2010 11th Int. Conf. Control, Automation, Robotics and Vision
Singapore, 7-10th December 2010

A Learning Approach to Optimize Walking Cycle of a Passivity-based Biped Robot

Nima Fatehi
Mechatronic Group,
Electrical and Computer Engineering Faculty,
Islamic Azad University,
Qazvin Branch
Qazvin, IRAN
Email:
neamadeus@gmail.com

Masoud Asadpour
Control and Intelligent Processing Center of Excellence,
Electrical Engineering Faculty, University of Tehran
Tehran, IRAN
Email: asadpour@ut.ac.ir

Adel Akbarimajd
ECE Group, Faculty of Engineering,
University of Mohaghegh Ardabili
Ardabil, IRAN
Email:
akbarimajd@uma.ac.ir

Laila Majdi
Mechanical Department
Islamic Azad University,
South Tehran Branch
Tehran, IRAN
Email:
l.majdi@gmail.com

***Abstract*— A learning mechanism based on Powell's optimization algorithm is proposed to optimize walking behavior of a passivity based biped robot. To this end, a passivity-based biped robot has been simulated in MSC ADAMS and a control policy inspired from humanoid walking is adopted for a stable walking of the robot. Linear controllers try to control the joints of robot in each walking phase to implement the gait proposed by the control policy. Learning is employed using Powell's optimization algorithm to adjust the control parameters so that the robot enters to an optimum limit cycle in a finite time. The fitness function is defined to evaluate the robot's optimum behavior. The results are verified by simulations in SIMULINK+ADAMS.**

***Keywords*— Biped Robot, robot simulation, Passivity based walking, Powell's method, Optimization, fitness function.**

I. INTRODUCTION

In the recent decades there have been many interest and studies around biologically inspired locomotion systems and especially biped walking. Different approaches have been employed to develop and control biped robots including ZMP based approaches [1], passivity based walking [2,3,4] and trajectory tracking approaches [5,6]. The main challenge has been to find a best walking strategy which would be energy efficient, stable and adaptive to handle different terrain and disturbances; the features found mostly in the human walking. Due to complexity and high DOF of humanoid robots, there may be many possible gaits, but the question is how to reach the best answer. In animals, independent of the mechanical system, the best solution is learned through experience.

Reinforcement learning has been used by different approaches to learn some issues in biped walking [5,7]. Neural networks also have been used by many researchers to learn the actuation of joints [8]. Sprowitz et. Al used central pattern generators and Powell's optimization method to show that a robot with an arbitrary arrangement of Yamor modules can learn to move independent of mechanical configurations [9].

Linear controllers are simple and easy to design. They are widely used for industrial and research purposes. Their main drawback is their limitation in controlling nonlinear systems, according to the fact that most physical systems have nonlinear characteristics. However some researchers have used linear controllers in nonlinear problems using a model linearized over multiple state ranges called LOLIMOT [10].

In this research we have used such idea to divide state space into some sections in each of which a linear feedback controller would be responsible to control a joint for the task defined in the gait control policy. The Powell's optimization algorithm then helps to learn the best controller parameters leading to a stable walking with the best possible energy efficiency.

II. THE ROBOT MODEL

The robot is simulated as a planar 2D biped with 6 joints and 7 links in MSC ADAMS. Fig. 1 shows the developed model. We tried to make our model as close as possible to passive walking systems so that it could use the natural dynamics for walking instead of actuating every joint. This has the advantage of minimizing the energy consumption and would lead to a behavior similar to human walking. The knee joints thus have been considered without any actuators and a kneecap prevents them from reverse bending. On the hip and ankle joints actuators have been used to help controlling the motion of the robot on level ground. It is assumed that these actuators are not interlocked to the joints so that they would not bypass the passive dynamics of robot and their torques are supposed to be added to the inertial and gravity forces imposed to the links. Most passivity-based biped robots use curved feet to help them reduce the energy loss of heel strike and eliminate need for an active ankle joint, but for getting closer to a human walking and for analysis of the role of heel

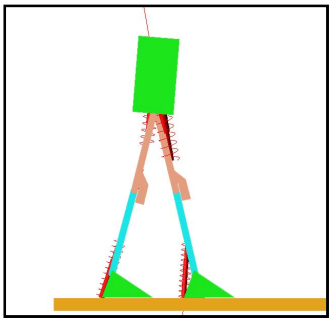


Figure 1. MSC Adams implementation of the simulated robot

978-1-4244-7815-6/10/$26.00 ©2010 IEEE

and toe in walking we have considered flat feet for the robot. Employing springs around ankle joint, as also done in [2,4] helps for storage and reuse of energy through walking just like what Achill's tendon does in human body. It also has the advantage of preventing feet from over-bending. In the same way for preventing the torso from over-bending around hip, springs are also used between torso and thigh links. Table I shows mass, inertia and geometric properties of links designed for the simulated robot. Using a relatively heavy torso helps for stability of walking by increasing the inertia of stance foot's inverted pendulum thus giving more time for swing leg to reach the gait angle and stretch the knee. The heavy mass of torso also could be employed to keep robot's balance by bending the torso forward in the situations when robot's hip is placed before the stance foot. Selecting a very light shank relative to thigh in passive bipeds would help stretching of the knee at the end of swing phase [11]. This however is not the case in the human body. We therefore have used a more realistic ratio for the link masses while swing leg's hip controller will be responsible for stretching the knee.

At the beginning of the simulation in order to make the test start from a non-stationary situation, an initial velocity of 0.75 (ms^{-1}) has been imposed to torso in x direction and the control system discussed in next section has to take the robot from this initial condition to a stable walking limit cycle. For the simulations a sample time of $T_s = 3\ ms$ (the communication interval between controller and robot simulator) and a total simulation time of $T_e = 5.5\ s$ have been used. Due to very slow simulation of the robot in ADAMS (with an AMD 3.11 GHz processor and 3GB RAM one second of simulation time would take about 30 seconds to run), less sample times takes very long learning time while greater sample times could lead to instability according to numerical errors.

III. CONTROL SYSTEM

The steady state stable motions in the walking robot are periodic sequences called limit cycle. This limit cycle can be divided into some subsections in terms of discrete states, in each of which a fixed control policy could be defined. The whole limit cycle for a biped is composed of two main symmetric phases; in one phase left leg accepts the support role and right leg swings and in the second phase the roles are exchanged between the legs. The discrete states are defined through some specific events for each of the support and swing legs and a control policy is assigned for each of them.

TABLE I. MASS AND INERTIA PROPERTIES OF LINKS

Link	Mass	Dimensions	Moment of Inertia
Foot	0.7 Kg	150x70 mm	3.289 x 10^{-3} kg.m2
Shank	1 Kg	250 mm	34.286 x 10^{-3} kg.m2
Thigh	2.45 Kg	250 mm	34.286 x 10^{-3} kg.m2
Torso	5 Kg	120x220 mm	200.305 x 10^{-3} kg.m2

Fig. 2 shows a diagram of discrete states considered in the limit cycle for this research. The specific events used to distinguish different states are as follows:

- Support Exchange: This event is determined by comparison of the foot-ground contact forces. The foot with greater contact force with ground will be considered as the support foot. We have used a simple first order filter for foot contact forces to eliminate the effect of impulses at heel strike on the comparison.
- Support Ready: When the support foot hits ground from heel point, it is not still ready to accept the weight of robot until the foot is placed on the ground horizontally and the leg is also near to a vertical position. When these two conditions are met, then the leg is considered ready to support the weight.
- Knee Stretched: When the knee angle for the swing leg is reached zero, it is stretched and can accept support for the next step.
- Robot Falling: If height of torso COM is reduced below a threshold, it is assumed that the robot is falling. In addition, while learning is in progress, some large torques might be generated in joints, causing the robot to jump off the ground. Since this is not an accepted state in the walking limit cycle, we will consider it also as falling.

Using the above conditions the state of robot could be segregated and in each state for each joint a specific control policy can be defined. The control policy for joints could be summarized as follows:

- The support ankle, as an inverted pendulum, should keep the leg vertical so that the swing leg has enough distance from ground for foot clearance. This is possible if the Support Ready conditions are met. This policy will continue until the swing leg reaches the gait angle and its knee is stretched so that it could accept support for next step. Assuming a linear model of inverted pendulum around zero angle (vertical point), a state feedback of this inverted pendulum, consisting of angle and angular speed, is used as:

$$\tau_{ankle}^{support} = k_{a1}(0 - \theta_{ankle}) + k_{\omega 1}(0 - \omega_{ankle}) \qquad (1)$$

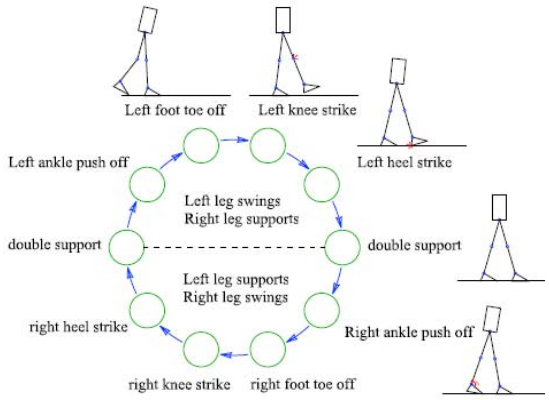


Figure 2. Important events in the limit cycle of walking biped

- The swing ankle should keep the foot parallel to ground surface for maximum foot clearance.

$$\tau_{ankle}^{swing} = k_{p1}(0 - \theta_{foot}) + \frac{k_{i1}}{s}(0 - \theta_{foot}) \quad (2)$$

- After the "Support Exchange" and before "Support Ready", the previously support foot's ankle could drive the robot forwards by pushing the toe to ground, until the support leg is vertical. Then the inverted pendulum could be controlled around this point by new support foot's ankle.

$$\tau_{ankle}^{PushOff} = k_{po}(\theta_{ankle}^{support}) \quad (3)$$

- The swing hip shall control the thigh to swing from its initial position to the desired gait angle while providing a suitable acceleration in order to make the knee bend during swing phase, and stretch at the end of it. A position PI controller is responsible for gait angle, while a proportional speed controller accelerates the thigh to a desired speed while swinging, and decelerates it toward zero when the gait angle is reached.

$$\tau_{hip}^{swing} = \begin{cases} \left(k_{p2} + \frac{k_{i2}}{s}\right)(\theta_{sp2} - \theta_{hip}) + k_{\omega 2}(\omega_{sp2} - \omega_{hip}) \\ if\ \ \theta_{hip} < \theta_{sp2} \\ \left(k_{p2} + \frac{k_{i2}}{s}\right)(\theta_{sp2} - \theta_{hip}) + k_{\omega 2}(0 - \omega_{hip}) \\ if\ \theta_{hip} \geq \theta_{sp2} \end{cases} \quad (4)$$

- Regarding that the stance foot tries to keep the hip in a vertical position relative to support ankle and assuming that this point has very little changes during single support phase, the stance hip then can set the torso angle. The torso might be considered to be kept upright, but learning results show that leaning the torso forwards would help robot in walking. A PI controller can keep the hip at the desired angle.

$$\tau_{hip}^{support} = \left(k_{p3} + \frac{k_{i3}}{s}\right)(\theta_{sp3} - \theta_{hip}) \quad (5)$$

- If the robot is falling the simulation will be terminated.

Parameters of above controllers should be tuned so that an optimum stable walking cycle is achieved. In next section, the learning and optimization algorithm is provided for this issue.

IV. Learning and Optimization

Learning in intelligent systems is done by correcting system behavior for reaching an end. Since the simulations take so long to run, we need a method which would converge very fast to the optimum. The most advantageous case would be a robot capable to learn walking itself, but since the learning task of whole walking without any teacher and any previously known policy could be very complicated, time consuming and probably impossible, we have fixed the control policy to what described in section III and made the robot learn the controller parameters. Since we do not have any known desired patterns to compare robot's behavior with and we want the robot to search for the best possible answer, we cannot use gradient based methods. In this research we have used Powell's optimization method for the controller parameters to learn the best possible value. Powell's method is fast and gradient free, but the drawback is that it could easily converge to local optimums.

A. Powell's Method

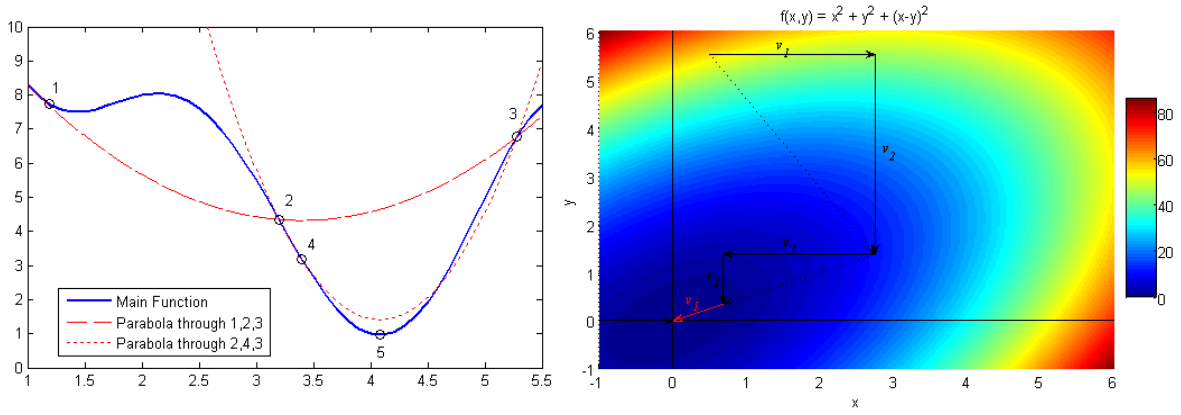
Powell's method uses Brent's one dimensional optimization in one direction of the learning variables space, and then employs a method for changing this direction so that after sequences of optimization along these directions the optimum of a function is reached. In this research a fitness function F_t shall be maximized so that the best behavior for the walking robot would be reached. Since Powell's method is typically a minimization method, we will consider $f = -F_t$ to be minimized by this method.

Brent's one dimensional optimization brackets a guessed minimum, x_1, with two other points, x_0 and x_2. The points should be selected such that $x_0 < x_1 < x_2, f(x_1) < f(x_0)$ and $f(x_1) < f(x_2)$. If f is non-singular, it must have a minimum between x_0 and x_2. We will select x_1 as the initial point of optimization, and will find x_0 and x_2 around this point by a deviation, $dx = d.x_1$ where $d < 1$. Searching algorithm continues until the starting conditions are satisfied. Having the three bracketing points, a quadratic equation in the form

$$g(x) = ax^2 + bx + c,$$

is assigned to them. By comparison of fitness function at the minimum of this equation, $f\left(\frac{-b}{2a}\right)$, and at the middle point, $f(x_1)$, the point with less function value will be chosen as x_1 and the two nearest points around it will be considered as x_0 and x_2 in the next iteration. Fig. 3-a illustrates this one dimensional optimization. In this way the bracketing range will be reduced towards minimum and the search method will continue until the outer bracketing range is less than a threshold.

Powell's method starts with unit vectors of the N dimensional learning variables space as the set of directions. In each iteration of the algorithm N one-dimensional optimizations along the N directions in the set are done. At the end of an iteration it is checked to see if another optimization of f along the direction of net optimizations in the iteration is beneficial. Namely if the starting point is P_0, then we will move from this point to P_1 by minimization along first direction in the set, the same will continue for other optimization along other directions until we get to P_N. If we consider $f_0 = f(P_0)$ and $f_N = f(P_N)$, then $f_E = f(2P_N - P_0)$ represents the function value if we move along the net optimization direction once again. If this reduces the function value, i.e. $f_E < f_0$ it means that the net optimization direction is towards minimum of the function, then considering some conditions one of the directions in the set is replaced by $P_N - P_0$ [12]. In Fig. 3-b the replacement of the horizontal vector with the net optimization vector has happened from 3rd iteration. The algorithm continues until the difference of function value between two successive iterations is less than a threshold.



(a) (b)

Figure 3. Illustration of optimization algorithm ; a) Brent's one dimensional optimization using parabola assignment b) Direction setting with Powell's method for function $f(x,y) = x^2 + y^2 + (x-y)^2$

B. *Learning Variables*

We have chosen the controller parameters illustrated in Table II as the learning variables. They make a 13-dimensional space where every point corresponds to a special behavior of robot.

The learning variables of Table II can be divided into three groups; first group are setpoints of the controllers, which are θ_{sp2}, ω_{sp2} and θ_{sp3}. Second group are coefficients of hip controllers with setpoints pointed out in first group; these coefficients are $k_{p2}, k_{i2}, k_{\omega 2}, k_{p3}$ and k_{i3}. In these controllers, it is desirable if the error is reduced to zero. The third group consists of coefficients of ankle controllers. In these controllers it is not necessary to reach a zero error. For example the swinging foot's ankle cannot always keep the foot parallel to ground as the joint workspace is constrained; in fact this control task is only vital when the foot is approaching the ground. The same is for support ankle in which a small rotation around vertical position (setpoint) is needed for stepping. This classification will be used later when sequence of learning variables is set.

TABLE II. CONTROLLER PARAMETERS USED AS LEARNING VARIABLES

Variable	used in controller	Duty
k_{p1}	Ankle – Swing	Swing foot's clearance
k_{i1}	Ankle – Swing	Error removal of foot clearance
k_{a1}	Ankle – Support	Keeping support leg vertical
$k_{\omega 1}$	Ankle – Support	Preventing support leg's motion
$k_{\omega 2}$	Hip - Swing	Accelerating the swing leg
k_{p2}	Hip – Swing	Reaching swing leg to gait angle
k_{i2}	Hip – Swing	Error removal for gait angle
θ_{sp2}	Hip – Swing	Gait angle setpoint
ω_{sp2}	Hip – Swing	Swinging speed setpoint
k_{p3}	Hip – Support	Keeping torso in a proper azimuth
k_{i3}	Hip – Support	Error removal for torso azimuth
θ_{sp3}	Hip – Support	Torso azimuth setpoint
k_{po}	Ankle – Push off	Ankle push off to ground

The initial point from which optimization starts is very important as Powell's method is likely to converge to local optimums if started from an improper point. We have manually tuned the controller parameters by trial and error so that the robot would walk two primary steps, and these parameters are used as the initial point. The walking then improves as a result of optimization progress.

C. *Fitness function*

Fitness function is the criterion according to which the behavior of the system (the biped robot here) is evaluated. Therefore, it should include criteria about all the important walking properties. We have considered four criteria functions, each being related to a specific issue about robot's walking, and the main fitness function is calculated by a weighted sum of these criteria functions as:

$$F_t = w_l F_l + w_b F_b + w_e F_e + w_c F_c \qquad (6)$$

where F_l, F_b, F_e and F_c are criterion functions related to walked length, balance keeping, energy consumption and controller optimization issues respectively and the corresponding w's represent the assigned weights of each. The criterion functions should be normalized so that their values would be comparable to each other. The four criteria functions are described as follows.

1) Walked Length: If the robot can walk the whole simulation time, T_e, the walked length represents walking speed, but if it falls before this time simulation ends and walked length will be limited to the falling point. Falling of the robot may occur toward front or rear and this shall not be considered as a positive or negative point; so we have used the position of the stance leg's ankle as the length criterion. The normalized function, F_L, is thus calculated as:

$$F_L = \frac{x_{ankle}^{support}}{x_{max}} \qquad (7)$$

where x_{max} is the maximum expected distance that robot could travel during the simulation time.

2) Balance Keeping: While the robot is not fallen it is considered to be successful keeping its balance. If we denote the balance time from start until the end of simulation with T_b (whether by falling of robot or reaching the final simulation time), then the normalized balance function F_b can simply be written as:

$$F_b = \frac{T_b}{T_e} \tag{8}$$

3) Energy Consumption: It is desired to have a walking system with minimum energy consumption. We have calculated the consumed energy as the mechanical work;

$$W = \sum_i W_i = \sum_i (\int |\tau_i d\theta_i|) \tag{9}$$

where τ_i is the torque exerted at joint i and $d\theta_i$ is the differential joint movement. For normalization of this function we need a maximum expectation of energy consumption, E_{max} which will be estimated as:

$$E_{max} = E_{max}^{step}.n \tag{10}$$

in which n is the maximum expected steps number, and E_{max}^{step} is the energy consumed in an unstable energy consuming step in which the robot is falling.

One problem with energy consuming criterion is that by moving forward more energy is consumed, so falling in the first steps might be preferred to moving. For overcoming this problem, we have used consumption power instead of energy. The criterion function, F_e, thus will be calculated as:

$$F_e = 1 - \frac{P_{sim}}{P_{max}} = 1 - \frac{\frac{E_{sim}}{T_b}}{\frac{E_{max}}{T_e}} = 1 - \frac{E_{sim}.T_e}{E_{max}.T_b} \tag{11}$$

in which E_{sim} is the energy consumed in simulation time, presented by mechanical work, W.

4) Controller Errors: In the walking learning process, a joint may need to reach a specific angle for best performance of robot. We want the controllers of these joints to track the setpoints and minimize the control error. As discussed in learning variables, hip controllers are of such interest. We have used the IAE (integral of absolute error) as another criterion function so that besides the optimization of robot's behavior the controllers would be optimized as well. For normalizing the error we have divided it to the setpoint value. The normalized error, ne, can be found as:

$$e_{hip}^{swing} = \theta_{sp2} - \theta_{hip} \Rightarrow ne_{hip}^{swing} = 1 - \frac{\theta_{hip}}{\theta_{sp2}} \tag{12}$$

$$e_{hip}^{support} = \theta_{sp3} - \theta_{Torso} \Rightarrow ne_{hip}^{support} = 1 - \frac{\theta_{Torso}}{\theta_{sp3}} \tag{13}$$

and the IAE could be calculated accordingly as:

$$IAE_{h1} = \int \left(\left| ne_{hip_L}^{swing} \right| + \left| ne_{hip_R}^{swing} \right| \right) dt \tag{14}$$

$$IAE_{h2} = \int \left(\left| ne_{hip_L}^{support} \right| + \left| ne_{hip_R}^{support} \right| \right) dt \tag{15}$$

where the subscripts L and R stand for left and right joints. Just like what said about energy consumption, we want this criterion to be measured over time, hence we will include T_b and finally the criterion function, F_c, will be:

$$F_c = 1 - \frac{1}{2T_b}(IAE_{h1} + IAE_{h2}) \tag{16}$$

V. Simulation And Discussion

The simulation results show that for the mechanical system and control policy we defined, a walking solution exists that can be optimized according to the fitness function defined by (6). Starting from the initial point P_0 of learning variables, the walking continues only for 2 steps and robot falls after 1.5 sec. The optimization however gradually improves the performance of walking until a stable walking is achieved and then optimized.

There are some issues that affect the learning process with Powell's optimization method and shall be considered carefully. The first issue, which was discussed previously, is the starting point. We tested the problem with different random initial conditions. The results show starting from these points will not lead to a stable walking and the robot whether falls before the end of simulation or tends to stand up and keep its balance until the end of simulation time. The robot must be able to walk some few steps before the learning starts so that during learning, experiments which lead to falling would have less fitness than the starting fitness value and thus rejected. Therefore for a successful optimization we tuned the starting point manually such that the robot would not fall for two steps.

Other issue is the sequence of learning variables to be optimized. In a problem like biped walking with the fitness function defined previously there exist many local optimums. Optimization with an inappropriate sequence of parameters may bring us to a local optimum which cannot be escaped. So we needed to find a proper sequence for parameters. As mentioned previously we can categorize the learning variables into three main groups; 1) Hip controller setpoints, 2) Hip controller coefficients and 3) Ankle controller coefficients. It is necessary that controllers would be optimized prior to the setpoints, so we have made the sequence as third group, second group and then first group. For setting the sequence among the variables in a group we have sorted them according to the amount of increase in fitness function by optimization of each one. Finally the best sequence was found as:

$[k_{p1}\ k_{a1}\ k_{i1}\ k_{\omega 1}\ k_{po}\ k_{\omega 2}\ k_{p2}\ k_{i2}\ k_{p3}\ k_{i3}\ \theta_{sp3}\ \theta_{sp2}\ \omega_{sp2}]$.

The remaining parameters to be set are the fitness weights vector, $W = [w_l \quad w_b \quad w_e \quad w_c]$. We optimized the walking with different weights to find the priority of criterion functions according to the walking results. We expect the robot to walk

continuously for the whole simulation time after optimization is done. This requires the balance criterion function to be 1, and the walked length criterion function to be as much as possible. Firstly we tested the optimization with $W = [1\ 1\ 1\ 1]$, $W = [2\ 1\ 1\ 1]$, $W = [1\ 2\ 1\ 1]$, $W = [1\ 1\ 2\ 1]$ and $W = [1\ 1\ 1\ 2]$. Fig. 4 compares the criterion functions for these weight vectors. When equal weights are used, criterion functions would cancel each other, so priorities are needed among them. As Fig. 4 shows, just giving priority to walked length criterion could make the robot walk forwards. Setting priority for energy or controller error could lower energy consumption and controller errors but would not lead to a stable walking. Secondly we tested the optimization using $W = [3\ 2\ 1\ 1]$, $W = [3\ 1\ 2\ 1]$ and $W = [3\ 1\ 1\ 2]$ to find the second priority for criterion functions. As Fig. 5 shows, this time $W = [3\ 2\ 1\ 1]$ showed best performance. Since we were satisfied with the walking behavior of robot this weighting was considered finally.

After the optimization robot was able to walk continuously with 6 steps in 5.5 seconds of simulation. Fig. 6 shows some frames of robot's walking. The proposed optimization method with the defined fitness function thus shows to be successful for optimization of this passivity based robot's walking.

REFERENCES

[1] P. Sardain and G. Bessonnet: "Forces Acting on a Biped Robot. Center of Pressure—Zero Moment Point", In: *IEEE Transactions on Systems, Man, and Cybernetics, Vol. 34, No. 5, Sep. 2004*

[2] M. Wisse, G. Feliksdal, J. V. Frankenhuyzen, B. Moyer: "Passivity Based Walking Robot; Denise, a simple, Efficient and Lightweight Biped", In: *IEEE Robotics & Automation Magazin, Vol. 14, No.2 June 2007*

[3] Tad McGeer: "Passive Dynamic Walking", In: *International Journal of Robotics Research, PP: 62-82, 1990.*

[4] J. E. Pratt and G. A. Pratt: "Exploiting Natural Dynamics in the Control of a Planar Bipedal Walking Robot" In: *Proceedings of the 36th Annual Allerton Conference on Communication, Control, and Computing, Sep. 1998*

[5] H. Benbrahim, J. A. Franklin: "Biped Dynamic Walking Using Reinforcement Learning", In: *Robotics and Autonomous systems 22 (1997)*

[6] L. Righetti and A. J. Ijspeert: "Programmable Central Pattern Generators: an application to biped locomotion control", In: *Proceeding of the 2006 IEEE International Conference on Robotics and Automation*

[7] J. Morimoto, C. G. Atkeson: "Learning Biped Locomotion; Application of Poincare-Map-Based Reinforcement Learning", In: *IEEE Robotics & Automation Magazin, Vol. 14, No.2 June 2007*

[8] A. L. Kun, W. T. Miller: "Adaptive Dynamic Balance of a Biped Robot Using Neural Networks" In: *Proceedings of international conference on Robotics and Automation, IEEE, 1996*

[9] A. Sproewitz, R. Moeckel, J. Maye, A. J. Ijspeert: "Learning to move in modular robots using central pattern generators and online optimization" In: *International Journal of Robotics Research, Vol. 27 , March 2008*

[10] H. Rouhani, M. Jalili, B. N. Araabi, W. Eppler, C. Lucas: "Brain emotional learning based intelligent controller applied to neurofuzzy model of micro-heat exchanger", in: *Expert System with Applications*

[11] V. F. Hsu Chen: "Passive Dynamic Walking With Knees: A Point Foot Model", *Thesis for Master Of Engineering in Electrical Engineering and Computer Science, MIT University, February 2007*

[12] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery: "Numerical Recipes in C, The Art of Scientific Computing", *Copyright 1988-1992 by Cambridge University Press.*

[13] Karl Astrom, Tore Hagglund: "PID Controllers", *2nd Edition, Copyright Instrument Society of America 1995*

[14] A. D. Kuo: "Choosing Your Steps Carefully; Trade-Offs Between Economy and Versatility in Dynamic Walking Bipedal Robots", In: *IEEE Robotics & Automation Magazin, Vol. 14, No.2 June 2007*

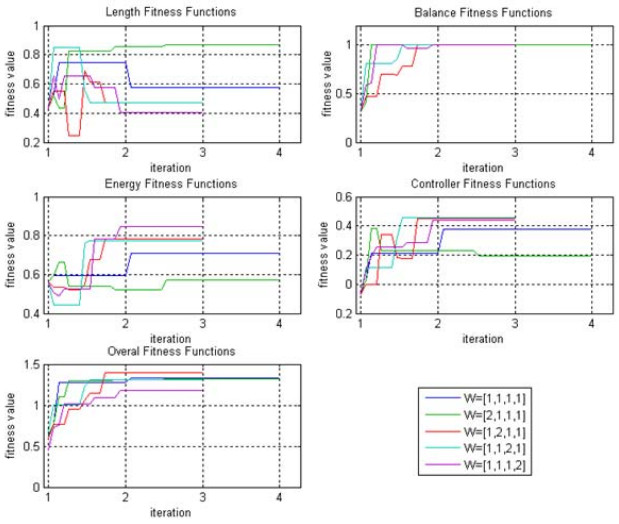


Figure 4. Criterion and fitness functions with different weight priorities

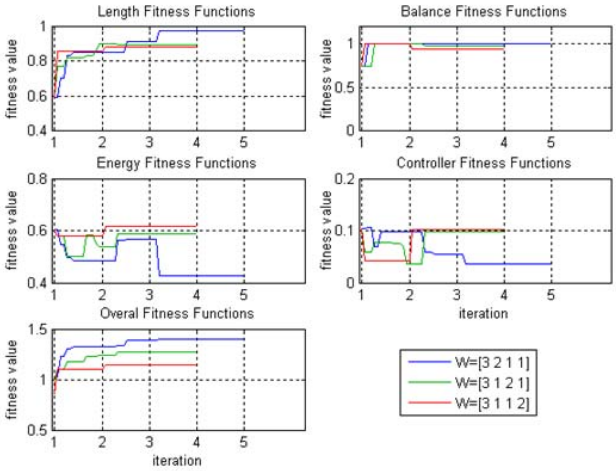


Figure 5. Optimization with $w_l = 3$ and $w = 2$ for other criterion functions

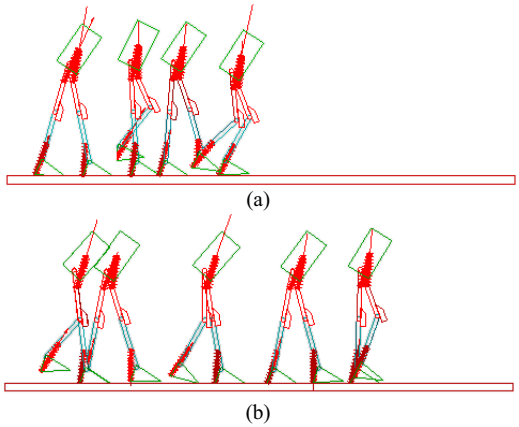


Figure 6. Robot simulation (a) falling before optimization (b) continuous walking after optimization