REALISTIC HUMANOID ROBOT SIMULATION WITH AN OPTIMIZED CONTROLLER: A POWER CONSUMPTION MINIMIZATION APPROACH

JOSÉ L. LIMA, JOSÉ C. GONÇALVES

Electrical Engineering Department of Polytechnic Institute of Bragança
{jllima, goncalves}@ipb.pt
Bragança, Portugal

PAULO J. COSTA, A. PAULO MOREIRA

Department of Electrical and Computer Engineering
Faculty of Engineering of University of Porto
{paco, amoreira}@fe.up.pt
Porto, Portugal

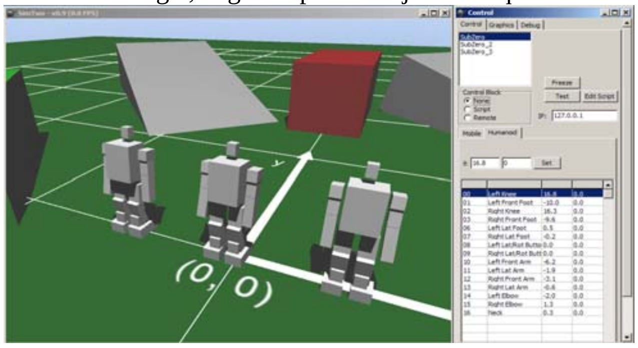
This paper describes a humanoid robot simulator supporting joint trajectory optimization, following accurately the real robot characteristics. The simulator, based on a rigid body simulator (Open Dynamics Engine) and an OpenGL based graphics library (GLScene), provides instant visual feedback and realistic dynamics. It allows to design and test behaviours and control methods without access to the real hardware, preventing damages in the real robot in the earlier stages of development. Having in mind the energy consumption minimization, the low level trajectory planning is discussed and experimental results are presented. The proposed methods are shown to minimize the total energy assuming two intervals of constant acceleration.

1. Introduction

The importance of development and research in biped robots have been increased rapidly and resulted in a variety of prototypes that resemble biological systems [1]. Legged robots have the ability to choose optional landing points, an advantage to move in rugged terrains, and two legged robots are also able to move in typical human environment. Nowadays, robot competitions like the RoboCup [2] Humanoid League [3] are responsible for the notable increase in this area.

As developing new control software for robots can be a difficult and challenging task, the ability to rapidly prototype software, within a simulation environment, can speed up the process if the resulting software can be easily transfered from simulation to real world systems [4].

While simulation is a powerful tool for speeding up the control software development there is an essential fact that simulators must capture the most important environment characteristics. In the humanoid robot simulator, servo motor limits and dynamics must be addressed. However, developing simulators with accurate dynamic models that can be simulated in real-time is a non trivial problem [4].
A screenshot of the developed simulator is shown in Figure 1, where a 3D scene shows the three robots human like and some obstacles, a table shows the desired joint variables such as angle, angular speed and joints torque.



SubZero
SubZero_2
SubZero_3

Control Block: [✓ None] Script Remote

IP: 127.0.0.1

θ: 16.8 0

00	Left Knee	16.8	0.0
01	Left Front Foot	-10.0	0.0
02	Right Knee	16.3	0.0
03	Right Front Foot	-9.6	0.0
06	Left Lat Foot	0.5	0.0
07	Right Lat Foot	-0.2	0.0
08	Left Lat/Rot Butto	0.0	0.0
09	Right Lat/Rot Butt	0.0	0.0
10	Left Front Arm	-6.2	0.0
11	Left Lat Arm	-1.9	0.0
12	Right Front Arm	-3.1	0.0
13	Right Lat Arm	-0.6	0.0
14	Left Elbow	-2.0	0.0
15	Right Elbow	1.3	0.0
16	Neck	0.3	0.0

Figure 1. Screenshot of the simulator.

As an important new feature, robots can be built through a configurable structure based on a *xml* description file. The *xml* robot description and program script window are the main improvements since humanoid robot simulator was presented [5]. The new and accurate servo motor model and multi robot support platform are also a progress too.
There are several robot simulators, such as *Simspark*, *Webots*, *MURoSimF* and *Microsoft Robot Studio* that provide humanoid simulation capability. Meanwhile, the developed simulator allows to build and to test the low and high level controllers, with a configurable frequency, in a way that can be mapped with the reality.
The paper is organized as follows: Initially, the developed simulator basis and structure is presented. Then, section 3 presents the joint trajectory planning where minimum energy consumption method is described. Experimental results are presented further in section 4. Finally, section 5 rounds up with conclusions and future work.

2. Open Dynamics Engine Simulation

The dynamical behaviour of an object is possible when its model is known. A precise modeling of legged locomotion systems requires high dimensional nonlinear multibody systems dynamics with constraints. Further complex tasks are generation, optimization and control of stable motions for such systems [6]. A simulator of a humanoid robot can be built, avoiding explicitly building its complex model, based on a physics engine. A physics engine is a key to make simulation useful in terms of robot control. There is a number of simulation engines for rigid body motion that are unusable for simulating the mechanical interactions of rigid parts [4]. For real-time simulation, an accurate and fast simulation engine as ODE, Open Dynamics Engine, must be used [7]. It is essentially a simulation library that provides support for rigid body motion, rotational inertia and collisions treatment. It also allows to use any visualization library, in this case it was GLScene that provides visual components and objects allowing description and rendering of 3D scenes in an easy way [8]. GLScene also enhances visualization including shadows, textures, projections, illuminations that provides depth perception. Camera positioning and zooming around the objects allow to obtain detailed data in the scene.

2.1. *Humanoid construction based on XML*

The developed simulator builds each robot based on a description expressed by a *xml* (Extensible Markup Language) file. It is a general specification language that allows its users to define their own elements [9]. Positions, sizes and masses perform the description of bones and positions, axis, limits and types perform the description of joints that imitate servo motors with dynamics and limits. By this way, it is an easy task to modify the robot structure once there is no necessity to compile a new application, making the simulator useful to others beyond the developers.

3. Humanoid Joint Trajectory controller

This controller level accepts, for each servo, angles, angular speeds and timings requirements from the higher level. The main objective of this controller is to build and to follow the trajectories established by angles and angular speeds requirements having in mind the energy consumption minimization. The *xml* language is also used to store the movements and timings of each joint establishing the humanoid robot attitude.

3.1. *Servo motor and simulator models*

The servo motor response, such as maximum acceleration and speed, must be known in order to draw simulator trajectories compatible to the real robot. The joints closed loops controllers must also have the same response as servos have.

An input step, from 0 to 50 degrees with a sample frequency of 30 Hz and an inertial mass, is presented in Figure 2 (orange lozenges) and allows to obtain the desired parameters. The maximum speed can be found by the servo motor angle maximum derivative (ω_{max}=281 *deg/s*) and the acceleration can be found by maximum second derivative (a_{max}=1400 *deg/s*^{2}). This test was made assuming that friction and wind-up saturation non linearities are neglected.

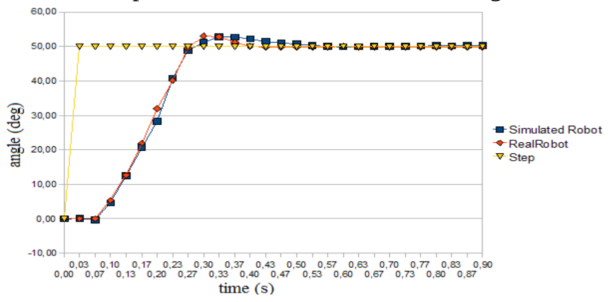


Figure 2. Real and simulator servo motor step response to a step input.

The implemented servo motor model in simulator, based on the real servo motor dynamics, is tested for the same input step and with the same inertial mass in order to validate its similarity with the real one and is presented in the same figure (blue squares).

3.2. *Trajectory planning*

The joints controller finds the intermediate trajectories that takes joints to the desired states and follows them. It is developed to minimize the energy consumption. Suppose that for t=t_1 (actual time) it is measured angle θ_1 and angular speed ω_1, and for t=t_2 (next period) it is desired position θ_2 and angular speed ω_2, as illustrated in Figure 3 a) and b), that shows the used symbology and some examples of possible trajectories, assuming a constant acceleration in $[t_1,t_m]$ and $[t_m,t_2]$.

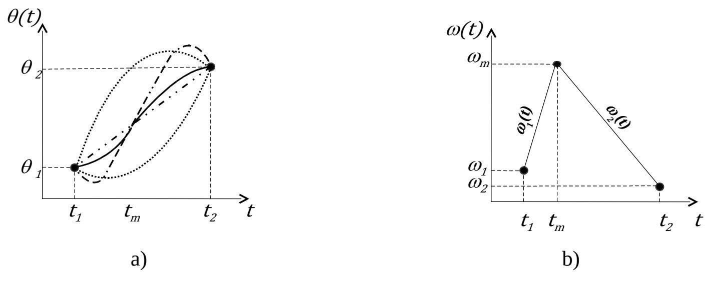


Figure 3. Joint state: a) angle ref. and b) speed ref.

It is necessary to calculate the angle and angular velocities equations that result in the desired conditions. There are several solutions for t_m, for the same initial and final conditions as presented in next equations. The covered angle ($\theta_2 - \theta_1$) can be expressed by the triangle areas (A_I, A_{II} and A_{III}) presented in Figure 4 and equation 1. Equation 2 allows to find the linear function $\omega_m = f(t_m)$, represented by *L2* line in Figure 4.

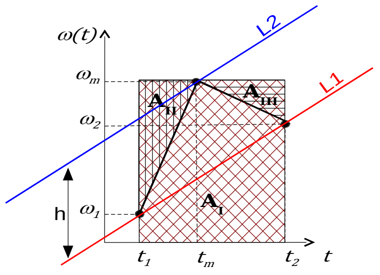


Figure 4. w_m position freedom.

$$\theta_2 - \theta_1 = A_I - A_{II} - A_{III} \tag{1}$$

$$\omega_m = \frac{2\left(\theta_2 - \theta_1 - \frac{t_m - t_1}{2}\omega_1 - \frac{t_2 - t_m}{2}\omega_2\right)}{t_2 - t_1} \tag{2}$$

In fact, equation 2 is a linear function of t_m and its slope (derivative function) can be found by $(\omega_2 - \omega_1)/(t_2 - t_1)$, the same slope as *L1* line. These lines, *L1* and *L2* are deviated by *h* that depends on the covered angle and it can be found by equation 3.

$$h = \omega_m(t_1) - \omega_1 \tag{3}$$

So, the *h* value can be determined by equation 4.

$$h = 2\frac{\theta_2 - \theta_1}{t_2 - t_1} - \omega_2 - \omega_1 \tag{4}$$

The angle equation, $\theta_{ref}(t)$, can be found resorting to the formula of uniform linear accelerated movement for each time interval $[t_1, t_m]$ and $[t_m, t_2]$ as presented in equations 5 and . By this way, t_m can be placed in $[t_1, t_2]$ interval, according to equation 2, that allows to cover the desired angle. Next subsection discusses t_m positioning from the point of view of energy consumption.

$$\theta^{Ref}(t) = \theta_1 + \omega_1(t - t_1) + \frac{1}{2}\frac{\omega_m - \omega_1}{t_m - t_1}(t - t_1)^2 \tag{5}$$

$$\theta^{Ref}(t)=\theta^{Ref}(t=t_m)+\omega_m(t-t_m)+\frac{1}{2}\frac{\omega_2-\omega_m}{t_2-t_m}(t-t_m)t^2 \tag{6}$$

3.3. *Energy consumption minimization controller method*

As humanoid robot is powered by on-board batteries, energy consumption must be reduced as much as possible. Trajectories design task should care energy consumption and minimize it. The presented method allows to save up to 13 % of energy only by using a better software implemented controller.

Assuming that the instant power consumption by a servo motor can be determined as the torque and the angular speed ω product ($P=k.I.a.\omega$), where I is the moment of inertia, k a scalar gain and a the angular acceleration, it is possible to place t_m in the energy minimization instant as described in this subsection. The moment of inertia depends on each joint and robot posture but can be considered constant as a first approach. As example, an extended arm that measures approximately 0.2 *m* and weights 0.174 *kg* has an moment of inertia of 0.00232 $kg.m^2$ ($I=mL^2/3$, where m is the mass, and L the body length). The k value allows the conversion from *deg/s* to S.I. units expressed as $4\pi^2/360^2$. For the first time part (where $t \in [t_1,t_m]$) there is an angular speed $\omega_1(t)$ with an acceleration a_1 and for the second time part (where $t \in [t_m,t_2]$) there is an angular speed $\omega_2(t)$ with an acceleration a_2. The instant power consumption can be described by $P(t)=kIa\omega(t)$. The total energy can be described by the power integral as presented in equation 7.

$$E_{Tot}=\int_{t_1}^{t_m} P_1(t)+\int_{t_m}^{t_2} P_2(t)\,dt \tag{7}$$

The minimum energy consumption t_m instant can be found when its first derivative equals zero, $dE_{Tot}/dt=0$, that allows to find the t_m instant presented in equation 8. Equation 2 allows to find the ω_m value for the desired instant t_m.

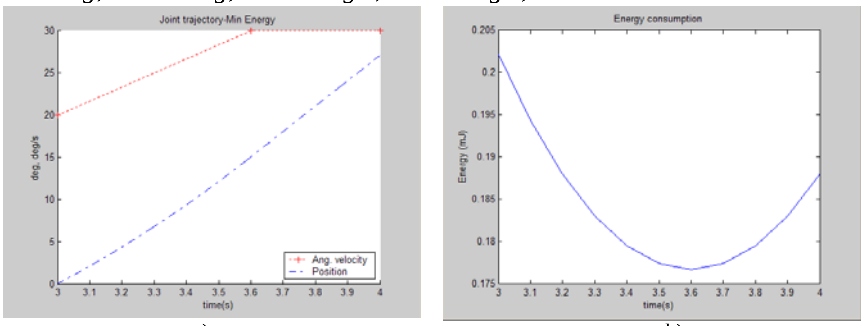
$$t_m=\frac{w_1 t_1-2w_2 t_2-2\theta_1+2\theta_2+w_2 t_1}{-w_2+w_1} \tag{8}$$

If the optimal t_m timing is outside $[t_1,t_2]$ interval, maximum acceleration should be done at t_1 if $t_m < t_1$ or to t_2 if $t_m > t_2$.

4. Experimental Results

This chapter presents, in a short way, the results that simulator can achieve when the conditions previously presented are applied. The energy consumption minimization is tested with successful results.

Having in mind the energy consumption minimization, Figure 5 a) presents the trajectory and angular speed for an example with the conditions:
θ_1=0 *deg;* θ_2=27 *deg;* ω_1=20 *deg/s;* ω_2=30 *deg/s;* t_1=3 *s* and t_2=4 *s*.



a) b)

Figure 5. a) Energy minimization method; b) Energy consumption in *[t_1,t_2]* interval.

Minimum energy consumption function from t_1 to t_2 timing is also illustrated in Figure 5 b) that shows the optimal t_m value at 3.6 seconds.
As example, the same conditions were applied to the neck joint. The angle and reference angle are presented in Figure 6.

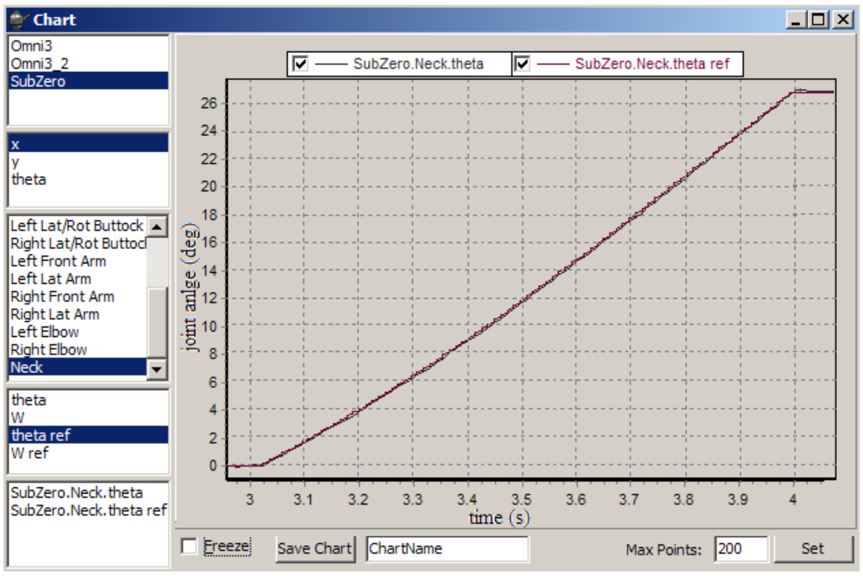


Figure 6. Neck joint minimum consumption energy test.

At the left side, user can add which robot, joint and variables are requested to appear in the graphic at the right side. In the presented example, neck joint angle follows the reference from 0 to 27 degrees in 1 second ($t_1=3\ s$ and $t_2=4\ s$) using the minimum energy consumption trajectory. The overlapped curves also allow to validate the servo motor model implemented in the simulator.

5. Conclusion and Future Work

In this paper a humanoid simulator, based on a dynamics engine and a 3D visualization engine, is presented. The real robot limitations and dynamics are taken into account. A low level trajectories controller that allows to minimize energy consumption is presented. The results allow to validate the simulator and show the realistic simulation.
As future work, more control strategies will be implemented using the high level programming based on a Pascal script dialect that allows users to create their own control programs while results are real-time presented. Enhancing the simulator with realistic sensors, such as accelerometers and gyroscopes, is also a future step.

References

1. T. Suzuki and K. Ohnishi, *Trajectory Planning of Biped Robot with Two Kinds of Inverted Pendulums*, Proceedings of 12th International Power Electronics and Motion Control Conference (2006).
2. Robocup, 2007, http://www.robocup.org/
3. Humanoid League, 2007, http://www.humanoidsoccer.org/
4. B. Browning and E. Tryzelaar, *UberSim: A Realistic Simulation Engine for RobotSoccer*, Proceedings of Autonomous Agents and Multi-Agent Systems (2003).
5. J. Lima, J. Gonçalves, P. Costa and A. Moreira, *Realistic Behaviour Simulation of a Humanoid Robot*, Proceedings of 8th Conference on Autonomous Robot Systems and Competitions (2008).
6. M. Hardt, O. Stryk, D. Wollherr and M. Buss, *Development and Control of Autonomous, Biped Locomotion using Efficient Modeling, Simulation, and Optimization Techniques*, Proceedings of the 2003 IEEE International Conference on Robotics & Automation (2003).
7. Russell Smith, Open Dynamics Engine, 2000, http://www.ode.org
8. GLScene, 2000, http://glscene.sourceforge.net
9. E. Harold and W. Means, *XML in a Nutshell* (2004).