Neurocomputing 233 (2017) 72–80

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Efficient mechanical design and limit cycle stability for a humanoid robot: An application of genetic algorithms

Rafael Stanley Núñez Cruz*, Juan Manuel Ibarra Zannatha

Department of Automatic Control, CINVESTAV, México, Mexico

A R T I C L E I N F O

Keywords:
Optimal Design
Passive Dynamic Walkers
Genetic Algorithms
Optimal Feedback Control

A B S T R A C T

In this paper the application of Genetic Algorithms is presented to the task of designing a humanoid robot able to exhibit an efficient walking. This task is presented as an optimization problem. The objective function is the so-called Specific Cost of Transportation and the restrictions of the problem are based on the Limit Cycle Walking stability criterion. The mechanical design of the prototype and its walking trajectories are inspired on passive dynamic walkers. A basic genetic algorithm was used to find: the optimized mechanical parameters for design, the walking trajectories and the feedback gains used in the control of the current prototype.

1. Introduction

The present work shows the continuity that our group has given to the design of humanoid robots that began with the development of the prototype *AH1N1* [1]. Now we are focusing our approach on Energy efficiency which is difficult to obtain on humanoid walking robots, we decide to face this task as an optimization problem [2].

According to the Limit Cycle Walking paradigm [3], in order to find more efficient, natural, fast and robust walking motions it is necessary to reduce the artificial constraints added when using other stability criteria usually based on Zero Moment Point [4].

The most generic definition of walking stability, without artificial constraints, is ”to avoid falling”. The use of this definition implies to evaluate all possible walking trajectories and evaluate if the walker falls or not. In order to use this definition in a practical manner we propose the use of Genetic Algorithms by considering the task as an optimization problem.

Since the analysis of Passive Dynamic Walkers [5] follows Limit Cycle Stability we decide to study these machines as a base for the mechanical design of our humanoid robot platform. In Fig. 1 are presented some of the main architectures considered.

The original methodology used to built actuated robots based on passive dynamic walkers [6] consists on the addition of actuation and control systems to the passive version. Since passive walkers move on a downhill slope, the actuators emulate the effect of the gravity on the slope so the actuated mechanism can move on a flat surface. This transition can modify the original mechanical design in such a way that the efficiency of the passive version is mitigated on this transition.

This is the reason why we decided to change the approach by starting with an actuated walker, which uses the principles of passive dynamic walkers and find out which are the parameters that can be optimized to obtain energy efficiency. We study the next different approaches as a point of departure:

Frequency of the system: The oscillatory nature of walking is so evident that is reasonable to attempt to use some of the techniques that show good results on oscillatory systems. One of this methods [7] propose the addition of torsional springs on each joint of the system, the optimal stiffness for each joint is found by using an adjustment law, the idea of this method is to match the natural frequency of the system and the one of the reference trajectory.

Amplitude and frequency of the step: Other method [8] propose to find the optimal amplitude and frequency of the reference trajectory in order to exploit the inherent cyclic characteristics of the system.

Control law: Based on the Optimal Control Theory it is possible to express energy consumption in terms of the feedback control gains [9], however a realistic mathematical walking model is difficult to optimize by using conventional optimization techniques.

The main contribution of this paper is the use of Genetic Algorithms to put together all these partial points of view to define a more general optimization problem where mechanical parameters, walking trajectories and law control can be optimized and by using Limit Cycle Walking stability criterion the solutions are restricted to stable cycles.

In the area of robotics, Genetic Algorithms have already been used in other specific problems such as the design of optimal gaits [10] or to select mechanical parameters and proper morphology [11].

In the remainder of this paper we present our approach on the design of a humanoid robot called *Johnny*. Some of the characteristics of the passive dynamic walkers we used to design our own robot are

* Corresponding author.
E-mail addresses: rnunez@ctrl.cinvestav.mx (R.S. Núñez Cruz), jibarra@cinvestav.mx (J.M. Ibarra Zannatha).

http://dx.doi.org/10.1016/j.neucom.2016.08.113
Received 16 December 2015; Received in revised form 16 June 2016; Accepted 1 August 2016
Available online 19 November 2016
0925-2312/

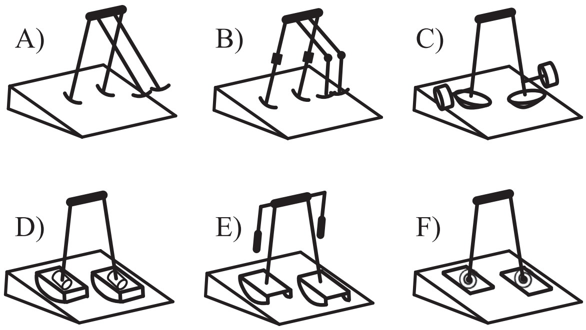

Fig. 1. Passive Dynamic Walkers previously studied.

described in Section 2. The mathematical model of the gait and definition of stability are discussed in Section 3. The gate design and its conversion to joint trajectories is presented in Section 4. The optimization problem, including cost function, restrictions and search space is formulated on Section 5. The characteristics of the Genetic Algorithm selected as optimization method are described in Section 6. The results of the optimization algorithm are presented along with the characteristics of the prototype built in Section 7. We conclude with a summary and future work in Section 8.

2. Passive dynamic walkers

A passive dynamic walker is a two legged machine designed to walk stably. This kind of walker has no actuators neither control systems, the movement of this machine is mainly a phenomena produced by the effect of the gravity on its limbs and the equilibrium between potential and kinetic energy. Passive dynamic walkers exhibit a stable gait when put on downhill slope and proper initial conditions of position and velocity are set.

McGeer began the study of passive dynamic walkers by using a mathematical approach, he built a straight legged prototype with rounded feet, whose movement was restricted to the sagittal plane, which demonstrates stable walking with no control and no actuators, Fig. 1A. This walker uses a mechanism to slightly retract the swing foot to avoid scuffing.

The methodology used to built this first prototype [5] consists in finding proper initial conditions of position and velocity that produce a limit cycle, for a given set of physical parameters of the used gait model, in case that no limit cycle can be found, a new set of physical parameters must be defined in order to continue the search, this can be done by modifying slightly some physical dimension, i.e. the length of the legs, the radius of the feet, mass distribution, etc.

Most of the time, is possible to find more than one set of parameters that produce a stable walking, each one of this possible solutions will produce a corresponding actuated mechanism that requires, in general, a different amount of energy to move on an horizontal surface.

After the original prototype some researchers have studied the addition of other elements:

McGeer [12] designed a second version of his original passive walker by adding knees to the legs (Fig. 1B), he also added a mechanism to lock and release the knees depending if it is the stance or the swing leg. With this addition, the machine was able to exhibit a more human-like movement and avoid scuffing while keeping low actuation.

Coleman and Ruina [13] built a rounded feet walker which allows lateral movement, this walker also have stable bars attached to the legs to increase stability, Fig. 1C. This straight leg walker was able to avoid scuffing without additional mechanisms although it can only move in small steps.

Wisse [14] designed a model with cylindrical feet and articulated ankles, Fig. 1D, showing that coupling Lean and Yaw movements can lead in to stable 3D passive movements above a minimum forward velocity just as bicycles.

Fig. 2. Passive Dynamic Walkers previously developed.

Collins [15] developed a sophisticated prototype that mechanically couples the movement of the legs to balance a torso and moving arms, Fig. 1E, this prototype exhibit a very stable and human-like walking with efficient use of the actuation provided.

Narukawa [16] used springs on the ankles of his flat feet version, Fig. 1F, this addition allows to modify the frequency of the gait in an efficient way, however a bad selection of the springs stiffness can lead in to undesired behaviors as foot oscillations or rebounds at contact.

The previous approaches were studied and compared by simulations and by building our own prototypes, these passive versions can be seen in Fig. 2.

Based on the performance obtained in these passive prototypes it was decided to include the next characteristics on our actuated platform:

- Additional masses on the limbs which form the legs in order to obtain an specific center of mass position.
- Rotational springs on each of the joints which form the legs in order to obtain an specific frequency of the mechanism.
- Flat feet and articulated ankle in order to apply control torques.
- Torso and swings arms to compensate rotational movements.

The mechanical architecture of *Johnny* consist in 22 actuator used as follows: 5 dof for each leg, 2 dof for the chest, 3 dof for each arm, 1 dof for each hand and 2 dof for the neck.

3. Mathematical model

Gait is modeled as an interconnection of continuous stages with discrete events, Fig. 3 shows the five stages of walking used in the model, the final state vector on each stage is used as initial conditions for the next one:

(A) *Swing phase*: When the robot is supported only on one foot, its movement is governed by a system of nonlinear equations, we consider three open kinematic chains whose origin coordinate frame are connected to the foot in the ground (one kinematic chain for each arm and another one for the swing leg).

The mathematical model, during this phase, can be obtained using Euler–Lagrange equation or Newton–Euler formulation, for simplicity we will consider no changes in the walking direction. The complete kinematic chains shown in Fig. 4 has 22 joints however in the mathematical model we will not consider the joints for the head and hands neither the joints used to change walking direction, so the mathematical model considers joints from q_0 to q_{15}.

(B) *Locked knee*: In the case of full passive walkers, knees have mechanical restrictions to avoid hypertension, when the knee is full extended a mechanism blocks its movement and keeps the swing leg straight.

This collision produce a discontinuity on joint's velocity. The

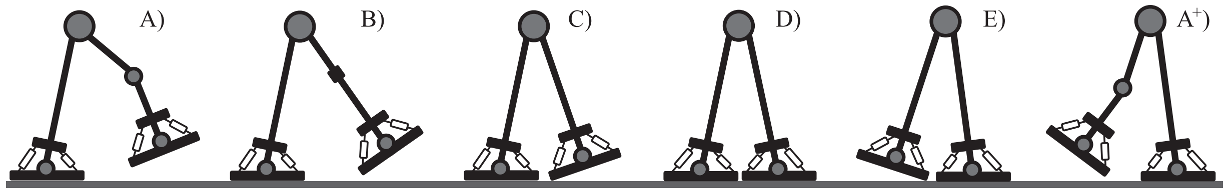

Fig. 3. Walking stages: (A) Swing phase (B) Locked knee (C) Heel Strike (D) Double support phase (E) Rear foot push up.

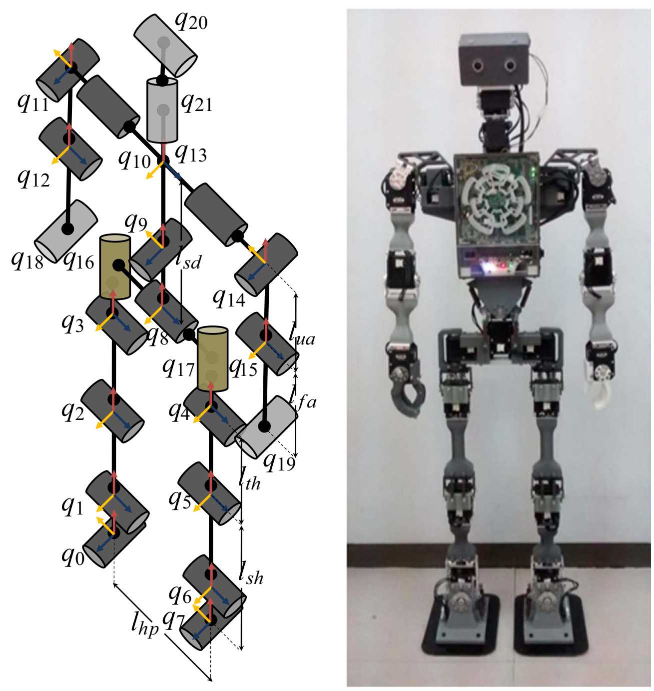

Fig. 4. *Johnny*'s Kinematic chain.

velocities after the impact are calculated by using the conservation of angular momentum.

After the knee is blocked the kinematic chain of the leg loses one joint and the system continues its movement with the new velocities.

(C) *Heel strike*: In the case of flat feet robots, the swing foot touches the ground by the heel before it makes full contact with the ground, however usually this phenomena is not modeled, but in the case the robot have springs on ankle's joint it is necessary to model the conversion from kinetic to potential energy, which occurs when these springs get compressed as the swing foot makes full contact with the ground.

(D) *Double support phase*: When the swing foot makes full contact with the ground another collision occurs and the support point is transferred to the opposite foot. This events is considered instantaneous so the position of the robot is the same after the impact and only velocities are changed. This collision is also modeled by using the conservation of angular momentum.

After this collision the equations of motion must be rewritten considering the origin at the ankle of the new support foot.

(E) *Rear foot push up*: As the rear foot leaves the ground the ankle's springs release the energy store pushing the robot forward, this effect is model as a conversion of potential to kinetic energy. This is the opposite phenomena of stage C.

These five stages constitute a complete walking cycle, the final conditions of the current cycle will be the initial conditions of the next one.

In most of the cases it is difficult to represent this model as a step-to-step transition, in a closed form, as shown in Eq. (1):

$$x^{k+1} = S(x^k) \tag{1}$$

Here $x^k = [q^k \dot{q}^k]$ represents the state vector form by the angular positions and velocities of each joint at the beginning of step k. This equation is known as *Stride Function* when it is related with a walker model.

Usually the way to evaluate Eq. (1), to obtain the trajectories of the robot, given some initial conditions x^0, consists in using numerical integrations of the corresponding nonlinear system depending on the stage of the robot.

At the beginning, the model of the swing phase is integrated until knee collision is detected. The final conditions of this model are used to calculate the angular velocities after the impact by using conservation of angular momentum about the base foot.

For this system it is possible to express angular momentum in a matrix of the form $H = A(q)\dot{q}$. Matching the angular momentum before and after the impact is possible to obtain an expression for the velocities after the impact:

$$\dot{q}^+ = (A^+(q^+))^{-1} A^-(q^-)\dot{q}^- \tag{2}$$

Here superscripts −,+ represent that the state is calculated before and after the impact. From this point the model is changed to one with the locked knee, the initial conditions used for this stage are the final positions obtained on the previous stage and the new velocities calculated. This model is integrated until the swing foot makes full contact with the ground.

After this stage, potential energy stored in the swing ankle springs is calculated, then the velocity of the stance leg is reduced in such a way that the reduction of the kinetic energy of the system match the potential energy calculated.

After the energy conversion is modeled, the collision caused by the change of the support point is calculated by using conservation of angular momentum about the point of contact.

It is also possible to calculate another expression in the form of Eq. (2) for this collision in order the calculate the velocities after this impact. But it is necessary to remember that the state vector after the impact must be referenced to the new stance leg (the previous swing leg) at the contact point.

Finally angular velocities must be adjusted in order to include the conversion of potential to kinetic energy caused by the release of the springs. First potential energy stored in the rear ankle springs is calculated, then the velocity of the stance leg is augmented in such a way that the increase of the kinetic energy of the system match the potential energy calculated.

This last state vector can now be used as initial condition for the next gait cycle.

3.1. Stability

The formal definition of *Limit Cycle Walking* according to [3] is as follows: “Limit Cycle Walking is a nominally periodic sequence of steps that is stable as a whole but not locally stable at every instant in time”.

This type of stability called *Cyclic* or *Orbital* stability, the nominal periodic sequence of step refers to a periodic or cycle motion of the walker observed in the absence of disturbances. This periodic motion is stable when the trajectories close enough to the nominal motion approach to the nominal cycle over multiple steps as shown Fig. 5.

To analyze this definition of stability the map $S(x)$ of Eq. (1) is considered. A periodic motion must fulfill the restriction $r_1(x^*) = 0$,

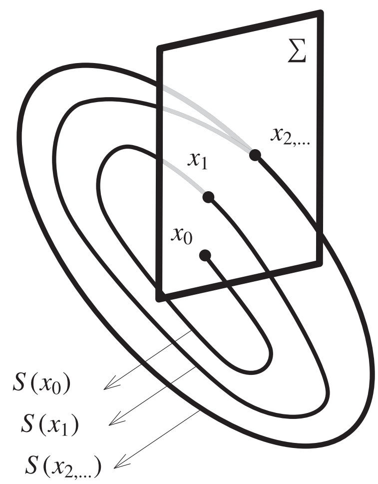

Fig. 5. Limit Cycle.

where:

$$r_1(x) = S(x) - x \tag{3}$$

The map $S(x)$ is generally called a Poincaré map and the state vector x^* is known as a fixed point and in this case belongs to the set $\sum$ of initial conditions.

The stability of a fixed point using nonlinear dynamics scheme can be defined as follows [17]: The fixed point x^* is stable if, for any $\epsilon > 0$, there exists $\delta > 0$ such that whenever $|x^0 - x^*|$, $|x^n - x^*|$ for all positive n.

It is possible to obtain a stability criterion for a fixed point by linearizing the map and considering small disturbances. The linearization of $S(x)$ about x^* is given by:

$$S(x^k) = S(x^*) + J(x^*)(x^k - x^*) \tag{4}$$

where $J(x^*) = \frac{\partial S}{\partial x}(x^*)$ is the Jacobian Matrix of map S. Considering small disturbances in the form:

$$x^k = x^* + \Delta x_k \tag{5a}$$

$$x^{k+1} = x^* + \Delta x_{k+1} \tag{5b}$$

By substituting Eq. (5b) in Eq. (4) it is possible to get the dynamic of the disturbance Δx.

$$\Delta x_{k+1} = J(x^*)\Delta x_k \tag{6}$$

Fixed point x^* will be stable if the magnitude of all eigen values σ_i of $J(x^*)$ are less than 1. In this case small disturbances will decrease step after step. This restriction for a fixed point to be stable can be written as follows:

$$r_2(x) = \max_i(|\sigma_i|) < 1 \tag{7}$$

According to this analysis, stability of walking cycle can be obtained by defining periodic motions such that the initial conditions of the step are stable fixed points. Usually Newton–Rapshon or similar algorithms are used to find the roots of Eq. (3) to obtain fixed points, however it must be considered that the algorithm can fail if the initial guess is not good enough so the solution does not converge to a fixed point or if there are no fixed points for the set of parameters.

In the case of a high degree of freedom mechanism, such as the humanoid robot we are considering, it will be very difficult to obtain periodic motions given only initial conditions of the step, so we decided to assign periodic reference trajectories so the conditions of Eq. (3) is fulfilled if the robot can follow those trajectories without falling.

After a reference walking trajectory is defined, the map $S(x)$ can be calculated by numerical integration of the model and using a feedback control then Jacobian matrix can also be calculated numerically to evaluate if the references produce a stable walking cycle. This procedure requires an accurate model of the actuators so the control torques in the simulation can represent the capabilities of the real actuators.

4. Parametric walking

Nowadays Passive Dynamic Walkers are the most representative machines that effectively use the Limit Cycle Walking paradigm, in order to define walking trajectories for an actuated walker we decide to consider the parametric trajectories of a simplified Passive Dynamic Walker model in such a way we add the less artificial constraints.

The parametric trajectories of the hip, foots and hands will be designed to mimic the movement of passive mechanisms. At the beginning the trajectories will be defined on the task space then inverse kinematic is used to find joint trajectories for each actuator.

4.1. Movement of the hip

In order to define the trajectories of the hip we will consider the simplified model of a 3D passive dynamic walker with straight legs. The simplest model of one of these mechanisms is an inverted pendulum without actuation.

By using the notation of Fig. 6, the model of the inverted pendulum on the XZ plane can be written as follows:

$$\ddot{\theta} - \frac{g}{P_l}\sin\theta = 0$$

By considering $\sin\theta = \theta$ for small angles and $\omega^2 = \frac{g}{P_l}$ the linearized model is:

$$\ddot{\theta} - \omega^2\theta = 0 \tag{8}$$

which has the solution, in terms of position and velocity:

$$\theta = a_1 e^{\omega t} + a_2 e^{-\omega t}, \quad \dot{\theta} = a_1\omega e^{\omega t} - a_2\omega e^{-\omega t} \tag{9}$$

By multiplying both sides of Eq. (8) by $\dot{\theta}$ and integrating with respect to time we get the orbital energy [18] equation:

$$\frac{1}{2}\dot{\theta}^2 - \frac{\omega^2}{2}\theta^2 = E$$

To minimize orbital energy, E=0, produces the next relation between angular position and velocity:

$$\dot{\theta} = -\omega\theta \tag{10}$$

Orbital energy remains constant during the movement, so taking E=0 implies that the pendulum will stand still when it reaches vertical position, in other words, the energy will be not enough to continue moving. If a minimum amount of energy δ is added, the pendulum will be able to complete the cycle. So the value of δ can be found in terms of the time defined to complete the cycle. Considering this explanation it is possible to rewrite Eq. (10) as follows:

$$\dot{\theta}_0 = -\omega\theta_0 + \delta \tag{11}$$

In order to get a cyclic movement, the initial condition should

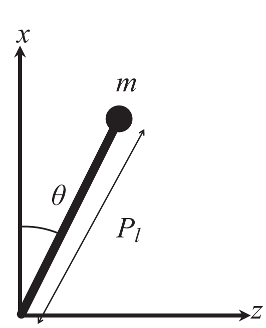

Fig. 6. Non actuated inverted pendulum.

repeat from step to step [5]. The initial and final angular positions should have the same magnitude with opposite direction and the velocity should be slightly bigger because the impact that occurs during the support transfer will produce an instantaneous decrease of the angular velocity, this leads to the next restrictions:

$$\theta(0) = -\theta_0, \quad \dot\theta(0) = \dot\theta_0, \quad \theta(t_z) = \theta_0, \quad \dot\theta(t_z) = \frac{1}{\eta}\dot\theta_0 \tag{12}$$

where parameter t_z is the time it takes the robot to complete a step and parameter $0 < \eta < 1$ is a constant used to describe the energy lost during support transfer.

In order to fully define the trajectory of the pendulum, describe by Eq. (9) it is necessary to find constants a_1, a_2 and δ by considering the restrictions written in Eqs. (11) and (12) for given t_z and η parameters.

By evaluating t=0 and $t = t_z$ in Eq. (9) the next system of equation are obtained:

$$-\theta_0 = a_1 + a_2 \tag{13a}$$

$$\omega\theta_0 + \delta = a_1\omega - a_2\omega \tag{13b}$$

$$\theta_0 = a_1 e^{\omega t_z} + a_2 e^{-\omega t_z} \tag{13c}$$

$$\frac{1}{\eta}(\omega\theta_0 + \delta) = a_1\omega e^{\omega t_z} - a_2\omega e^{-\omega t_z} \tag{13d}$$

By using (13a) and (13b) it is possible to find constants a_1 and a_2:

$$a_1 = \frac{\delta}{2\omega}, \quad a_2 = -\frac{\delta}{2\omega} - \theta_0$$

Finally using Eqs. (13c) and (13d) it is possible to find constant δ:

$$\delta = \frac{\omega\theta_0(\eta + 1)}{\eta e^{\omega t_z} - 1}$$

By substituting these constants in Eq. (9) we obtain the trajectory that drives the pendulum from $-\theta_0$ to θ_0 in time t_z, which is:

$$\theta(t) = \frac{\theta_0(\eta + 1)}{2\eta e^{\omega t_z} - 2}e^{\omega t} - \left(\frac{\theta_0(\eta + 1)}{2\eta e^{\omega t_z} - 2} - \theta_0\right)e^{-\omega t}$$

The initial position θ_0 can be found in terms of the length of the step L_z and the extension of the leg P_l, which remains constant for the stance leg:

$$\theta_0 = \arcsin\left(\frac{L_z}{2P_l}\right)$$

The trajectory of the hip $\overrightarrow{H}$ in the task space, with respect to the coordinate system in Fig. 7, is given by:

$$H_x(t) = P_l\cos(\theta(t)), \; H_z(t) = P_l\sin(\theta(t))$$

4.2. Movement of the limbs

When the first humanoid robots based on passive dynamic walkers were designed, the idea was to emphasize the low energetic consumption that these machines were able to achieve, so most of the times it was used just enough actuation to walk.

In the case of the robot build by Cornell University [19], each leg and the opposite arm make up a single link, that means that there is no relative movement. Another case is the robot *Denise* from Delft University [20], this robot uses chains to couple the movement of each leg with the opposite arm while maintaining the chest upright.

In the case of *Johnny*, energetic consumption is an important design parameter but also the capability to perform other tasks rather than walking, so it was decided to use independent actuators on each joint and synchronize the corresponding trajectories during walking. Since the movement of the hip was already defined, now is possible to design the movement of the remaining limbs: chest, swing leg and hands, by specifying the corresponding relations.

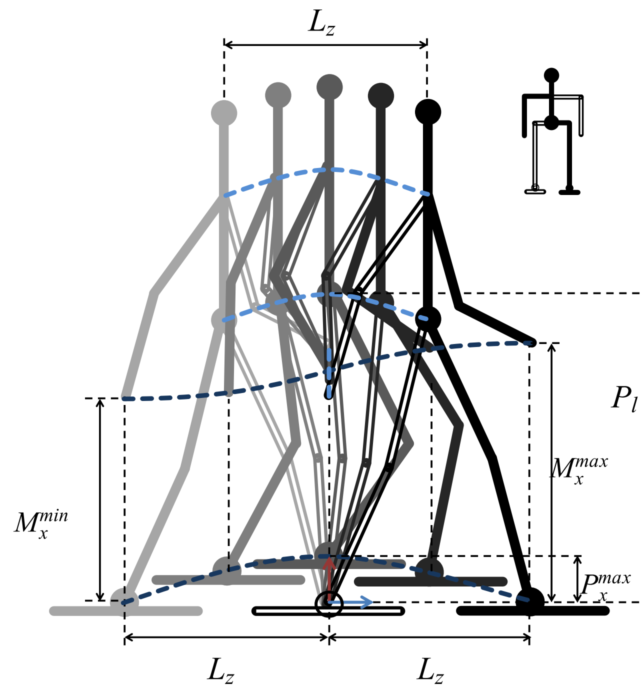

Fig. 7. Walking parameters.

In Fig. 7 the trajectories of the robot's limbs, during the single support phase, are shown. The origin of the coordinate system is on the ankle of the stance leg which is indicated with double lines, the same way that the corresponding coupled arm, we will refer to these limbs as *stance* because both of them are always above the origin of the coordinate system. On the other hand, we will refer to the leg that moves to the front and the corresponding arm as *swing* which are indicated with solid lines.

As in the previous examples the chest will be kept upright, so the position of the shoulders $\overrightarrow{S}(t)$, is defined as the position of the hip plus the length of the chest on the x-axis:

$$\overrightarrow{S}(t) = \overrightarrow{H}(t) + l_{sd}\hat{i}$$

The movement of the swing leg on XZ is defined as follows:

$$P_x(t) = \frac{P_x^{max}}{H_x^{max} - H_x^{min}}(H_x(t) - H_x^{min}), \quad P_z(t) = 2H_z(t)$$

It can be seen that the trajectory of the swing leg $\overrightarrow{P}$ is a transformation of the trajectory of the hip $\overrightarrow{H}$, the parameter P_x^{max} is the maximum height that the foot can reach and it happens at the middle of the stride. $H_{x,z}^{min}$ and $H_{x,z}^{max}$ represent the minimum and the maximum position that the hip can reach on the x- and z-axes respectively which are used to scale the movement of the swing leg.

In the case of the swing hand M^{sw}, the component on the z-axis is the same as the one of the swing foot. On the y-axis, a trajectory is defined to travel from M_x^{min} to M_x^{max} height.

$$M_x^{sw}(t) = Pol_3(M_x^{min}, 0, M_x^{max}, 0, t, t_z), \quad M_z^{sw}(t) = P_z(t)$$

where $x = Pol_3(x_0, \dot x_0, x_\tau, \dot x_\tau, t, \tau)$ describes the trajectory of third grade polynomial equation with initial conditions $(x_0, \dot x_0)$ and final conditions $(x_\tau, \dot x_\tau)$ which is valid for the time $0 \le t \le \tau$. In the case of stance hand M^{st}, the component on the z-axis is null because the stance leg is always hold on the origin. On the y-axis the stance hand moves from M_x^{max} to M_x^{min} to complete the cycle:

$$M_x^{st}(t) = Pol_3(M_x^{max}, 0, M_x^{min}, 0, t, t_z), \quad M_z^{st}(t) = 0$$

Since the trajectories are defined in task space, inverse kinematics

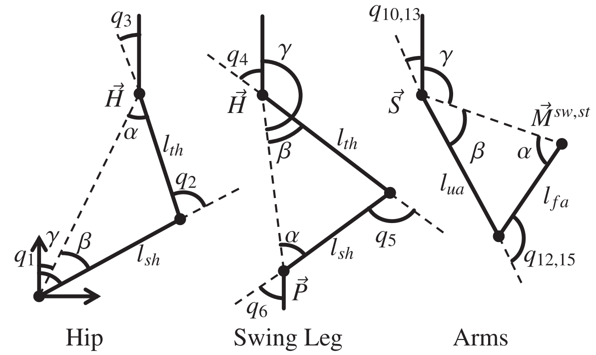

Fig. 8. Inverse kinematics.

of each limb is used to calculate reference trajectory for each actuator.

4.3. Joint trajectories

The limbs, in the case of *Johnny*, were designed in such a way that inverse kinematic would be easy to calculate, in Fig. 8 it can be seen that each arm is a 2 dof manipulator and the legs are 3 dof manipulator when the movement is restricted to the *XZ* plane.

Each limb forms a triangle, and the length of each side is known, two of them are links of the robot and the remaining one es defined by the position of the end effector (hip, swing foot or hands depending of the limb).

The angular value of each corresponding joint can be calculated with respect to the internal angles α and β, using cosine theorem, the angle γ can be calculated with respect to the position of the end effector:

$$\alpha(t) = \arccos\left(\frac{B^2 + C(t)^2 - A^2}{2BC(t)}\right), \quad \beta(t) = \arccos\left(\frac{A^2 + C(t)^2 - B^2}{2AC(t)}\right), \quad \gamma(t) = \operatorname{atan2}(C(t)_z, C(t)_x)$$

where A, B and C have the values in Table 1, depending on the limb, with respect to the vectors defined in the previous section.

After the values of $\alpha(t)$, $\beta(t)$ and $\gamma(t)$ are calculated for each triangle, it is possible to see, based on Fig. 8, that joint position of each actuator is given by the next expressions:

$$q_1(t) = \gamma(t) + \beta(t), \quad q_4(t) = \gamma(t) - \beta(t) - \pi$$
$$q_2(t) = -\alpha(t) - \beta(t), \quad q_5(t) = \alpha(t) + \beta(t)$$
$$q_3(t) = \alpha(t) - \gamma(t), \quad q_6(t) = -\alpha(t) - \gamma(t) + \pi$$
$$q_{10}(t) = \gamma(t) + \beta(t) - \pi, \quad q_{13}(t) = \gamma(t) + \beta(t) - \pi$$
$$q_{12}(t) = -\alpha(t) - \beta(t), \quad q_{15}(t) = -\alpha(t) - \beta(t)$$

Until this point the trajectories of the limbs were defined on the *XZ* plane, now the trajectories of the joints that generate movement outside this plane will be defined on joint space.

Since *Johnny* is a 3D walker it is necessary to add lateral movement on the ankles in order to avoid scuffing:

$$q_0(t) = q_7(t) = q_0^{max} \sin\left(\pi\frac{t}{t_z}\right)$$

Table 1
Inverse kinematic parameters.

Limb	A	B	C(t)
Hip	l_{th}	l_{th}	$\lvert\vec{H}(t)\rvert$
Swing foot	l_{th}	l_{sh}	$\lvert\vec{P}(t) - \vec{H}(t)\rvert$
Swing hand	l_{ua}	l_{fa}	$\lvert\vec{M}^{o}(t) - \vec{S}(t)\rvert$
Stance hand	l_{ua}	l_{fa}	$\lvert\vec{M}^{e}(t) - \vec{S}(t)\rvert$

where q_0^{max} is the maximum amplitude of the ankle.

On *Johnny*'s design the orientation of the chest can be modified by using joints q_8 and q_9 which are independent to the movement of the legs. Frontal movement driven by q_8 is useful to counter the torque produced by the movement of the swing leg. The lateral movement of the chest driven by q_9 is also useful to reduce the torque needed on the ankles in order to add lateral movement to the robot. These movements are presented using sine function as follows:

$$q_8(t) = q_8^{max} \sin\left(2\pi\frac{t}{t_z}\right), \quad q_9(t) = q_9^{max} \sin\left(\pi\frac{t}{t_z}\right)$$

Adding lateral movement to stance arm can be useful in order to decrease the torque produce by gravity on joint q_{10} however it can be dangerous to add similar movement on swing arm because it can collide with the chest. So the lateral movement of the arms is defined as follows:

$$q_{10}(t) = (q_9^{max} + q_0^{max})\sin\left(\pi\frac{t}{t_z}\right), \quad q_{13}(t) = 0$$

It is necessary to remember that the same way that legs switch roles depending on which foot is used as a supporting point, arms also switch roles every step.

With respect to head trajectories it is considered that the movement of joints q_{20} and q_{21} do not affect significantly energetic consumption so there is no movement added to this joints. This way the parametric walk shown depends on the next parameters: t_z, P_l, L_z, η, P_x^{max}, M_x^{max}, M_x^{min}, q_0^{max}, q_8^{max}, q_9^{max}.

5. Walking optimization problem

The efficiency of a walking robot depends on many parameters, considering the partial points of view mentioned in Section 1, these parameters are classified in one of the next vectors:

Mechanical parameters M_p: In this vector we should include the Denavit–Hartenberg parameters associated to the corresponding kinematic chain, in the case of our humanoid robot we select: length of the thigh l_{th} and shank l_{sh}, length of the upper arm l_{ua} and forearm l_{fa}, position of the shoulder l_{sd} and size of the hip l_{hp}.

In order to modify the mass distribution of the legs, metal disks were added to the thigh and shank, so we also include, in this vector the diameter $d_m^{1,2}$ and position $l_m^{1,2}$ of the additional masses on the legs. Finally we also include in this vector the Stiffness of the torsional spring $k_s{}^i$ that we add on each joint in order to modify the natural frequency of the system.

Gait parameters G_p: This vector include the parameters related to walking trajectories, in this case we include: Step time t_z, Extension of the leg P_l, length of the step L_z, constant of energy lost η, Maximum height of the foot P_x^{max}, Maximum height of the hand M_x^{max}, Minimum height of the hand M_x^{min}, Maximum lateral movement of the ankle q_0^{max}, Maximum lateral movement of the chest q_8^{max}, and Maximum frontal movement of the chest q_9^{max}.

Control parameters C_p: Feedback control will be used for each actuator to track its corresponding trajectory son in this vector we include PID gains for each joint: Proportional gain $k_p{}^i$, integral gain $k_i{}^i$ and derivative gain $k_d{}^i$.

The dependency of the energy consumed by the robot E_{tr}, on each walking cycle, in terms of the global parameters vector $P = [M_p\ G_p\ C_p]$ can be expressed in Eq. (14):

$$E_{tr} = f(P) \tag{14}$$

Once the parameters vector P is specified, the gait model can be evaluated to obtain walking trajectories, it is also possible to obtain control signals of each actuator and calculate the energy consumed on the stride, it is also possible to know the characteristics of the stride such as distance traveled and forward velocity, additionally some properties of the prototype can be obtained by passing the parameters to the CAD model, like total height and weight of the mechanism and the inertia properties of each limb.

Using energy consumption as optimization function will not produce the desired gait but a position of stillness, which is the trivial solution, were no energy is consumed at all. A better optimization criterion is the one called Specific Cost of Transportation C_{tr} a quantity commonly used to compare energy efficiency of walking robots.

$$C_{tr}(P) = \frac{E_{tr}(P)}{m_r(P)\ g\ d_{tr}(P)} \tag{15}$$

This criterion includes not only the energy consumed, but also the mass of the robot $m_r(P)$ and length of the step $d_{tr}(P)$ which also depends on the parameter vector P, acceleration of gravity is represented by g.

By optimizing walking parameters with this criterion, low energy consumption is patronized as well as long strides.

So formally the optimization problem is specified as follows:

$$\underset{P}{\text{minimize}} \quad C_{tr}(P) \text{subject to} \quad |r_1(x(P))| \le \delta, \text{ and } \quad |r_2(x(P))| < 1 \tag{16}$$

Here δ is the maximum error allowed between the initial conditions of two consecutive steps, as measured in Eq. (3), which implies that the trajectories of the robot should follow a fixed point, r_2 is the stability criterion described in Eq. (7), as initial conditions x dependent on the selected gait parameters, we wrote $x(P)$ to show this dependency.

6. Optimization method

When we face the problem of finding a local minimizer of a real valued function and this function is differentiable, we can choose from a wide selection of derivative based optimization methods. But when derivative of the function is unknown we can solve the problem by using direct search methods [21] or evolutionary computing [22].

The first time we intent to solve the walking optimization problem, we implement the Nelder–Mead (N-M) method [23], a direct search method that uses only function values, without any derivative information, we add a basic constraint handling rule because this method is intended to solve multidimensional unconstrained optimization problems.

N-M method uses at each step a *simplex*, which is a geometric figure in n dimensions of nonzero volume that is a convex hull of $n+1$ vertices, each one of these vertices is associated to a solution and its value function, one or more of this vertices are compared to select an operation to replace one of the vertices and modify the *simplex*, with the pass of the iterations the vertices of the simplex converge to the local minimum of the function.

As we define a more general problem N-M method became inefficient. So we focus our attention on evolutionary computing, this term now a days include most of the optimization techniques inspired by the biological evolution that uses some nature mechanisms such as mutation, recombination and selection. Among all the alternatives in this field, we choose Genetic Algorithms (GA).

6.1. Genetic algorithms

A GA is a probabilistic iterative search method which allows to find the optimal solution to multi-objective problems by convergence, this means that the probabilities of finding the solution increase as the iterations go by. GA as we know them nowadays were proposed by John Holland [24] in the early 1960s. The applications of GA include optimization problems, machine learning, pattern recognition, prediction, etc.

GAs are a good alternative to face optimization problems where cost function is not derivable or hard to calculate. These algorithms are called genetics because they used a population of solutions coded in binary sequences called *genes*. These solutions are improved over the iterations by applying mathematical operators inspired by natural evolution such as mutation, crossover, recombination and natural selection. According to the operators used in the algorithm and the way they are applied is possible to find the wide variety of GA that can be found on the literature.

A basic GA can be described as follows:

1. *Initialization*: Randomly generate an initial population of individuals.
2. *Evaluation*: Calculate the fitness of each individual.
3. *Selection*: Probabilistically select the *best* individuals.
4. *Crossover*: Apply recombination operators to selected individuals in order to generate a new population.
5. *Mutation*: Apply mutation operators on the new population.
6. *Stop*: Continue from step 2 until the stop criterion is satisfied.

The iteration form by steps 2–6 is called generation.

6.2. Implementation

On this application, the genotype that represents each individual is generated by converting the values in vector P to binary chains and concatenating them.

Initialization: To generate the initial population it is necessary to calculate the total length L_{tot} of the genotype, which depends on the number of parameters of the problem, the range each one of them belongs to and the resolution specified to describe them. Then T_{max} empty genotypes are generated of L_{tot} length randomly specified with 1 and 0 values.

Evaluation: Fitness value F can be calculated based on the function value of each individual, a higher fitness represent a better solution, so given the fitness function:

$$F_i = \frac{1}{C_{tr}(P_i)} \tag{17}$$

It follows that a better solution is related to a larger value of the fitness function. In order to evaluate an individual we need to decode each segment of the genotype to the corresponding value of vector P, the restriction function is evaluated directly.

Selection: The selection method used is called binary tournament selection, we sort the individuals in a random sequence, then each pair of individuals are compared, the *best* one is selected to become a parent, we need to execute this process twice to obtain T_{max} parents.

The criterion say that one solution is better than other depends not only on the fitness value but also on the restriction value, we say a solution is feasible if the restriction function is true and we say it is infeasible if false. We use a rule call *Superior of the feasible* that is described by Deb [25], which is useful even for multi-objective problems, the explanation is as follows:

Case	**Action**
Both are feasible	Choose the fittest solution
One is feasible and the other is not and	Choose the feasible solution
Both are infeasible	Choose the solution with smaller restriction violation

Crossover: The recombination operator is the process used to generate new individuals based on the information of the selected parents. We used a basic operator called two-point crossover which consist on generating the offspring by taking some genes of the mother or by the father depending on the interval defined by two points, randomly selected, in the genotype. There is a parameter of the GA called crossover percentage $p_{cross} \in [0\ 1]$, after selecting two parents, we generate a random value $r \in [0\ 1]$ if $r \le p_{cross}$ the crossover operator is execute otherwise the offspring are the same as their parents.

Table 2
Search space and optimal values.

Parameter	Min value	Max value	Resolution	Optimal value	Units
l_{th}	174.2	204.2	0.5	202.2	mm
l_{sh}	189.7	219.7	0.5	189.7	mm
l_{ua}	129.4	149.4	0.5	129.4	mm
l_{fa}	109.7	129.7	0.5	128.2	mm
l_{sd}	200.5	230.5	0.5	200.5	mm
l_{hp}	127	147	0.5	127	mm
d_m^1	20	50	0.5	23.5	mm
l_m^1	85.32	115.32	0.5	107.32	mm
d_m^2	20	50	0.5	49	mm
l_m^2	59.53	109.53	0.5	101.53	mm
${k_s}^1$	0	10	0.001	3.71	$\frac{g}{s^2}\times 10^9$
${k_s}^2$	0	10	0.001	9.68	$\frac{g}{s^2}\times 10^9$
${k_s}^3$	0	10	0.001	0.59	$\frac{g}{s^2}\times 10^9$
${k_s}^5$	0	10	0.001	0.03	$\frac{g}{s^2}\times 10^9$
${k_s}^6$	0	10	0.001	9.35	$\frac{g}{s^2}\times 10^9$
${k_s}^7$	0	10	0.001	9.29	$\frac{g}{s^2}\times 10^9$
t_z	0.3	0.7	0.001	0.633	s
P_l	350	400	1	390	mm
L_z	0	50	0.1	48.0	mm
η	0.01	1	0.01	0.94	mm
$\mathrm{P}_x{}^{max}$	2	20	0.01	3.39	mm
$\mathrm{M}_x{}^{min}$	240	280	1	242	mm
$\mathrm{M}_x{}^{max}$	200	240	1	223	mm
q_0^{max}	0	5	0.01	4.98	deg
q_8^{max}	0	5	0.01	1.07	deg
q_9^{max}	0	5	0.01	2.67	deg

Mutation: There is also a parameter of the GA called mutation percentage $p_{mut} \in [0\ 1]$, in this case we generate a random value $r \in [0\ 1]$ for each gene, when $r \le p_{mut}$ we invert the value of the corresponding gene. The mutation operator is included to guarantee that all possible solution can be generated during the pass of the generations.

Stop: Stop criterion depends on the user, sometimes it is defined in terms of fitness value, or a diversity parameter in this case we define max number of generations G_{max} to limit the total time of execution.

Due to the crossover and mutation operators it is possible to lose the fittest individual found on the previous generation. To avoid this situation some GA's include a process called Elitism which consists on comparing the fittest individual of the present generation and the one of the previous generation, if this last one is better we include it in the new generation by randomly select an individual to get replaced.

Due to the random processes involved in GA's is possible to obtained different results on each execution of the program, also the performance of the GA will depend on the problem and the selected parameters: T_{pop}, G_{max}, p_{cross} and p_{mut}.

7. Results

The parameters selected for the GA were: $T_{pop} = 32$ individuals, $G_{max} = 500$ generations, $p_{cross} = 0.8$ and $p_{mut} = 0.01$. For the restriction we choose $\delta = 0.015$

As the genetic algorithms works on a discrete search space we need to define a set of admissible values for each one of the parameters in vector P, this can be done by specifying a lower limit and an upper limit as well as a resolution for each parameter. In *Johnny*'s case, after defining this limits, the size of the search space (the number of all possible solutions) is 1.1517×10^{164}.

Table 2 shows the admissible values that define the search space and optimal values found after the execution of the GA.

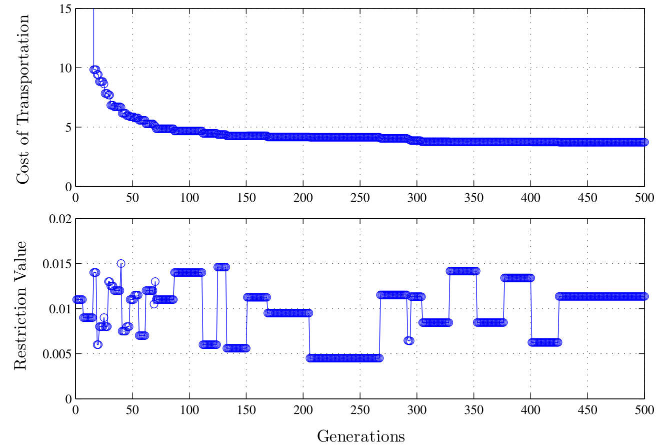

Fig. 9. Cost of transportation and restriction value.

The optimal gains found for the 16 actuators modeled are the next:

$$k_p = [1.67\ \ 4.00\ \ 2.71\ \ 4.02\ \ 3.54\ \ 2.71\ \ 4.00\ \ 1.67\ \ 1.03\ \ 1.36\ \ 2.13\ \ 3.76\ \ 0.39\ \ 3.02\ \ 1.36\ \ 0.39]$$

$$k_i = [4.35\ \ 0.77\ \ 3.04\ \ 4.48\ \ 0.62\ \ 3.04\ \ 0.77\ \ 4.35\ \ 4.86\ \ 4.56\ \ 0.70\ \ 2.78\ \ 2.93\ \ 1.34\ \ 4.50\ \ 0.51]$$

$$k_d = [4.94\ \ 3.23\ \ 4.34\ \ 4.53\ \ 3.02\ \ 4.34\ \ 3.23\ \ 4.94\ \ 2.75\ \ 4.91\ \ 1.12\ \ 1.42\ \ 1.47\ \ 4.32\ \ 0.72\ \ 4.67]$$

With these parameters the Cost of transportation obtained is $C_{tr} = 3.72$ and error at the end of the cycle is $r(x) = 0.01$ rad (Fig. 9). The length of the gait is 96 mm and it lasts for 0.633 s equivalent to a speed of 0.15 m/s. The height of the robot is 890 mm and its weight is 3782 g. The information about the mechanical properties is obtained automatically by using a link between the mathematical model and the CAD software, this link allows to get the information about the inertia tensor and the center of mass position of each limb for each individual during the execution of the GA.

The average time cost for evaluating each individual is about 2 s, on a commercial personal computer, which leads to a total time cost of approximately 9 h using the mentioned parameters in the GA, the time cost depends mainly on the accuracy chosen to integrate the mathematical model and the time it takes for the CAD software to manipulate and update the information about limb's parameters.

8. Conclusions and future work

By using Genetic Algorithms it was possible to optimize the mechanical design of a 22 joint humanoid robot on a search space which includes 13×10^{99} possible solutions. The use of GAs is fully justified not only because of the dimension of the search space but also because of the difficulty of the cost function which is not possible to be represented as a differentiable function. The optimal solution refers to minimum energetic consumption with maximum distance traveled while respecting the constraints that define a Limit Cycle.

By using the optimal values found it was possible to build the corresponding prototype called *Johnny* which is able to successfully walk. Nowadays we are working on the measurement of the actual Cost of Transportation of the real prototype in order to make a comparison against the existing robots. We are also working on the design of a global control law to improve stability of walking by using disturbance rejection [26] and observer based techniques which allows the use of inertial information of the robot to avoid falling.

Finally we are developing real-world applications for *Johnny* such as manipulation and object recognition by using visual information, these tasks involves autonomous walking and requires the robot to learn at some level [27] how to move on unknown environments.

References

[1] M. Sánchez, R. Cisneros, J. Ibarra, Mechanical design and kinematic analysis of the ah1n1 humanoid robot, in: Proceedings of the IEEE 21st International Conference on Electronics, Communications and Computers.

[2] R. Núnez, J. Ibarra, Optimal design for a humanoid robot based on passive dynamic walkers and genetic algorithms, in: Proceedings of the 12th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE).

[3] D. Hobbelen, M. Wisse, Humanoid robots, human-like machines, chapter limit cycle walking, I-Tech Educ. Publ. (2007) 277–294.

[4] M. Vukobratovic, B. Borovac, Zero-moment point—thirty five years of its life, Int. J. Humanoid Robot. 1 (2004) 157–173.

[5] T. McGeer, Passive dynamic walking, Int. J. Robot. Res. 9 (1990) 62–82.

[6] H. Ohta, M. Yamakita, From passive to active dynamic walking, in: Proceedings of the 38 Conference on Decision and Control, 1999, pp. 3883–3885.

[7] M. Uemura, K. Kanaoka, S. Kawamura, A new control method utilizing stiffness adjustment of mechanical elastic elements for serial link systems, in: IEEE International Conference on Robotics and Automation, 2007, pp. 1437–1442.

[8] H. Ohta, M. Yamakita, Periodic stabilizing of control systems with collisions—application to walking robots, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2001, pp. 980–985.

[9] D. Kirk, Optimal Control Theory: An Introduction Dover Publications, San José, California

[10] J. Lee, J. Lee, Optimal walking trajectory generation for a biped robot using multi -objective evolutionary algorithm, in: Asian Control Conference, 2004, pp. 357–364.

[11] K. Endo, T. Maeno, Co-evolution of morphology and walking pattern of biped humanoid robot using evolutionary computation—designing the real robot, in: Proceedings of the 2003 IEEE International Conference on Robotics and Automation, 2003, pp. 1362–1367.

[12] T. McGeer, Passive walking with knees, in: Proceedings of the IEEE Conference on Robotics and Automation, 1990, pp. 1640–1645.

[13] M. Coleman, A. Ruina, An uncontrolled toy that can walk but cannot stand still, Phys. Rev. Lett. (1998) 3658–3661.

[14] M. Wisse, A. Schwab, Skateboards, bicycles and three-dimensional biped walking machines: velocity-dependent stability by means of lean-to-yaw coupling, Int. J. Robot. Res. (2005) 417–429.

[15] S. Collins, M. Wisse, A. Ruina, A three-dimensional passive dynamic walking robot with two legs and knees, Int. J. Robot. Res. (2001) 607–615.

[16] T. Narukawa, K. Yokoyama, M. Takahashi, K. Yoshida, Design and construction of a simple 3d straight-legged passive walker with flat feet and ankle springs, JSME J. Syst. Des. Dyn. 3 (2009) 1–12.

[17] M. Coleman, A stability study of a three dimensional passive-dynamic model of humanoid gait (Ph.D. thesis), Cornell University.

[18] S. Kajita, H. Hirukawa, K. Harada, K. Yokoi, Introduction to humanoid robotics, Springer Tracts in Advanced Robotics 101 (2014) 1–210.

[19] S. Collins, A. Ruina, A bipedal walking robot with efficient and human-like gait, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, 2005, pp. 1983–1988.

[20] M. Wisse, Three additions to passive dynamic walking; actuation, an upper body, and 3d stability, in: Proceedings of the International Conference on Humanoid Robots.

[21] T. Kolda, R. Lewis, V. Torczon, Optimization by direct search: new perspectives on some classical and modern methods, SIAM Rev. 45, 384–482.

[22] A. Eiben, J. Smith, Introduction to Evolutionary Computing, Springer, Natural Computing Series Amsterdam, The Netherlands

[23] J. Lagarias, J. Reeds, M.H. Wright, P. Wright, Convergence properties of the nelder - mead simplex method in low dimensions, SIAM J. Optim. 9 (1998) 112–147.

[24] J. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, Michigan; re-issued by MIT Press.

[25] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: nsga-ii, IEEE Trans. Evol. Comput. 6 (2002) 182–197.

[26] Y. Huang, Q. Wang, Disturbance rejection of central pattern generator based torque stiffness controlled dynamic walking, Neurocomputing 170 (2015) 141–151.

[27] K. Madani, C. Sabourin, Multi-level cognitive machine-learning based concept for human-like artificial walking: application to autonomous stroll of humanoid robots, Neurocomputing 74 (2011) 1213–1228.

R.S. Núñez Cruz is a Ph.D. scholar in the Department of Automatic Control at CINVESTAV-IPN, México. He is pursuing his research in Model and Control of Humanoid Robots. His research interests include Passive Dynamic Walking, Genetic Algorithms, Optimal Control, Artificial Vision. He has received his bachelor's degree in Mechatronics Engineering from UPIITA National Polytechnic Institute, 2009, Master's degree in Automatic Control from CINVESTAV, 2012.

J.M. Ibarra Zannatha is a Professor in the Department of Automatic Control at CINVESTAV-IPN, México. He has received his bachelor's degree in Comunications and Electronics Engineering from ESIME-IPN. He has received Ph.D. from Institute de Recherche en Informatique et Systémes Aléatoires, Université de Rennes II, Rennes, Francia. His research interests include Artificial Vision and Control of Mobile Robots and Manipulator Arms.