Implementation and control of a biped walk on a Humanoid Robot

Gustavo R. Silva, Leonardo L. R. Barbosa ,Débora T. Valverde, Gabriel de B. Silva, Heitor M. S. Cunha, Iago A. Peixoto, João V. G. Silva, Renata Bernardes, Rogério M. R. Filho.

***Abstract*— This paper presents the implementation of a biped walking using a simple hip control to generate stability in a humanoid robot. It includes the technic used to apply this control using a single Inertial Measurement Unit (IMU), and also a schematic of the software implemented to generate such pattern.**

***Index Terms* — Humanoid robots, Legged locomotion, Robot control, Robot programming.**

I. Introduction

This work was developed by EDROM(Equipe de Desenvolvimento de Robótica Móvel), a mobile robotic development team, that does researches about humanoids robots and LEGO robots in order to increase the students' knowledge in various fields of mechatronics engineering, furthermore, the project is used to compete at events around the Latin America and also sometimes out of Latin American, at world competitions like the RoboCup[1].

The humanoid robot, which was named EVA, was completely made by EDROM, including the mechanic and electric projects. The robot was made to be able to play soccer games following the RoboCup rules. And for that, the robot body needs to match human proportions, and perform human like movements, so it is necessary to develop a biped walking that allows the robot to move like a human.

To generate the biped walking we used a kinematics that depended only in some parameters, such as hip and pitch offset, to calculate the movement, not having any feedback information. Because of that the walking didn't work for different conditions of the terrain, so using an IMU, we were able to implement a control system that resulted in a more stable movement.

To implement all that in the software the concepts of Object-Oriented Programming (OOP) were used, and the language chosen was C/C++, in order to make the code easier to read and use.

Débora T. Valverde is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: debora_tahara@ hotmail.com).

Gabriel de B. Silva is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: gabriel_gx7@yahoo.com).

Gustavo R. Silva is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: gustavorezendesilva@ hotmail.com).

Heitor M. S. Cunha is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: heitormenezes10@ gmail.com).

Iago A. Peixoto is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: iagoaph@gmail.com).

João V. G. Silva is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: joaovitor_g.s@hotmail.com).

Leonardo L. R. Barbosa is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: leonardobarbosa@meca.ufu.br).

Renata Bernardes is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: renata.bernardes@ufu.br).

Rogério M. R. Filho is a Mechatronics Engineering student of the Department of Mechanical Engineering (FEMEC) from the Federal University of Uberlandia (UFU) (e-mail: rfilho_11@hotmail.com).

II. Development

With the purpose to create a stable biped walk an appropriated kinematics, alongside a control system that adjust the hip inclination using the feedback information provided by a IMU placed at the hip of the robot, were implemented.

But before utilizing the information provided by the sensor must be treated since it has a lot of noise, that is done by applying two kinds of filters, if the raw data wasn't treated the control system wouldn't work.

In order to effective make the robot walk using an embedded system, all that theory must be implemented in an optimized way in the software. With the intention to do that, a software that is divided in two processes, one handling the acquisition of the sensor and sending that data and the other calculating the movement using the information sent was developed.

A. Kinematics

To generate the biped walking that looks like human's walking it was used a combination of three sins for the foot trajectory, one in each axis, that together makes the robot walking seems like a human. And for that, it is needed to find the angles that result in the trajectory hoped.

In the three sins it is possible to change some parameters and with that change the movement as well. Other parameters besides the sins can also be changed, such as hip and pitch offset, to change the upper body position, and given the parameters the movement can be calculated.

The kinematics developed was based in the one used in the DARwIn-OP [2], but since EVA has a different size and proportion it was needed to be adapted.

This model has some parameters that can be changed to correspond with the robot that is being used, and how we want the gait to be.

Parameters that can be set:

- Robot dependent:
  - Leg size;
  - Thigh size;
  - Calf size;
  - Ankle size;
- Gait dependent[3]:
  - X offset (front and back);
  - Y offset (right and left);
  - Z offset (up and down);
  - Roll offset (foot degree on x axis);
  - Pitch offset (ankle degree on y axis);
  - Yaw offset (foot degree on z axis);
  - Hip pitch offset (HPO) (degree on y axis);
  - Period Time (time of one step);
  - Dps Ratio (time of 2 foots touching ground/Total time);
  - Step forward back (Differential distance in mm between both foots in x axis);
  - Step left right(Differential distance in mm between both foots in y axis) ;
  - Foot Height(Foot elevation during walk);
  - Step Direction (Direction of the walking);
  - Step forward back ratio;
  - Swing right left;
  - Swing top down;
  - Pelvis offset(pelvis roll offset x axis);
  - Arm swing gain;

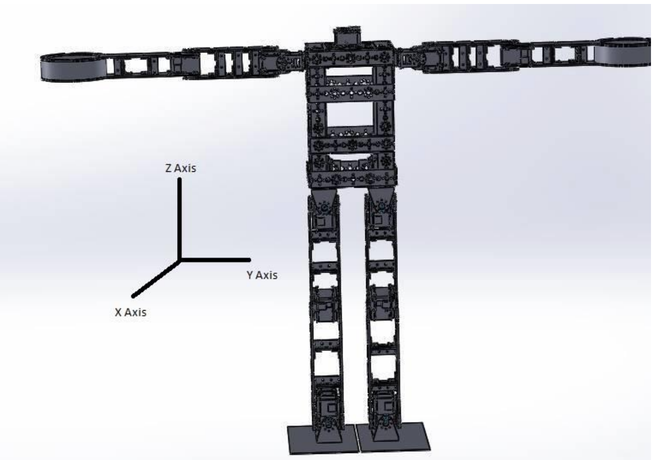

Figure 1. Robot Axis.

The movement is generated dividing one step of the walk in various poses, them calculating the angular position of each joint, based on the parameters set, for all the poses. The group of poses that form a step is called a page.

Those parameters need to be set to each terrain that the robot walks, and when they are tuned, the movement is reliable as long there is no disturbance or variation in the terrain.

B. Control System

Since a soccer match may include several robots, a ball and two goals, which all can be a disturbance to the robot walking and yet the floor may be inclined, so a gait without any kind of control is not reliable.

The lack of stability can be explained by the Zero-Moment Point theory [4-5], which states that the sum of all ground reaction forces **R** located in a point **P** that results in a balance of all the external moments that causes the robot to tip around the edge of the foot, such as the inertial and gravitational moment, is called Zero-Moment Point (ZMP).

And in order to satisfy the moment equilibrium and result in a dynamic stable movement, the support foot must rest fully on the ground, and the ZMP must be within the support polygon, Fig. 2. But in the EVA's case, the disturbances that appear in a match cause the stability requirements to fail.

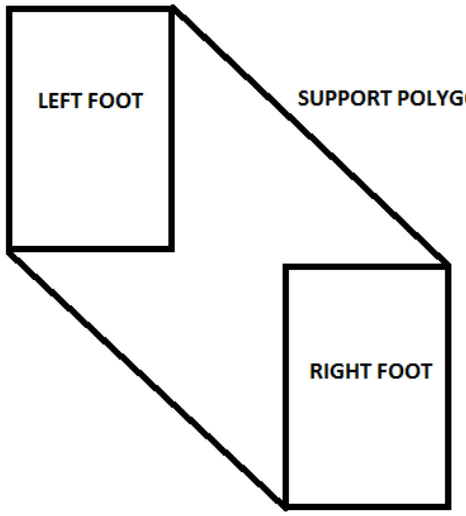

Figure 2. Support polygon of a biped Humanoid.

Therefore, it was necessary to implement a control system to make the biped walking more stable. Through observations of numerous tests, on the open loop walk, it was noted that when the hip inclination was greater than a certain value the robot would fall.

Due to the great influence of the hip angulation in the gait stability, it had to be corrected in order to prevent it to become bigger than the critical angle, the first inclination that results in the robot to fall. And this can be achieved by controlling the kinematics parameter called hip pitch offset.

To do that an IMU, sensor PhidgetSpatial 3/3/3 model 1042, containing accelerometer, gyroscope and compass on three axes, was utilized to measure the angle between the hip and the vertical axis (z axis), this difference will be called robot angle.

Although, the information provided by the sensor feature high frequency noise, especially in respect of acceleration data, and yet, the quantities measured by the sensor does not directly offer information about its tilt. Thus, first a digital low pass filter was used to filter the high frequency noise from the acceleration data. Then the cutoff frequency was dimensioned in 6Hz by analyzing the result obtained using a calibrator of accelerometers.

The resulting data were subjected to a second filter, developed by Sebastian Madgwick [6]. The Madgwick filter uses an approach called "fusion sensor", where different data from different sources are grouped in order to generate a result with less noise.

The filter receives as input the acceleration data and angular velocity, groups them, converts to quaternion and calculates the difference between this and the quaternion calculated before that, that is, to the filter, the referential is the former quaternion. But this former quaternion has its value associated with the quaternion that preceded it, as well as all previous ones, that way being oriented towards an initial quaternion, which is nothing more than a coordinate system which is established in the first iteration with the filter. The quaternion value obtained must be converted to Euler angles, and only then, can be interpreted.

For this application, the reference is set at two axes (X and Y) based on the gravitational field. In the third axis (Z) the reference is the point at which the sensor was in this axis when it started, i.e., the first position in the Z axis is the reference.

Contextualizing the axes of the sensor with the axes considered while modeling the robot, a rotation around the Y axis for the sensor is equivalent to an angle of pitch, a rotation in the Z axis corresponds to an angle of yaw, and lastly, a rotation in the X-axis of the sensor is equivalent to an angle of roll.

After measuring the robot angle it's needed to apply a correction to the hip pitch offset, so the step can be recalculated generating a different page which results in a more stable walk.

And with the objective to correct the robot tilt, a purely proportional control system, was implemented. This kind of control is described in several control system books [7-8] as a system that simply feeds the error multiplied by a proportional gain to the plant - what is being controlled. The proportional controller is essentially an amplifier with an adjustable gain.

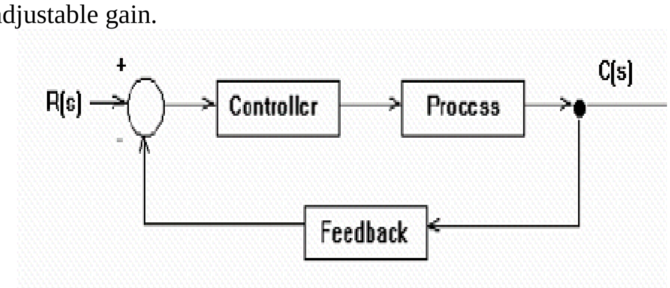

Figure 3. Robot Control System.

The purely proportional control system applied to the humanoid robot can be represented by the Fig. 3. In this system the input R(S) represents the desired inclination of robot hip, while C(S) is the real tilt of the body that is monitored by the IMU. The data collected by the sensor are compared to the reference value, input value, resulting in an error that is multiplied by a constant and feedback to the controller.

This action results in a variation of the hip pitch offset that is proportional to the error of the system, which leads the output signal to approach the input.

After the control system is applied the calculation of the new hip pitch offset can be described by (1).

$$HPO_C = HPO + RA * K_p \quad (1)$$

HPO_C – Hip pitch offset corrected;
HPO – Hip pitch offset;
RA – Robot angle;
K_p – Proportional gain;

` Succeeding the calculation of the new hip pitch offset a new step is calculated and then performed, and in the end of the execution the robot angle and the HPOc is calculated again in order to keep correcting the walking, this process is repeated while the robot continues to walk. In average one single step takes about 2 seconds to be performed, which means that the robot angle is corrected every 2 seconds.

C. *Software*

To effectively make the robot walk the kinematics and control system must be implemented in a programming language, alongside some necessary tools to communicate with the motors used in EVA.

And for that the software was structured based on the principles of Object Oriented Programming (OOP), the language used was C/C++ because of its versatility, the amount of libraries that are available, and because C++ supports OOP, which enables to make the code more modularized, easier to manipulated and read, and that is fundamental since the code gets really complex when it's fully developed, especially when the behavior necessary to play a soccer match is implemented.

With the purpose to translate the movement generation and execution into a programming language, several classes were created to abstract what is used by the robot, such as the kinematics, the pages and poses that describe the movement, the USB converter to serial communication, a single motor, and a set of motors.

The name and function of each class are:

- Pose: this class represents a pose of the robot gait, and it stores an angle in degrees for each motor, the time of execution of the pose, and the pause time between poses;
- Page: it is a set of poses that together represents a single step;
- Walking: it contains the mathematical model of the kinematics and all the parameters of the gait, it is used to generate the page of the walking;
- Channel: it is responsible for the communication between the computer and the motors, it is an abstraction of the couple USB convertors to serial, which can be used. This class can write and read from the motors regardless of which drive is being used;

- Motor: this class represents a single motor and its functionalities such as turning ON/OFF the LED, moving to a position, setting the speed, setting the torque etc.;
- MotorSet: it is a set of motors that abstract a robot, there is an object of this class for each robot we have, a page is executed by a MotorSet, but only if the number of motors of both are equal;

The schematic of the software encapsulation used to generate an open loop walking is shown in Fig. 4, the Walking class is used to generate a page that contains several poses, and then the MotorSet class, that contains a group of Motors and a Channel, interprets this page and sends the desired positions to the engines.

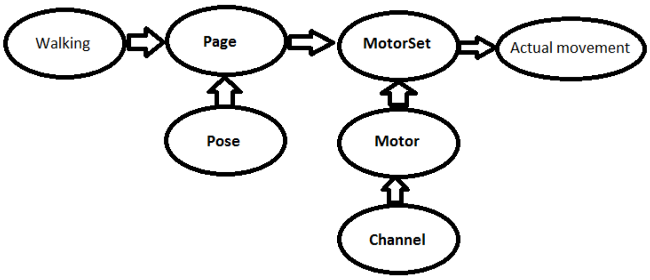

Figure 4. Open loop walk software schematic.

But to generate the controlled walking that was discussed in this article a more elaborated software is needed, the sensor must also be processed and its data applied to the walking chain.

In order to develop a more optimized software, it was separated in 2 process one that handles the gait calculation and execution, and another that collects and treats the information provided by the sensor, and within those processes there are some threads to make it even faster.

The optimization of the software doesn't show to be a must do when its running on a regular computer such as laptops and PCs, but when its processed on an embedded system with lower processing capacities, such as the one EVA has it is necessary to optimize as much as you can.

Since the software was divided in two processes it is need to send data between them, and for that an open-source platform name Yet Another Robot Platform (YARP) [9-10] was utilized to communicate both.

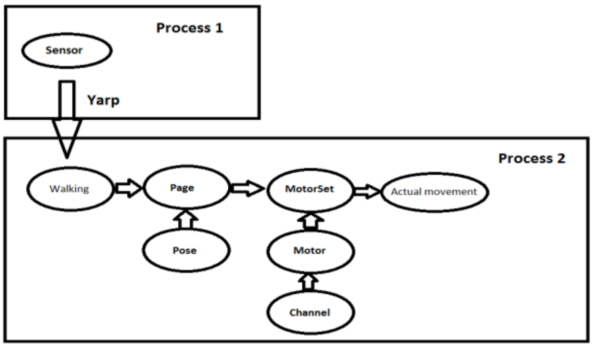

Figure 5. Schematic of the processes communication.

The schematic shown in Fig. 5 represents the communication between the processes, showing that the data processed and treated by the sensor in the process 1 is sent to the Walking class in the process 2. And with that the second process recalculates and execute the walking with the hip pitch offset corrected, which is exactly what was desired to happen given the control system demonstrated in the section above.

III. Conclusion

The task to make a humanoid robot walk like a human is not the simplest one, since humans are very complex. The perfection of the human walk cannot be imitated on its full integrity, but with some simplifications and approaches the robot gait can be accepted as a human like.

With a proper kinematics it is possible to generate a really good walk that would work in a robot with no mass, no frictions, and no gaps in the joints, in an ideal robot. But since that's not the practical case, to improve the biped walking a control system must be applied to make it more stable.

As observed in the experiments conducted by EDROM, a simple proportional control system showed to have a huge difference in the stability of a humanoid robot during the walk. Making a movement calculated by the kinematics that wouldn't work in different terrains, work for several ones and even in the case that disturbances are applied to the robot.

But developing the kinematics and control system is not all that is needed to make a robot walk, all the theory developed must be transformed in lines of code to make the robot know what to do.

Since EVA is an autonomous robot it must have its own processor, embedded system, and because of that a software cannot be developed indiscriminately. The speed of execution and computational cost has a high value since an embedded system is not that powerful.

As shown in this article with another simple solution, dividing the software in 2 processes, the processing cost can be optimized.

Acknowledgment

The authors thank all the supporters and sponsors. This work was supported by the FEMEC/UFU, FAPEMIG, and it was sponsored by Radix, Central Máquinas and CNPq.

References

[1] RoboCup. Available: http://www.robocup.org/
[2] DARwIn-OP: Available: http://support.robotis.com/
[3] Gait dependent parameters are described at the DARwIn-OP website (http://support.robotis.com/en/product/darwin-op/development/tools/walking_tuner.htm)
[4]M. Vukobratovic and D. Juricic, "Contribution to the synthesis of biped gait," IEEE Trans. Biomed. Eng., vol. 16, no. 1, pp. 1–6, 1969.
[5]M. Vukobratovic and B. Borovac in "Zero-Moment Point – thirthy five years of its Life" International Journal of Humanoid Robotics, Vol. 1, No. 1 (2004) 157–173.

[6] Sebastian O.H Madgwick. (2010, April). An efficient orientation filter for inertial and inertial/magnetic sensor arrays. Available: http://www.x-io.co.uk/res/doc/madgwick_internal_report.pdf
[7] Norman S. Nise, *Control Systems Engineering*, 6th ed, California State Polytechnic University.
[8] K. Ogata, *Modern Control Engineering*, 5th ed.
[9] Yet Another Robot Platform. Avalable: http://www.yarp.it/
[10]YARP: Yet Another Robot Platform, Giorgio Metta, Paul Fitzpatrick, and Lorenzo Natale, International Journal of Advanced Robotic Systems, 3(1):43-48, 2006.