www.ijemr.net ISSN (ONLINE): 2250-0758, ISSN (PRINT): 2394-6962

Volume-8, Issue-3, June 2018

International Journal of Engineering and Management Research

Page Number: 66-70

DOI: doi.org/10.31033/ijemr.8.3.9

Developing a Humanoid Robot Platform

Nuralem Abizov[1], Jia Yuan - Huang[2] and Fei Gao[3]
[1]MS.c Student, Department of Computer Science, Hanghzou Dianzi University, CHINA
[2]BS.c Student, Department of Mechanical Engineering, Hanghzou Dianzi University, CHINA
[3]Professor, Department of Computer Science, Hanghzou Dianzi University, CHINA

[1]Corresponding Author: nuralem.hdu@yandex.ru

ABSTRACT

This paper is focused on developing a platform that helps researchers to create verify and implement their machine learning algorithms to a humanoid robot in real environment. The presented platform is durable, easy to fix, upgrade, fast to assemble and cheap. Also, using this platform we present an approach that solves a humanoid balancing problem, which uses only fully connected neural network as a basic idea for real time balancing. The method consists of 3 main conditions: 1) using different types of sensors detect the current position of the body and generate the input information for the neural network, 2) using fully connected neural network produce the correct output, 3) using servo-motors make movements that will change the current position to the new one. During field test the humanoid robot can balance on the moving platform that tilts up to 10 degrees to any direction. Finally, we have shown that using our platform we can do research and compare different neural networks in similar conditions which can be important for the researchers to do analyses in machine learning and robotics.

Keywords— Fully-Connected Neural Network, Humanoid Balancing, Humanoid Robot, Machine Learning, PLEN2, PID

I. INTRODUCTION

Balancing is one of the main problem in robotics, especially for the humanoid robots with many free movement parts [1][2]. Nowadays, the most popular and simple mathematical algorithm for balancing systems is PID [3] (PD or P), that is widely used in industrial control systems and a variety of robot's applications requiring continuously modulated control. The proms of this method are high stability and low memory usage. Cons: for the simple systems the coefficients are not so hard to find, but for the complex systems it is going to be nontrivial task [4][5]. For this reason, in this paper we tried to focus on machine learning, because it can do most of the optimization work for you. Another problem in robotics – hardware part. Developing a platform for this kind of task is not easy – you should think about the price, time and reliability of the whole system. High maintainability during the research is also important.

Our platform has two main parts – robot and moving plat-form. With the help of 3D-printing we have many open source humanoid robots that can be easily printed at home without any problem. For our platform we are using an open source project – Plen2 [6] (Figure 1). Plen2 has 18 joints which means that it is highly maneuverable, also it is very light (21.16 oz) and tall enough (7.87 inch). It can be easily printed at any 3D-printer, also we can easily change it, modify and fix hardware issues during the research with low time and money cost.

Figure 1: PLEN2 - the world's first printable open-source humanoid

II. PLATFORM DETAILS AND PROBLEM FORMULATION

In our platform we have many sensors to determine the robot's body position and orientation. It is very important to collect the accurate data from the environment in real time with lowest possible delay. This section describes these sensors and the hardware part that have been used to control the robot and get the data from the environment. In addition, we formulated the balancing problem that we try to solve.

Hardware specification

The Figure 2 shows the main parts of the designed platform

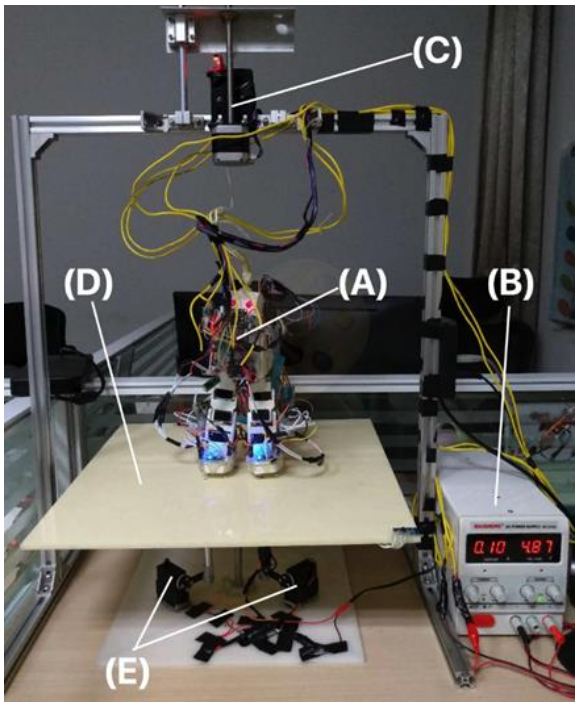

Figure 2: The main parts of the platform

A) ***Humanoid robot:*** Plen2 has a lot of sensors that have been used for body control and stabilization. Each of the robot's sensors provide position and force measurements.

B) ***Power supply:*** The robot doesn't have internal power sup-ply that's why our system uses external power supply for the robot.

C) ***Safety system:*** This system tries to minimize the probability of breaking the robot when it falls down. Also, after the robot falls down it keeps the start position of the whole body.

D) ***Moving platform:*** The platform was designed to move in any direction up to 30 degrees. It controls by a computer via USB interface and is also has some gyroscope sensors and a microcontroller inside to get absolute position and control the servo-motors of the platform.

E) ***Servo-motors:*** We are using two motors that can move the whole platform in each direction. The detailed architecture of the robot is described in Figure 3.

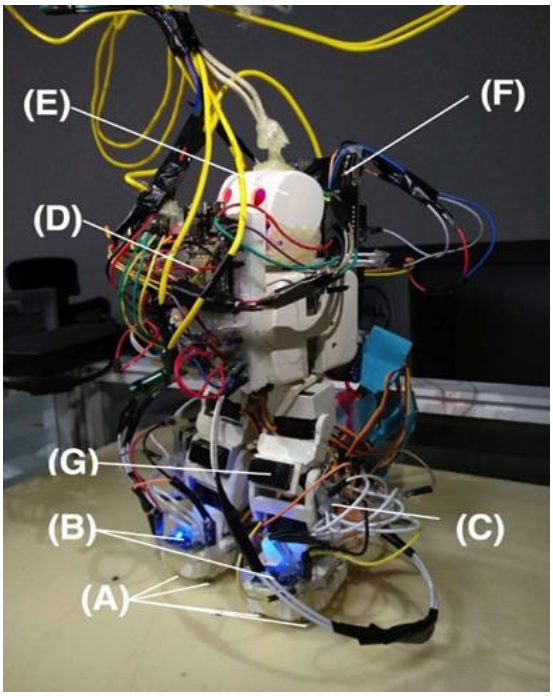

Figure 3: PLEN2 with different types of sensors

A) ***Force feedback sensors:*** Plen2 has 8 force-sensing resistors that allow him to get force data from ground. It is very important data that we get from the environment that helps the robot statically balance and dynamically adapt to the moving platform. The measurement range is from 1 gram to 1000 grams that makes possible to feel slightly changes on the ground and adapt to it.

B) ***Gyroscope on each leg:*** To get raw data from each foot we are using MPU-6050. It gives 3-axis gyroscope data and 3-axis accelerometer data.

C) ***Microcontrollers (ATmega328P)*****:** After getting the raw data from the MPU-6050 and force-sensing resistors, using microcontrollers we collect data through a software noise filter then make some calculations to get the absolute position of each leg relatively to the gravity vector and force feedback data without any noise.

D) ***ESP32*****:** This is the main part of the robot where our neural network is running. It has dual-core 32-bit LX6 micro-processor, that is operating at 240 MHz although it has a lot of communication modules that allows to connect the computer via WI-FI, Bluetooth and USB.

E) ***Gyroscope in the head:*** For balancing problem we need to know the absolute position of the whole body. Collecting this data and the data from the force-sensing resistors we can define balanced and non-balanced states of the body.

F) ***Microcontroller (ATmega2560):*** 2560 has a lot of serial communication ports that allow to collect data from other microcontrollers and send it to the ESP32.

G) ***Servo-motors:*** ES08MA2 – compact light (12 gram) servo with 2.0 kg.cm of torque. Plen2 uses servo motors for movements.

Hardware diagram

Our platform is designed to be a modular platform where each block is independent and has own task to solve. Robot collects data from 26 sensors and proceed it using 4 micro-controllers and finally send it to the ESP32 that runs neural network and produce the output for the servo-motors (Figure 4).

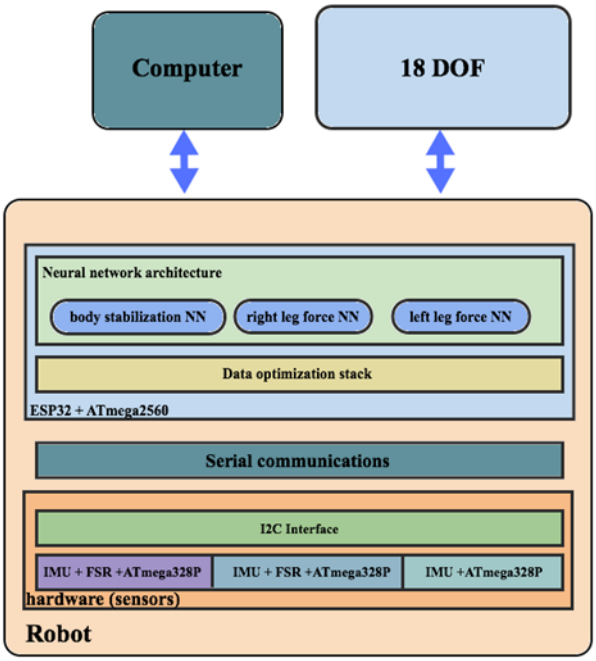

Figure 4: Hardware block diagram

Problem formulation

To solve the balancing problem, we need to deal with next problems:

- COM – Center of mass. Most researchers do mathematical calculation in the simulated model to define the COM of the robot model [2][7], because the environment is fully pre-defined, but for the real physical environment this is not so easy to do.
- Base of Support. On the moving platform robot deal with dynamically changing ground. To keep balancing robot should adapt the base of support and continuously keep it reliable for the algorithms while balancing.
- Gravity vector. G-vector that goes from COM should always points to a base of support. Going out of base means that robot will be unbalanced which is the main problem in humanoid balancing systems.

The goal of the whole system is direct balanced position of the humanoid robot with respect to gravity and using the force feedback sensors, gyroscope, accelerometer information and pre-trained neural network for controlling the robot body, which will produce some movements to achieve a correct stable and balanced position.

III. THE ARCHITECTURE OF THE NEURAL NETWORK

The platform uses fully connected neural network with different parameters that works together to make the robot balancing on the moving platform. The networks described here is a feed-forward backpropagation network, which is perhaps the most common type (Figure 5). In the feed-forward network we're building here, the neurons are arranged in three layers called the input, hidden, and output layers. All the neurons in one layer are connected to all the neurons in the next layer. For the input layer, we use data from the IMU and FSR sensors. We have 18 output nodes, each of them produces a controlling signal.

To train the networks, we have defined a training sets with appropriate truth table values in the Input and Target arrays. For training the NN we also have defined different situations, that our robot can run into.

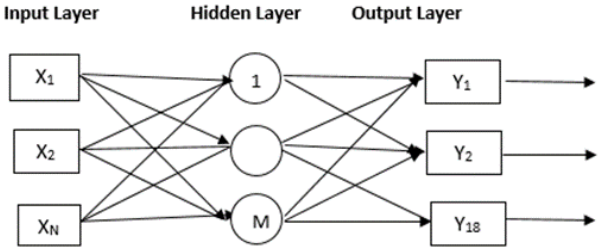

Figure 5: The architecture of the fully-connected neural network

We also have defined the learning rate = 0.3, momentum = 0.9 (how much this value affects to the results of the previous and current iteration) and Success parameter = 0.0014 (a threshold for error).

As a general idea, hidden layer learning rate and momentum – they all optimize the network for learning speed and effectiveness, to minimizing pitfalls that are encountered in our design. The Success, is a threshold of error which consider training set into learned and not-learned states.

The logic of the balancing algorithm:

1 step. Every 20 milliseconds the robot gets raw data from IMU sensors and force-sensitive resistors.

ISSN (ONLINE): 2250-0758, ISSN (PRINT): 2394-6962

2 step. Using microcontrollers on each leg the robot reduce noise from IMU sensors and resistors data then calculate the absolute position of each leg.
3 step. Algorithms adapt data for using it in the neural networks.
4 step. All data goes into Input layer of the neural networks.
5 step. Neural networks produce the output signals, that control the servos.

IV. EXPERIMENTAL RESULTS

The neural networks described in this paper were tested on the moving platform, that can perform any movements. The goal is direct balanced position of the body with respect to gravity, also to make legs stable and change their positions according to the FSR sensors to get maximum force contact with the ground. For the test we made different architectures of the neural network. The only thing we change is the number of nodes in hidden layer (Table 1).

	Pattern 1	Pattern 2	Pattern 3
Hidden nodes	5	10	3

Table 1: Different numbers of hidden nodes in each pattern

Our experiment shows that different number of hidden nodes can produce interesting results (Figure 6, Figure 7, Figure 8, Table 2). All three tests the platform produced yes-motion. Head pattern shows the gyroscope data from the head (body), platform pattern shows the data from the plat-form gyroscope. As you can see Pattern 1 has the best balancing performance results, because it has lowest wavering which makes the system more stable, but according to the Table 2 the mean and deviation values are higher than in Pattern 2. Adding more hidden cells will not make the system more stable, over against, it adds more fluctuations on it. Finally, reducing the number of hidden cells will make the system unbalanced. In Pattern 3 we can see that the robot fell down.

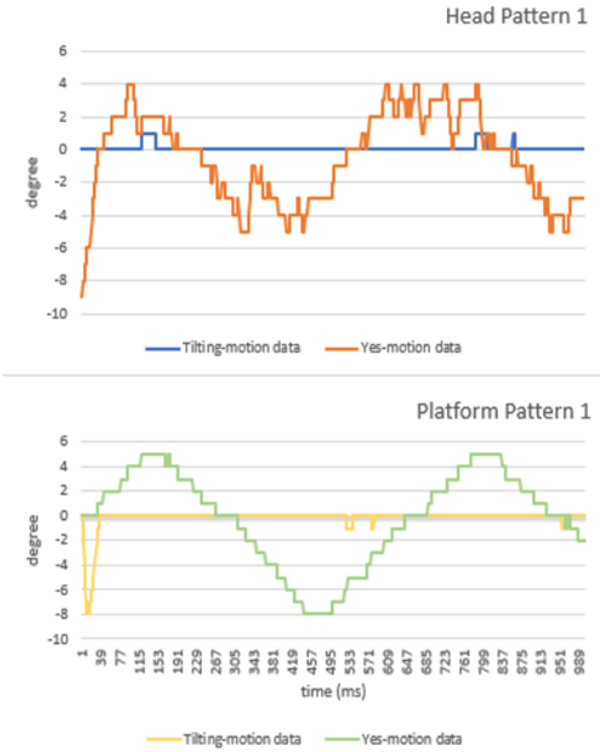

Figure 6: Head Pattern 1. The performance diagrams of the 5 hidden nodes NN architecture

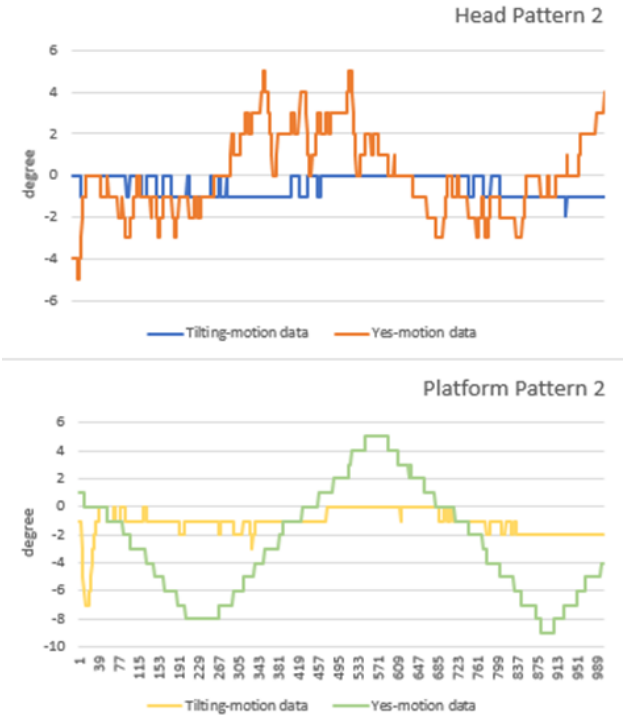

Figure 7: Head Pattern 2. The performance diagrams of the 10 hidden nodes NN architecture

Figure 8: Head Pattern 3. The performance diagrams of the 3 hidden nodes NN architecture

	Pattern 1	Pattern 2	Pattern 3
Mean	-0.525	0.006	4.218
Standard deviation	2,736	1,934	26,569

Table 2: The mean and standard deviation table

V. CONCLUSION AND FUTURE WORK

We have shown that using our platform we can analyze different variations of neural networks and see the performance in real psychical world, in our case we proved that different numbers of hidden nodes in fully connected neural network can obtain different results. Also, we shown that using only pre-trained fully connected neural network we can achieve real time balancing in a complex environment with many parameters. An obvious next step is generating more data from the platform and use different neural network architectures to get some performance results and compare it with fully connected neural networks. All these will help to teach the robot how to walk like natural organism with some pre-defined rules and see how it will solve complex problems like efficient natural walking, going up/down stairs and keep balancing with external sweeping forces.

REFERENCES

[1] A. Takanishi, T. Takeya, H. Karaki, & I. Kato. (1990). A control method for dynamic biped walking under unknown external force. *IEEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications*, *29*(6), 795–801.

[2] Y. F. Zheng & J. Shen. (1990). Gait Synthesis for the SD-2 biped robot to climb sloping surface. *IEEE Transactions on Robotics and Automation*, *6*(1), 86-96.

[3] Kiam Heong Ang, G. Chong, & Yun Li. (2005). PID control system analysis, design, and technology. IEEE *Transactions on Control Systems Technology*, *13*, (4), 559-576.

[4] Dalei Pan, Feng Gao, Yunjie Miao, & Rui Cao. (2015). Co-simulation research of a novel exoskeleton-human robot system on humanoid gaits with fuzzy-PID/PID algorithms. *Advances in Engineering Software*, *79*, 36-46. Available at: https://www.sciencedirect.com/science/article/pii/S0965997814001483

[5] A. Macchietto, V. Zordan, & C. R. Shelton. (2009). Momentum control for balance. *ACM Transactions on Graphics*, *28(*3), 1-8.

[6] M. Vukobratovic. (2004). Zero-moment poin- Thirty five years of its life. *International Journal of Humanoid Robotics*, *1*(1), 157-173.

[7] W. T. Miller. (1994). Real-time neural network control of a biped walking robot. *IEEE Control Systems*, *14*(1), 41-48.

[8] Q. Huang, K. Yokoi, & S. Kajita, et al. (2001, June). Planning walking patterns for a biped robot. *IEEE Transactions on Robotics and Automation*, *17*(3), 280-289.