Biped Robot Walking using a Combination of Truncated Fourier Series and GALA

(Genetic Algorithm parameters adaption using Learning Automata)

Omid Mohamad Nezami

Mechatronics Research Laboratoy (MRL)
Qazvin Islamic Azad University
Qazvin, Iran
omid.mnezami@mrl.ir

Mohammad Reza Meybodi

Computer Engineering Department
Amirkabir University of Technology
Tehran, Iran
meybodi@ce.aut.ac.ir

Abstract—Controlling a biped robot with a high degree of freedom to achieve stable, straight and fast movement patterns is one of the most complex problems. With growing computational power of computer hardware, simulation of such robots in high resolution real time environment has become more applicable. This paper introduces a novel approach to Generate Bipedal gait for humanoid locomotion. In this scene, first we have used a modified Truncated Fourier Series (TFS) to generate angular trajectories, then to find the best angular trajectory we built an improved Genetic Algorithm (GA). One of the major difficulties of GAs is choosing appropriate values for mutation and crossover parameters. Hence, we present GALA (Genetic Algorithm parameters adaption using Learning Automata) to adjust these parameters by recruiting Learning Automata. As results show, my approach could generate better values for angular trajectories for biped walking, hence in my approach the robot could walk with high stability and faster than other approaches. Evaluations performed on Simulated NAO robot in RoboCup 3D soccer simulation environment.

Keywords-Bipedal Locomotion; Learning Automata; Genetic Algorithm; Truncated Fourier Series

I. INTRODUCTION

In recent years, bipedal locomotion, especially "bipedal walking" has been one of the interesting research topics in multi disciplinary arena. Bipedal walking as a very complex motion, involves most of the humanoid joints including its sensors and actuators. Many researchers have focused on this issue and a lot of approaches have been presented. There are two major approaches in bipedal walking researches; model-based and model-free approaches [2, 3]. In model-based approach, the designer first derives the model of the robot and then builds a controller for the model. Two well known methods in this approach are "Zero Moment Point"[4] (ZMP) and "Inverted Pendulum"[5].

In model-free approach, which is also called "Dynamics Based", it is common to make use of the sensory information and associate it with motions. No physical model is used in this method that eases the implementation of the skills. There are three important studies done in this field; Passive Dynamic Walking (PDW) [6], Central Pattern Generator (CPG) [7] and Ballistic Walking [8]. In PDW approach, the robot does not have any actuator model and it walks just by utilizing the gravity force. The Ballistic walking is originated from the simple human walking model based on the observation of human walking in which the muscles of the swing leg are activated only at the beginning and the end of the swing phase. In CPG approach, special neural circuits take the role of the rhythmic walking controller using the non-linear equations to model the neural activities. Researchers usually focus on complex mathematical models like Hopf [9] or Matsuoka [10] to model these neural activities and generate rhythmic walk patterns (Gait).

In 2006, Truncated Fourier Series (TFS) formulation is used for gait generation in bipedal locomotion [11]. TFS together with a ZMP stability indicator is used to prove that TFS can generate suitable angular trajectories for controlling bipedal locomotion. It does not require inverse kinematics and Stable gaits with different step lengths and stride frequencies can be readily generated by changing the value of only one parameter in the TFS.

Taking the advantages of TFS as a model-free approach, we present a novel approach to Generate Bipedal gait for humanoid locomotion based on TFS. For this purpose, we used a modified TFS to generate angular trajectories. To find the best angular trajectory in TFS, an adapted GA is obtained. One of the major difficulties of GAs is choosing appropriate values for mutation and crossover parameters. Hence, we present GALA to adjust these parameters by recruiting Learning Automata (LA).

The remainder of this paper as follows; in the next section, we explain the simulation environment and biped model. Section III is devoted to TFS. In this section, we will talk about how the TFS used for trajectory generation. An overview of GA and LA are presented in section IV. In section V we introduced the GALA, novel algorithm, which is used to find the best angular trajectory. Experimental results and conclusions are discussed in sections VI and VII.

II. SIMULATOR AND BIPED MODEL

In this paper, a new approach for walking behavior in a simulated humanoid robot will be discussed.

The simulation is performed by Rcssserver3d simulator which is a generic three-dimensional simulator based on Spark and Open Dynamics Engine (ODE). 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 [12]. The robot in this study is a simulated model of NAO that is a real

humanoid Robot with two arms, two legs and a head. This robot has 4.5kg weigh, 57cm height and 22 degrees of freedom (DOF). There are six DOFs in each leg; two in the hip, two in the ankle one at the knee. And an additional DOF that exists at each leg's hip for yaw causes the legs to rotate outward and inward.

As a test-bed, in our soccer simulation team (MRL) we have implemented and tested our new bipedal locomotion approach on this simulated NAO robot and generated software based on this simulator that is developed by MRL team from scratch. According to our studies, to able robot fast walking, 6 DOFs (three for each leg) are more effective than other DOFs. The DOFs of hip, knee and ankle which move on the same plane of forward-backward are the major ones. Although other DOFs are effective in walking behavior, but in fact, their role smoothes the robots walking motion, So here, it's preferred to ignore them to decrease learning search space. Like in [13], 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.

III. TFS GAIT GENERATOR

Bipedal walking as a complex motion, involves most of humanoid robot's joints. Researchers attempt to imitate the human walking style as well as its speed. Therefore analyzing human walk pattern has been used for acquiring beneficial information about this motion. Human walk has been investigated from many angles; walking trajectory is one of them.

The walking trajectory is divided into several types. Positional trajectory and angular trajectory are two of them. In angular trajectory, the angle of each joint is plotted at a certain time slice. Therefore the angular trajectory is obtained by angular variation of each joint. Biped angular trajectory of two joints; hip and knee captured from human walking are shown in "Fig. 1".

The angle of each joint in one period of walking signal from t_0 to t_6 is represented in "Fig. 2" by capturing the main features of "Fig. 1" and gives a general form to make it applicable to robots. In time range $[t_0, t_2]$ and $[t_5, t_6]$ the left
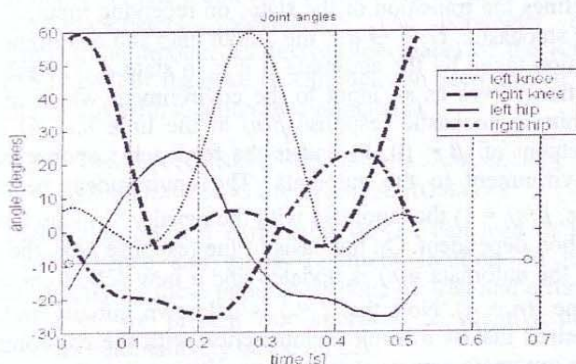

Figure 1. Human walking angular trajectory (Angles of hip and knee captured from human walking) [14].
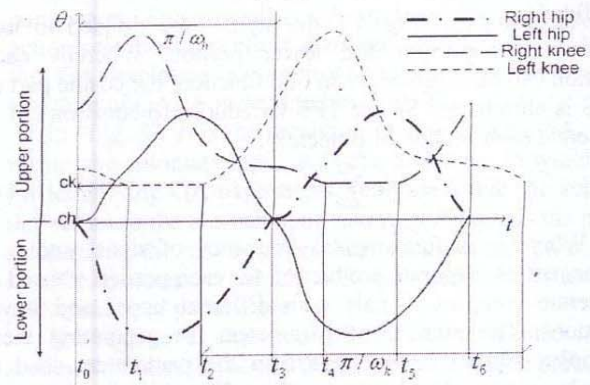

Figure 2. gaits elaborated from human gaits features (The angle of hip and knee joints in one period of walking signal from t_0 to t_6)[14].

leg is support leg and the right one is swing leg but in range of time $[t_2, t_5]$ the left and right legs play the role of support and swing legs respectively. In another word, in two times of t_2 and t_5 the roles of two legs are switched with each other. At time t_3 where two hip trajectories intersect, two thighs cross each other.

A. Angular trajectory generation

Regarding the fact that all joint trajectories of human walking are periodic and similar to sine or cosine signals [15], the generation of these angular signals can be done by Fourier series.

B. Basic Fourier series

The original definition of Fourier series is described by following formula:

$$\mathbb{F}(t) = \frac{a_0}{2} + \sum_{i=1} (a_i \cos \frac{i\pi t}{L} + b_i \sin \frac{i\pi t}{L}) \qquad (1)$$

The first term ($\frac{a_0}{2}$) of equation (1) represents the DC bias of the signal and the L represents half of the largest period that exists in the signal.

By $w = \frac{\pi}{L}$ then the frequency form of Fourier series is achieved as follows:

$$\mathbb{F}(t) = \frac{a_0}{2} + \sum_{i=1} (a_i \cos(iwt) + b_i \sin(iwt)) \qquad (2)$$

Where w is frequency of periodic signal. Any complicated signal can be produced by this formula when i is considered infinite. But when the value of i is limited to a definite number, precision of generating signal is reduced and this type of Fourier series is called partial sum of the Fourier series. According to "Fig. 1", Human Walking angular trajectories are too complicated to be produced by a definite Fourier series band limited to the second harmonic. Therefore a modified definite Fourier series as a Truncated Fourier series (TFS) is used in this study.

C. Trajectory generation using TFS

According to "Fig. 1", the signals are divided in two parts; upper portion and lower portion. Whereby each portion can be assumed as an odd function, the cosine part of TFS is eliminated. So the TFS is reduced to equation (3) to generate each portion of trajectory.

$$F(t) = a + \sum_{i=1}^{n} b_i \sin(iwt) \qquad (3)$$

Where w is fundamental frequency of signal and a is signal offset. Separate production for each portion, caused to generate complex signals with different upper and lower portions. The number of parameters for generating these complex signals is also less than the parameters used in Fourier series. As shown in "Fig. 2", each signal has an offset. c_h, c_k are hip trajectory and knee trajectory offsets respectively. From t_0 to t_2 the left leg is considered as supporting leg and the variation of its knee angle is so minute that can be assumed fixed. This duration of walking is named lock phase. In addition, the amount of shift phase of the two leg trajectories signal is half of the period of each signal. The trajectories for both legs are identical in shape but are shifted half of the walking period in time. Therefore by figuring out walking angular trajectory of one leg the other leg trajectory is obtained. Using equation(3) and considering curves of "Fig.1", the TFS for generating each portion of hip and knee trajectories are formulated as equation (4):

$$\theta_k^+ = \sum_{i=1}^{n} C_i . \sin(iw_k t_2) + c_k, w_k = \frac{2\pi}{T_k}$$

$$\theta_k^- = c_k \geq 0$$

$$\theta_h^+ = \sum_{i=1}^{n} A_i . \sin(iw_h t_3) + c_h, w_h = w_k \qquad (4)$$

$$\theta_h^- = \sum_{i=1}^{n} B_i . \sin(iw_h t_6) + c_h, w_h = w_k$$

In these equations, the plus (+) sign represents the upper portion of walking trajectory and the minus (-) shows the lower portion. A_i, B_i, C_i are constant coefficients for generating signals. The h and k index stands for hip and knee respectively. c_h, c_k are signal offsets and T_k is assumed as period of knee trajectory. Considering the fact that all joints in walking motion have equal movement frequency [2, 3, 15], the equation $w_k = w_h = \frac{2\pi}{T_k}$ can be concluded.

Parameter t_3 shows the end time of hip trajectory in upper portion and starts its down portion, t_6 shows the end time down portion. These parameters are not significant since they can be obtained when the hip trajectory intersects the c_h line. But parameter t_2 represents the end time of knee lock phase and must be considered to produce knee trajectory. Therefore Truncated Fourier series parameters to produce trajectories are: $c_h, c_k, A_i, B_i, C_i, t_2, w_k$. In this essay there are some constraints to be dealt with as shown in the following equation:

$$0 < t_2 < T_k, w_k = \frac{2\pi}{T_k} \Rightarrow 0 < t_2 < \frac{2\pi}{w_k} \qquad (5)$$

$$c_k \geq 0$$

Finally an optimization algorithm is needed to optimize a 7_dimension Problem for finding the best gait generator.

IV. OVERVIEW OF GENETIC ALGORITHM AND LEARNING AUTOMATA

In this section, in all brevity, we discuss the fundamentals of genetic algorithm and learning automata.

A. Genetic Algorithm

The sequence of Genetic Algorithm(GA) implementation is summarized as follows [16]:

1. Select a fitness function (the goal for the optimization process).
2. Initialize population.
3. Evaluate population.
4. While (number of generations has been generated)
   4.1 Begin.
   4.2 Select parents for reproduction.
   4.3 Perform crossover and mutation.
   4.4 Evaluate population.
   4.5 End.

B. Learning Automata

Learning automata (LA) can be classified into two main families, fixed and variable structure learning automata [1].

A fixed structure learning automata (FSLA) is a quintuple $< \alpha, \phi, \beta, F, G >$, $\alpha = (\alpha_1, ..., \alpha_R)$ where is the set of actions that it must choose from. $\phi = (\phi_1, ..., \phi_s)$ is the set of states. $\beta = \{0, 1\}$ is the set of inputs, where 1 represents a penalty and 0 represents a reward. $F: \phi \times \beta \to \phi$ is a map called the transition map. It defines the transition of the states on receiving input, F may be stochastic. $G: \phi \to \alpha$ is the output map and determines the action taken by the automata if it is in state ϕ. The selected action serves as an input to the environment which in turn emits a stochastic response $\beta(n)$ at the time n. $\beta(n)$ is an element of $\beta = \{0, 1\}$ and is the feedback response of the environment to the automata. The environment penalizes (i.e. $\beta(n) = 1$) the automata with the penalty c_i, which is the action dependent. On the basis of the response $\beta(n)$, the state of the automata $\phi(n)$ is updated and a new action chosen at time $(n + 1)$. Note that $\{c_i\}$ is unknown initially and it is desired that as a result of interaction with the environment, the automata arrives at the action which presents it with the minimum penalty response in an expected sense. If the probabilities of the transition from one state to another and

probabilities of correspondence of action and state are fixed, the automata is said to be a fixed-structure learning automata and otherwise the automata is said to be a variable structure learning automata (VSLA). In this paper we use the VSLA type.

Variable structure learning automata is represented by the sextuple $< \beta, \phi, \alpha, P, G, T >$, where β is a set of inputs actions, ϕ is a set of internal states, α a set of outputs, P denotes the state probability vector governing the choice of the state at each stage k, G is the output mapping, and T is learning algorithm. The learning algorithm is a recurrence relation and is used to modify the state probability vector.

It is evident that the crucial factor affecting the performance of the variable structure learning automata is learning algorithm for updating the action probabilities. Let α_i be the action chosen at time k as a sample realization from distribution $p(k)$. The linear reward-inaction algorithm (L_{R-I}) is one of the earliest schemes. In an L_{R-I} scheme, the recurrence equation for updating is defined as equation (6).

$$p_j(k) = \begin{cases} p_j(k) + a(1 - p_j(k)) & if\ i = j \\ p_j(k)(1-a) & if\ i \neq j \end{cases} \quad (6)$$

If β is zero and P is unchanged if β is one. The parameter a, which is called step length, determines the amount of increase(decreases) of the action probabilities. In linear reward-penalty algorithm (L_{R-P}) scheme, the recurrence equation for updating p is defined as equation(7) and (8).

$$p_j(k) = \begin{cases} p_j(k) + a(1 - p_j(k)) & if\ i = j \\ p_j(k)(1-a) & if\ i \neq j \end{cases} \quad (7)$$

if $\beta(k) = 0$, and

$$P_j(k) = \begin{cases} P_j(k)(1-b) & if\ i = j \\ \dfrac{b}{r-1} + (1-b)P_j(k) & if\ i \neq j \end{cases} \quad (8)$$

If $\beta(k) = 1$. The parameters a and b represent reward and penalty parameters, respectively. The parameter $a(b)$ determines the amount for increase(decrease) of the action probabilities. For the sake of simplicity in presentation, we denote VSLA with k action by automata(k).

V. GALA (GENETIC ALGORITHM PARAMETERS ADAPTION USING LEARNING AUTOMATA)

Identifying optimal parameters of GA can lead us to a fast convergence and global optimum solution. In most cases, specifying these parameters done by trial and error. In this section, we present our approach to identify these parameters automatically based on LA.

The main idea is as follows; at first LA identifies the GA parameters by applying a set of actions, and then the GA uses these parameters for specific generation and eventually

sends a feedback to LA by estimating the performance of parameters. Subsequently, LA updates the probability of action selection dependent upon outcome feedbacks. In fact, the GA exploited as environment for LA. The overall structure of GALA is depicted in "Fig. 3"

In "Fig. 3", β Is the receiving feedback from GA, which means the automata input, α Is the action set in which used for identifying GA parameters, *Individuals* are the solutions that return to the environment, and *Best Fitness* is the perfect solution to the environment that GA earned after some specific generation, which will be used to evaluate the performance of parameters.

The high level algorithm of GALA represented in next section. Initially, we assign the identical selection probability to every action. Suppose if we have n possible action, the probability of choosing every action is $\dfrac{1}{n}$ at the beginning of the algorithm.
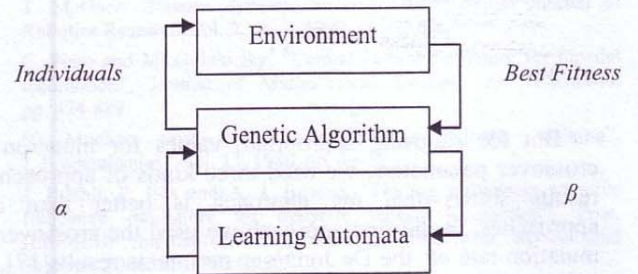

Figure 3. GALA diagram

1. Begin

2. Choose action(i) of LA based on Roulette wheel selection.

3. Set this action as the parameter of GA and apply GA to the environment.

4. Compare best fitness received from the environment with the average of preceding acquired fitness,

4.1. If the fitness is better than the average fitness (that means we earn the favored feedback), update the probability of actions by the following equation,

$$p_j(k) = \begin{cases} p_j(k) + a(1 - p_j(k)) & if\ i = j \\ p_j(k)(1-a) & if\ i \neq j \end{cases}$$

4.2. Else

$$P_j(k) = \begin{cases} P_j(k)(1-b) & if\ i = j \\ \dfrac{b}{r-1} + (1-b)P_j(k) & if\ i \neq j \end{cases}$$

5. End.

The parameter $a(b)$ determines the amount for increase (decrease) of the action probabilities. We considered these equations in section IV.

VI. EXPERIMENTAL RESULTS

In this section, we evaluate the GA to find the best parameters for generating angular trajectories. According to 7 parameters of TFS which mentioned in section III we consider 7 genes in every chromosome of GA and roulette wheel is assumed as selection method. Population for each generation is 100 and termination condition is to have a generation counter greater than 10. To achieve more stable and faster walk, a fitness function implemented based on robot's straight movement with having limited time for walking. Fitness function is calculated when the robot falls or time duration for walking is finished. Equation(9), is assumed fitness function formulation, where the robot is initialized in x = y = 0 and time duration for walking is assumed as 15 seconds.

$$\text{if} \left(\begin{array}{l} (\text{Test Time} >= \text{time duration for walking}) \text{ or} \\ (\text{robot is fallen}) \end{array} \right) \quad (9)$$

$$\text{Fitness} := 10 * x;$$

End if

But for choosing appropriate values for mutation and crossover parameters, we used three kinds of approach and results shows that my approach is better than other approaches. In the first approach we used the crossover and mutation rate of the De Jong's experiments results[17], that indicated the best crossover rate was 0.6 and the best mutation rate was 0.001. Convergence diagram of GA after 10 generation with use of these values of crossover and mutation rate shown in "Fig. 4". In this approach the best fitness is 5.58956, hence the robot could walk 0.54 meters in 15 seconds with average body speed of 0.036 m/s and the period of 0.41s for each step.
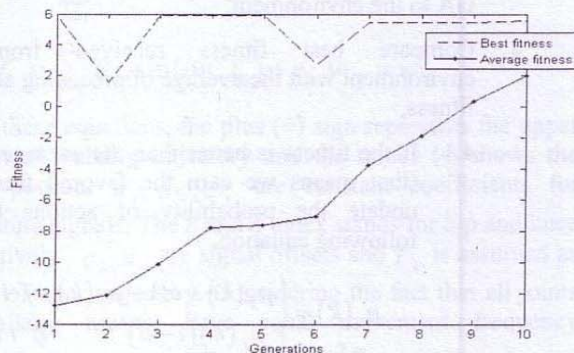

Figure 4. Convergence diagram of GA after 10 generation with use of De Jong's parameters (Crossover rate = 0.6 and mutation rate = 0.001).

For second approach we used the crossover and mutation rate of the Grefenstette's experiments results, that indicated the best crossover rate was 0.95 and the best mutation rate was 0.01[17]. Convergence diagram of GA after 10 generation with use of these values of crossover and mutation rate shown in "Fig. 5". In this approach the best fitness is 39.5016, hence the robot could walk 4.12 meters in

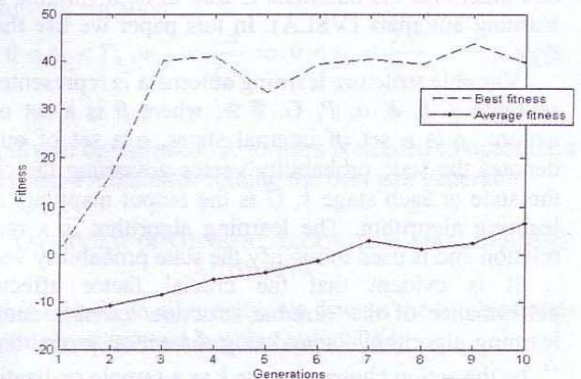15 seconds with average body speed of 0.27 m/s and the period of 0.41s for each step.

Figure 5. Convergence diagram of GA after 10 generation with use of Grefenstette's parameters (Crossover rate = 0.95 and mutation rate = 0.01).

For last approach we evaluated our approach to find the appropriate values for mutation and crossover parameters. Hence as we mentioned in section V, We considered a set of constant values for crossover rate in the range of 0.1 to 0.9 steps by 0.1 and for every value of crossover rate, we execute a GALA algorithm with N = 100, a = 0.1, b = 0.2 and automata(10) to find an optimum value of mutation rate with action set from 0.00 to 0.09 steps by 0.01. After applying GALA with above conditions upon TFS, we will have a set of optimal crossover and mutation pairs and best fitness values will be calculated by using mentioned pairs. These pairs and best fitness values are shown in Table I.

TABLE I. Optimal crossover and mutation pairs and best fitness values will be calculated by using mentioned pairs.

Crossover rate	Optimal mutation rate	Best fitness
0.1	0.08	12.4560
0.2	0.08	23.3489
0.3	0.06	31.2345
0.4	0.09	35.0432
0.5	0.04	39.0234
0.6	0.05	42.3210
0.7	0.07	41.2623
0.8	0.07	46.4982
0.9	0.06	37.2310

According to table I, in this problem best parameters for GA are 0.8 for crossover rate and 0.07 for mutation rate, because biggest "best fitness value" occurs in these parameters and the robot could walk 4.95 meter in 15 second with average body speed of 0.33 m/s and the period of 0.41s for each step. In best state of table I, convergence diagram of GALA after 100 iteration shown in "Fig. 6".
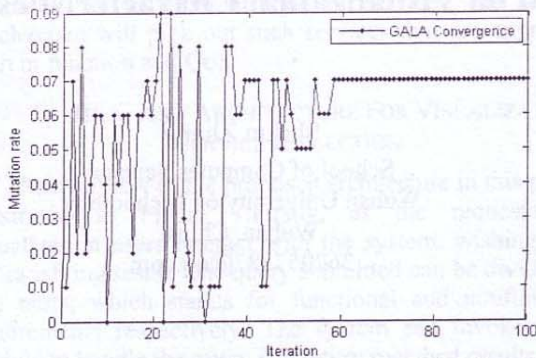
Figure 6. Convergence diagram of GALA after 100 iteration, for finding best value of mutation rate with crossover rate = 0.8.

Also Convergence diagram of GA after 10 generation with use of optimal values of crossover and mutation rate (Crossover rate = 0.8, Mutation rate = 0.07) shown in "Fig. 7".
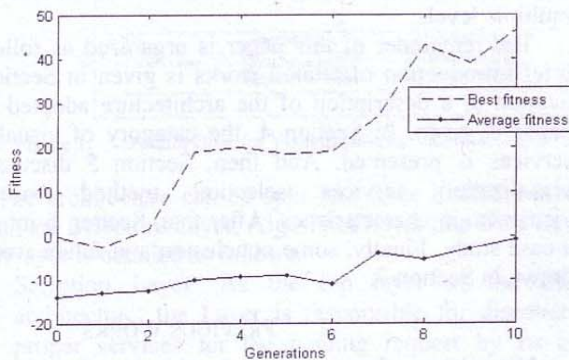

Figure 7. Convergence diagram of GA after 10 generation with use of optimal value of parameters (Crossover rate = 0.8 and mutation rate = 0.07).

As results show, my approach could generate better values for angular trajectories for biped walking, because my approach could achieve biggest fitness value, hence in my approach the robot could walk faster than other approaches.

VII. CONCLUSION

In this paper for first time TFS with GALA is implemented in a simulated robot that can walk fast and stable. This technique has some advantages. First, it can be implemented on many humanoid robots without considering any mathematical modeling. Second, by using GALA the robot has achieved an optimal walk, that means GALA choosing best values for mutation and crossover rates and by use these values can achieves optimal solution of problems.

REFERENCES

[1] M. R. Meybodi and H. Beigy, "New Learning Automata based algorithms For adaption of Backpropagation algorithm parameters", International Journal of Neural Systems, vol. 12, no. 1, 2002, pp. 45-67.

[2] N. Shafii, S. Aslani, O. Mohamad Nezami and S. Shiry, "Evalation of biped walking using Truncated Fourier Series and Particle Swarm Optimization", Lecture Notes in Computer Science, vol. 5949, pp. 344 -354, 2010.

[3] N. Shafii, S. Aslani, S. M. H.S.Javadi, Vahid Azizi and O. Mohamad Nezami, "Robust humanoid walking using Truncated Fourier Series gait generator", Proc. Robocup Iran Open Symposium, April 2009.

[4] M. Vukobratovic, B. Borovac and D. Surdilovic, "Zero Moment Point proper interpretation and new applications", Proc. 2nd IEEE-RAS. International Conference on Humanoid Robots, 2001, pp. 237-244.

[5] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi and H. Hirukawa, "The 3D linear Inverted Pendulum mode: a simple modeling for a biped walking pattern generation", Proc. IEEE/RSJ. International Conference on Intelligent Robots and Systems, 2001, pp. 239-246.

[6] T. McGeer, "Passive dynamic walking", International Journal of Robotics Research, vol. 9, no. 2, 1990, pp. 62-82.

[7] C. Pinto and M. Golubitsky, "Central Pattern Generator for bipedal locomotion", Journal of Mathematical Biology, vol. 53, 2006, pp. 474-489.

[8] S. Mochon and T. A. McMahon, "Ballistic walking", J. Biomechanics, vol. 13, 1980, pp. 49-57.

[9] J. Buchli, F. Iida and A. J. Ijspeert, "Finding resonance: adaptive frequency oscillators for dynamic legged locomotion", Proc. IEEE/RSJ. International Conference on Intelligent Robots and Systems, 2006, PP. 3903-3910.

[10] K. Matsuoka, "Sustained oscillations generated by mutually inhibiting neurons with adaptation", Biol. Cybern, vol. 52, 1985, pp. 367-376.

[11] L. Yang, C. M. Chew and A. N. Poo, "Adjustable bipedal gait generation using Genetic algorithm optimized Fourier Series formulation", Proc. IEEE/RSJ. International Conference on Intelligent Robots and Systems, 2006, PP. 4435-4440.

[12] J. Boedecker, Humanoid robot simulation and walking behaviour development in the Spark simulator framework. Technical report, Artificial Intelligence Research University of Koblenz, 2005.

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

[14] L. Yang, C. M. Chew, T. Zielinska and A. N. Poo, "A uniform bipedal gait generator with offline optimization and online adjustable parameters", Robotica, Cambridge University Press, vol. 25, 2007, pp. 549-565.

[15] P. K. Schot and M. J. Decker, "The force driven harmonic oscillator model accurately predicts the preferred stride frequency for backward walking", Human movement science, vol. 17, 1998, pp. 67-76.

[16] D. E. Goldberg, "Genetic algorithms in search, optimization and machine learning", Addison-Wesley, 1989.

[17] M. Melanie, "An introduction to genetic algorithms", A Bradford book the MIT press, Cambridge, London, England, 1999.