) | 5 | -5 |
| \( c \) | 2 | 4 |
| \( w_{11} \) | 1 | -4 |
| \( w_{12} \) | 0 | -5 |
| \( b_1 \) | 20 | -95 |
| \( b_2 \) | -5 | -125 |

Fitness function has a critical role in heuristic algorithms and is used to judge how well a solution represented by a vector is. To achieve more stable and faster walk, a fitness function based on robot’s straight movement with having limited time for walking is assumed. The amount of deviation from straight walking is subtracted from the fitness as a punishment to force the robot to walk straight. The simulator runs for 15 seconds each time, during first 3 seconds robot is at lock phase. We assume when the robot is initialized each time, its X and Y values equal to zero and fitness function is calculated whenever robot falls or time duration for walking is over. Equation 9 shows the pseudo code of walking fitness function.

\[
\text{if} \ (\text{CurrentTime} \geq \text{TimeDuration}) \ \text{on} \\
\text{fitness} = X - Y;
\]

\[
\text{if} \ (\text{RobotIsFallen}) \ \text{on} \\
\text{fitness} = \frac{X - Y}{\text{TimeDuration} \ - \ \text{CurrentTime} \ \text{on}};
\]

According to equation 9 when robot falls down during walk the fitness is divided to remaining time of simulation. This punishment forces the robot to achieve a stable walk.

### 5 Results
We ran the simulator on a Pentium IV 3 GHz Core 2.6 Duo machine with 4 GB of physical memory, with Linux SUSE 10.3 O.S. The time period for the simulation was 15 seconds. 2 hours after starting HS under the MATLAB, 750 trials were performed. Fig. 4 shows the average and best fitness values during these 750 trials.

Running the GA on the same system with 10 generations and 200 populations for each generation, after 7 hours results of figure 5 have been achieved.

Gait period at the best found fitness equals to 0.1 s. according to this consequence figures 6, 7 and 8, respectively show angular trajectories of hip, knee and ankle, generated by oscillators.

Considerate towards the punishment set for changing direction of robot during movement, Fig. 9 shows the walking direction of the robot.

According to the fact that during movement, pivot leg changes and this change causes changing human’s hight, and we know that the ideal walking for a biped robot is a
walking that is more similar to human walking. Thus, changes in height during walk are mandatory for a good movement. Figure 10 that illustrates the z-position of robot’s torso during movement shows that our offering algorithm is much the same as human walking.

5 Conclusion

We have demonstrated the suitability of an evolutionary robotics approach to the problem of stable three-dimensional biped walking in simulation. The current implementation is capable of walking in a straight line on a planar surface with the use of proprioceptive input. When the neural network sends the signals to the robot, the input layer receives the feedback signals from hip of the robot on the other hand, in this study for the first time HS used for optimization of neural network weights. Using from HS as an optimizer shows that the harmony search algorithm is a computationally fast multi-objective optimizer tool for complex engineering multi-objective optimization problems. The main advantage of our walking controller model is having the least parameters compare with other gait generator models that it can helps the robot to find best solution sooner and sooner. It is also capable of being implemented on any kinds of humanoid robots without considering its physical model.

References

[1] Hirukawa H., “Walking biped humanoids that perform manual labour,” Philos Trans A Math Phys Eng Sci. PMID: 17148050, pp.65–77, 2007.
[2] Vukobratovic M., Borovac B., and Surdilovic. D., “Zero-moment point proper interpretation and new applications.” In: Proceedings of the 2nd IEEE-RAS International Conference on Humanoid Robots, pp. 237–244, 2001.
[3] S. Zhang, C. Zhu, J. K. O. Sin, and P. K. T. Mok, “A novel ultra-thin elevated channel low-temperature poly-Si TFT,” IEEE Electron Device Lett., vol. 20, pp. 569–571, Nov. 1999.
[4] Kajita S., Kanehiro F., Kaneko K., Yokoi K. and Hirukawa H., “The 3D linear inverted pendulum mode A simple modeling for a biped walking pattern generation.” In: Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 239–246, 2001.
[5] J. K. Hodgins, “Three-dimensional human running,” in Proc. IEEE Int. Conf. Robotics and Automation, vol. 4, 1996, pp. 3271–3276.
[6] L. Yang, et al., “Truncated Fourier series formulation for biped walking balance control,” www.robotica.org: page 1 of 16. © 2009 Cambridge University Press.
[7] Yang, L., Chew, C.M., Poo, A.N. (2006). “Adjustable Bipedal Gait Generation using Genetic Algorithm Optimized Fourier Series Formulation.” Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, 4435-4440.
[8] Y. Fukuoka, H. Kimura, and A. Cohen, “Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts,” Int. J. Robot. Res., vol. 22, no. 3, pp. 187–202, 2003.
[9] K. Tsujita, K. Tsuchiya, and A. Onat, “Adaptive gait pattern control of a quadruped locomotion robot,” in Proc. Int. Conf. Intell. Robots Syst., 2001, pp. 2318–2325.
[10] S. Aoi, K. Tsuchiya, and K. Tsuchita, “Turning control of a biped locomotion robot using nonlinear oscillators,” in Proc. Int. Conf. Robot. Autom., 2004, pp. 3043–3048.
[11] J. Nakanishi, J. Morimoto, G. Endo, G. Cheng, S. Schaal, and M. Kawato, “Learning from demonstration and adaptation of biped locomotion,” Robot. Auton. Syst., vol. 47, pp. 79–91, 2004.
[12] S. Aoi and K. Tsuchiya, “Locomotion control of a biped robot using nonlinear oscillators,” Auton. Robots, vol. 19, no. 3, pp. 219–232, 2005.
[13] J. Ijspeert, “A connectionist central pattern generator for the aquatic and terrestrial gaits of a simulated salamander,” Biol. Cybern., vol. 84, pp. 331–348, 2001.
[14] Matsuoka K., “Sustained oscillations generated by mutually inhibiting neurons with adaptation.” Biol. Cybern. 52, 367–376, 1985.
[15] G. Endo, J. Nakanishi, J. Morimoto, and G. Cheng, “Experimental studies of a neural oscillator for biped locomotion with QRIO.” in Proc. ’05 IEEE Int. Conf. Robot. Auton. (ICRA’05), pp. 596–602.
[16] S. Miyakoshi, G. Taga, Y. Kuniyoshi, A. Nagakubo, “Three dimensional bipedal stepping motion using neural oscillators towards humanoid motion in the real world,” in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 1998, pp. 84-89

[17] Geem, Z.W., J.H. Kim and G.V. Loganathan, 2001. “A new heuristic optimization algorithm: Harmony search. Simulation,” 76: 60-68.

[18] K.S. Lee, Z.W. Geem, “A new meta-heuristic algorithm for continues engineering optimization: harmony search theory and practice,” Computer Methods in Applied Mechanics and Engineering 194 (2004) 3902–3933.

[19] Z.W. Geem, C. Tseng, Y. Park, “Harmony search for generalized orienteering problem: best touring in China,” Springer Lecture Notes in Computer Science 3412 (2005) 741–750.

[20] Smith, R.: Homepage of Open Dynamics Engine project http://www.ode.org.

[21] Boedecker J. “Humanoid Robot Simulation and Walking Behaviour Development in the Spark Simulator Framework.” Technical report, Artificial Intelligence Research University of Koblenz (2005)

[22] Kagami, S., Mochimaru, M., Ebara, Y., Miyata, N., Nishiwaki, K., Kanade, T., Inoue, H. “Measurement and comparison of humanoid H7 walking with human being.” Robotics and Autonomous Sys. vol. 48, pp. 177–187, 2003.

Ebrahim Yazdi obtained his bachelor degree in Computer-software engineering from Qazvin Azad University, Qazvin, Iran. Currently, he is a master student in Department of computer-software engineering, Qazvin Azad University, Qazvin, Iran. He is a member of Mechatronic Research Lab. (MRL), Qazvin, Iran. He got third place in Atlanta Robocup 2007 3DDevelopment league and got first place in China Robocup 2008 3DDevelopment league.