A Neutral Emulation of the PID Controller for the Positioning Inverted Pendulum System

LOPE BEN PORQUIS

Abstract

A neural network is one of the artificial intelligence solutions whose ability to clone transfer functions can be used to solve complex real life problems that can not be modeled using mathematical equations. This project is about a neural clone of a simple non-optimal proportional-integral-derivative (PID) controller that is used to control a position–based inverted pendulum system. The neuro-controller is allowed to adapt the characteristics of the PID controller and it will be used as an alternative controller for the inverted pendulum system. Results showed that the neural net was able to clone the non-optimal PID controller and exhibit a more stable characteristic than the latter. The neuro-PID controller was able to control the inverted pendulum system over prolonged period of operation without going unstable.

Key Words: Artificial Intelligence, Neural networks, control system, PID controller, neuro-PID controller, Position-based inverted pendulum.

LOPE BEN C. PORQUIS is Assistant Professor I, Department of Electronics and Communication Engineering, College of Engineering MSU-IIT. His interests are robotics and hardware development. He is from is from Iligan City.

I. Introduction

A neural network is one of the artificial intelligence techniques used to solve complex real life problems. Its ability to clone a transfer function made them useful in solving complex systems that can not be modeled using mathematical equations. The neural emulation of a simple PID controller demonstrates the ability of the neural network to be used as a system controller. The former control system was a PID controller; however its parameters are not optimized to control the inverted pendulum system. The way it controls produces too much vibration which made the overall system unstable. The neuro-PID controller will be used as an alternative controller for the inverted pendulum system. The inverted pendulum is fundamentally a servo positioning system attached with a pendulum arm. A reference position must be set in order for the pendulum arm to follow.

II. Significance

The inverted pendulum is a popular reference problem for non-linear controllers such as fuzzy logic, neuro-controllers, PID, etc. Designing one could lead to a stepping-stone toward developing more powerful non-linear controllers. The neuro-PID controller simply demonstrates the capability of a neural network to clone and control an inverted pendulum system.

III. Objective

The objective of this project is to make a neural network emulate the capabilities of a PID controller and observe its performance.

IV. System Design and Implementation

This project uses MATLAB® / Simulink® configured in real time mode using the Real Time Workshop Toolbox.

The first stage of the project is to design the model of the pendulum system. Such model will be used as a basis for fabricating a physical pendulum system. The pendulum system consists of a DC motor coupled with a continuous turn potentiometer and a pendulum arm is attached to the shaft of the motor. The design of the PID controller is the next step of the process. The values of the proportional, integral, and derivative coefficients are obtained by trial and error, thus their values are expected to be non-optimal.

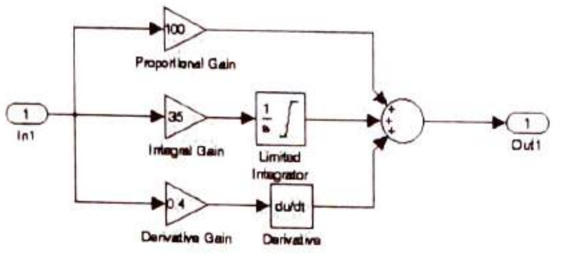

Figure 1. Simulink® Implementation of the PID Controller.

After designing the PID controller, the neural network controller was developed. The neuro-controller is based on a feed-forward neural network architecture. Built in MATLAB functions of the neural network toolbox were utilized in designing the neuro-controller. The neural network was interfaced to an AD512 data acquisition card so that it can acquire real world data.

The neural network is trained off-line. The training values for the neural network are obtained by feeding some random input values to the PID controller and collect the corresponding output data. The collected input and output data are used as the input and target parameters for the neural network, respectively. This data is used to train the neural network. Figure 2 shows how the training parameters for the neural network were collected.

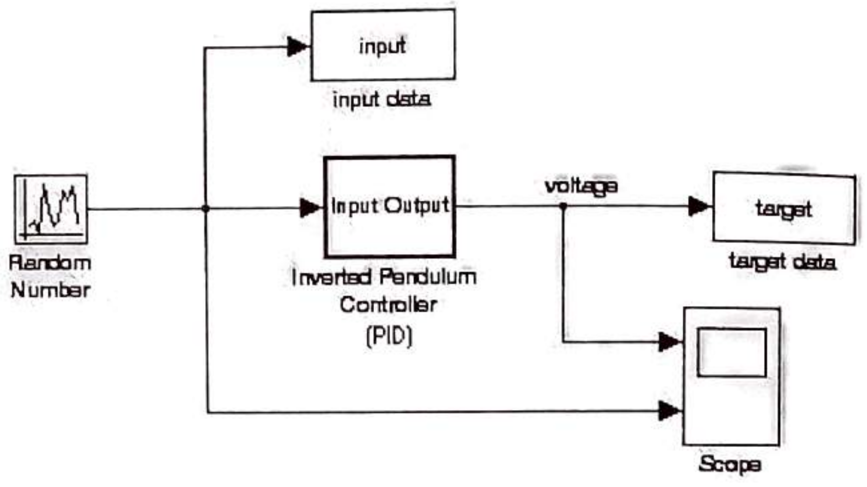

Figure 2. Extraction of the training parameters for the Inverted Pendulum System.

The PID Control System

The proportional, integral and derivative coefficients of the PID controller in Figure 2 were taken by trial and error. These values were not optimal thus making the controller operate in a slightly unstable condition. Figure 3 below shows the implementation of PID controller for the inverted pendulum system. The control system is connected to the AD512 data acquisition card so that it can control the physical inverted pendulum system in real time.

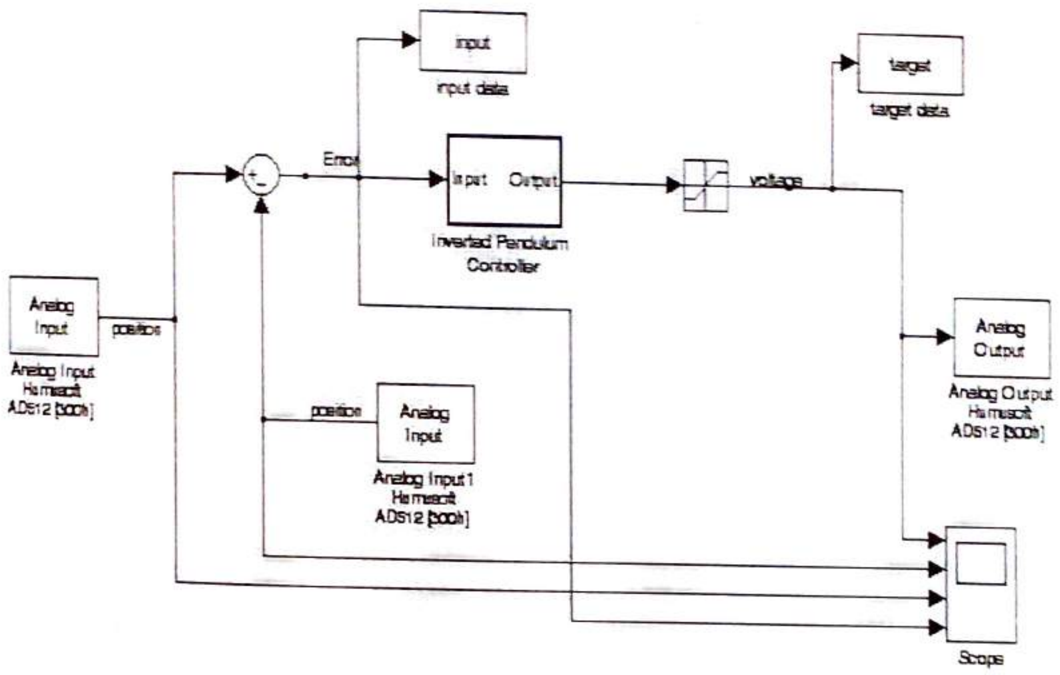
input
input data
target
target data
Error
Input Output
Inverted Pendulum Controller
voltage
Analog Input
position
Analog Input Humusoft AD512 [boot]
position
Analog Input
Analog Input1 Humusoft AD512 [boot]
Analog Output
Analog Output Humusoft AD512 [boot]
Scope

Figure 3. Implementation of the PID controller in Real Time Workshop.

The Analog Input of the figure above is the reference position port. This port is connected to an external potentiometer for the user to control the position of the pendulum arm. The Analog Input 1 is the feedback port of the system. This port is connected to the continuous turn potentiometer that is coupled to the DC motor. This provides feedback information to the PID controller regarding the exact position of the pendulum arm. The Analog Output is the actuation port. It is linked to the digital to analog converter of the data acquisition card. It provides an actuation signal to the DC motor to move the pendulum arm to its target position.

The Neural Control System

The implementation of the neural controller was initially developed as a MATLAB® code. The training data extracted from the PID controller was used to train the neural net after it was created. The following are the MATLAB® code for creating and training the network.

```
net = newff(minmax(transpose(input)), [500 1], {'tansig' 'purelin'});
net.trainParam.epochs = 1000;
net.trainParam.goal = 1e-2;
net.trainParam.show = 25;
net = init(net);
net = train(net,transpose(input),transpose(target));
```

Code 1. Implementation and Training of the neural network.

The neural net architecture was based on the new feed-forward network. The maximum training iterations was set to 1000 and error goal was set to 0.001. This neural network is only available to MATLAB® but it can be converted into a Simulink® block for real-time simulation with Real Time Workshop. The process of converting the neural net into a Simulink® block is very trivial.

The code *gensim(net,-1)* was used and is typed in the MATLAB® command window. A Simulink® environment will pop-up containing the neural network block. The neural network block can be copied to the neuro-PID control system.

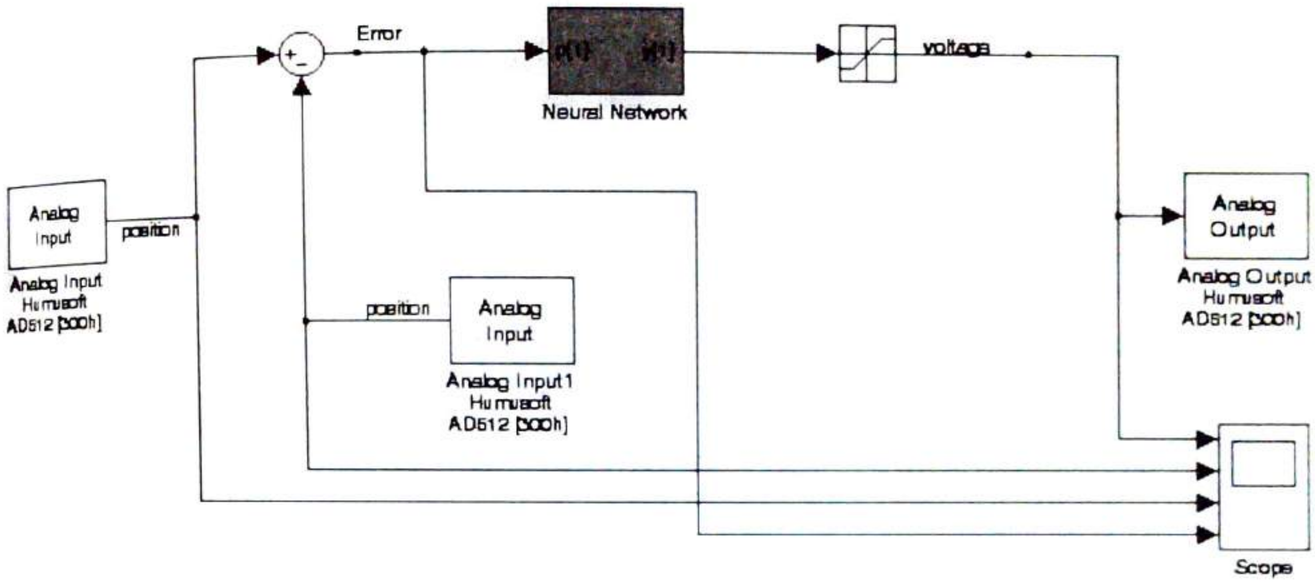
Error
Neural Network
voltage
Analog Input
position
Analog Input Humusoft AD512 [300h]
position
Analog Input
Analog Input1 Humusoft AD512 [300h]
Analog Output
Analog Output Humusoft AD512 [300h]
Scope

Figure 4. The neuro-PID control system.

V. Results

The non-optimized PID control system was allowed to control the inverter pendulum. It produced rough vibrations during its operation. Although it was able to move the pendulum arm to its desired position but at its settling point the pendulum arm starts to oscillate. If it is allowed to oscillate over time the pendulum system becomes more unstable and eventually the pendulum arm will rotate around rapidly. The optimization of a PID controller is yet a difficult problem. Several techniques can be employed however, it involves complex mathematical equations.

During the test of the neuro-PID controller, it was discovered that the pendulum arm still oscillate but its oscillations are more stable compared to the non-optimal PID controller. If allowed to oscillate overtime, the pendulum arm does increase its instability. The range of oscillation is more limited and smooth. Although the neuro-controller is still unstable, its characteristics were cloned from the non-optimal PID controller, its instability is more confined and does not increase over time.

The occurrence of the oscillations is due to the settling point, the controller must find a way to keep the pendulum arm as close to the settling point as possible. In an optimal PID controller such oscillations won't happen. Since we based the neuro-controller from a non-optimized PID, the oscillation characteristic of the PID controller was also cloned by the neural net as observed during evaluation.

VI. Conclusion

It was observed in this project that a neural network is capable of emulating the characteristic of a PID controller. However due to the non-optimal selection of the PID values, the neuro-PID controller also behaved similar to its original counter part. The significant finding in this project showed that the neuro-controller was able to control the inverted pendulum system over time without getting unstable as compared to the non-optimized PID controller whose control ability gets unstable as the system operates over time. It showed that the neural network did not exactly clone the non-optimal PID controller and only its important characteristics were inherited. This project can be a good reference for developing more robust and powerful controllers in the future.

VII. References

Demuth, H., Beale, M., Neural Network Toolbox User's Guide, Version 3. Natic, MA. The MathWorks Inc. 1998.

MATLAB®. Getting Started with MATLAB, Version 5. Natic, MA. The MathWorks Inc. 1997.

MATLAB Runtime Server Application Developer's Guide. Natic, MA. The MathWorks Inc. 2001.

Savant, et.al., Electronic Design Circuits and Systems, Redwood, CA. Benjamin/Cummings Publishing Company, Inc. 1991.

Simulink®. Writing S-Functions, Version 4. Natic, MA. The MathWorks Inc. 2001.