International Journal of Electrical and Electronics Engineers
Vol. No. 9, Issue No. 01, January-June 2017
ISSN (O) 2321-2055
ISSN (P) 2321-2045

NEUROBOT: A NEURAL NETWORKROBOT BASED ON RASPBERRY PI

Sarita Chauhan[1],Archit Ranka[2],Niranjan Khoiwal[3],
Rishabh Sharma[4],Rohit Tripathi[5]

[1,2,3,4,5]*ECE Department, MLVTEC Bhilwara (INDIA)*

ABSTRACT

This research develops a Neural Network Robot with Raspberry Pi model, to predict the key Artificial Intelligence outcome "User Satisfaction" using causal factors present during an implementation as predictors.Data for training and testing the models was from across section of firms that had implemented this. In today's era of automation everyone wants to reduce manual efforts so as to improve accuracy and efficiency of real time processes. Recent Researches in Artificial Intelligence using Neural Network uses various algorithms to train the robot to perform multi-purpose tasks as physical condition changes. Automation can be best achieved if we can make judicious use of Internet with everything i.e. INTERNET OF THINGS. Raspberry Pi is currently the best option to have easy Remote Access from anywhere just assigning a Network I.P and have control over every hardware. Recent research showed Graphical Interface along with Command line can be achieved like making a separate GUI which shows all processes taking place in hardware on screen simultaneously in a easy and attractive way.

Keywords: ANN, Failure Detection, GUI,Internet Of Things , Prediction,Raspberry Pi

I.INTRODUCTION

In the present study the data from the earlier study was used to develop predictive models for ANN with Raspberry Pi and outcomes measured in terms of User Satisfaction.Three processing techniques: Python, MATLAB, Artificial Neural Networks (ANN) and OpenCV were tested. Of the three Python was found to be significant.

This paper is organized as follows: The literature review and establishes the need and relevance of this research work. Outlines the method used in the research. This section also explains different techniques with specific emphasis on Neural Logic. The results of the modeling and compares the results of the various techniques used. The paper with the direction for continuing research.

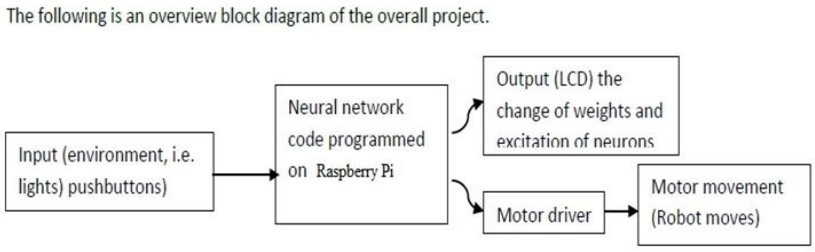

Figure1:ConceptualModel

International Journal of Electrical and Electronics Engineers
Vol. No. 9, Issue No. 01, January-June 2017
ISSN (O) 2321-2055
ISSN (P) 2321-2045

Robotics technology is emerging at a rapid pace, offering new possibilities for automating tasks in many challenging applications, especially in space explorations, military operations, underwater missions, domestic services, and medical procedures. Particularly, in space exploration, robotic devices are formally known as planetary rovers or simply rovers and they are aimed at conducting physical analysis of planetary terrains and astronomical bodies, and collecting data about air pressure, climate, temperature, wind, and other atmospheric phenomena surrounding the landing site. Basically, rovers can be autonomous capable of operating with little or no assistance from ground control or they can be remotely controlled from earth ground stations called RCC short for Remote Collaboration Center.

In essence, the movement of autonomous rovers is not directed by human operators; instead, it is controlled by complex algorithms that allow the rover to traverse paths on multiple terrains while avoiding obstacles and path errors. This capability is more formally known as path-planning in which a rover or any robotic vehicle can perform terrain analysis and select the safest route to travel across. The rover can then proceed towards the goal location over the selected trajectory while avoiding obstacles without previous knowledge of their existence.

This paper proposes a path-planning solution for autonomous robotic planetary rover systems based on artificial neural network (ANN). The proposed neural network is multi-layer consisting of three consecutive layers: an input, a hidden, and an output layer. The input layer is made out of two neurons that are fed by the rover's sensors which are designed to detect obstacles of any size and shape. The hidden layer is made out of three neurons and its purpose is to read input data and multiply them by a certain weight and then forward the results to the next layer. The output layer is made out of two neurons that are directly linked to the rover's motors which control its movement and its mechanical operation. The proposed ANN uses a mix of activation functions including Sigmoid for the hidden neurons and linear for the output neurons. Moreover, the model employs a supervised learning approach using the back-propagation algorithm to train the network in offline mode.

The proposed artificial neural network is meant to allow the rover system selects the best path through any given ground by predicting the existing obstacles along the path and the harsh structure of the landing terrain. This would allow the rover to navigate autonomously and safely toward its goal location and complete its designed task.

II.METHODOLOGY

Our project consists of an elementary neuron network that uses ANN and Hebbian Learning to train a robot to respond to the change in various physical conditions. First, we decided to take up a task challenging enough involving electronics along with some programming and involving as many features of Raspberry Pi with live streaming with graphical interface. We found this project very innovative and we found this task interesting as well as challenging enough.

Through an earlier empirical study data was collected from a cross section of around 60 organizations and 156 respondents representing three user cohorts: Strategic, Technical & Operational responded to a pre-tested and validated (for content validity) structured questionnaire.

Artificial neural networks or ANNs for short are very influential brain-inspired computational models, which have been employed in various areas such as computing, medicine, engineering, economics, and many others. ANNs are composed of a certain number of simple computational elements called neurons, organized into a

International Journal of Electrical and Electronics Engineers
Vol. No. 9, Issue No. 01, January-June 2017
ISSN (O) 2321-2055
ISSN (P) 2321-2045

structured graph topology made out of several consecutive layers and immensely interconnected through a series of links called the synaptic weights. Synaptic weights are often associated with variable numerical values that can be adapted so as to allow the ANN to change its behavior based on the problem being tackled.

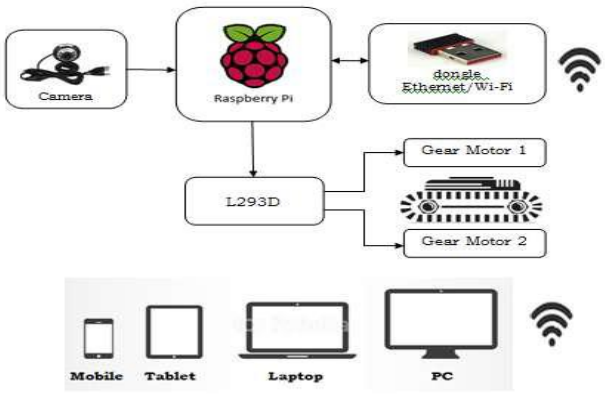

Figure 2: Schematic Layout

Training an artificial neural network is usually done by feeding the network's input with a pattern to learn. The network then transmits the pattern through its weights and neurons until it generates a final output value. Afterwards, a training or a learning algorithm compares the produced output value to an expected output and if the error range is high, the algorithm marginally alters the network's weights so that if the same pattern is fed again to the network, the output error would be smaller than the previous iteration. This process gets repeated for many cycles called epochs using different set of input patterns until the network produces acceptable outputs for all inputs. This learning progression allows the network to identify many patterns and further generalize to new and unseen patterns. Such type of training is called supervised learning which uses classified pattern information to train the network in offline mode. On the other hand, there exists what so called the unsupervised learning which uses only minimum information without pre-classification to train the network while being in online mode. Some of the most successful supervised learning approaches are feed-forward and back-propagation; while, the most successful unsupervised learning approaches are the Hebbian and the competitive learning rule.

III. ALGORITHMS

III.(a) The Back-Propagation Algorithm

The proposed ANN model is trained through a supervised learning approach using the back-propagation algorithm. The back-propagation algorithm comprises two passes: A forward pass which propagates the input data in the forward direction from the input layer to output layer of the network. The pass eventually ends up by generating an output value and computing an error value while leaving synaptic weights intact. In effect, the error is calculated by subtracting the desired output from the actual output just generated. If the error is within an acceptable range, then the network is trained with new set of input data; otherwise, a backward pass is executed. The backward pass is a reverse pass which propagates the error signal backward through the network

International Journal of Electrical and Electronics Engineers
Vol. No. 9, Issue No. 01, January-June 2017
ISSN (O) 2321-2055
ISSN (P) 2321-2045

layers so as to update the synaptic weights of the network. The different steps of the back-propagation algorithm can be summarized as follows:

[i] Feed the network with an input vector and a corresponding desired output vector.

[ii] Calculate the output of the network using forward pass.

[iii] Calculate the output error signal.

[iv] If error is within an acceptable range, move to the next input vector, otherwise go backward and update the weights of the network.

[v] Keep repeating the above steps until all input vectors are consumed

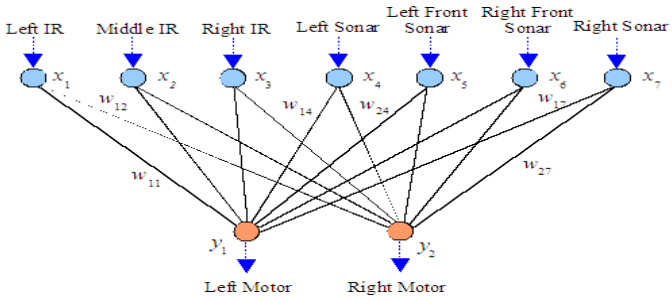

Figure 3: Back Propogation Algorithm

Additionally, and in order to attain more accurate results, the back-propagation algorithm was fine-tuned with extra parameters whose purpose is to regulate and add more accuracy to the learning process by shifting the activation function to the left or to the right. The controlling parameters are listed below:

• Biases: *bh* and *bo*

• Learning Parameter: η

III.(b) Computing the Back-Propagation Algorithm

Computationally, the proposed model is governed by the following steps and mathematical equations:

i. Perform the forward propagation and calculate the output signal using *(X1 * W1)+(X2 * W2)+...+(Xi * Wi)+...+ (Xn * Wn)*

ii. Calculate the Sigmoid activation function *(1 / 1+e–input)* for the hidden neurons and the linear activation function *(y = v)* for the output neurons.

iii. Calculate the error using *error = desired output – actual output*

iv. Perform the back propagation algorithm using the following equations to update the weights of the network:

Case1: For the output neurons:

*Weight (n+1) = Weight (n) + □[ΔWeight(n-1)] + (N * Output (previous neuron) * error)*

Case2: For hidden neurons:

*Weight (n+1) = Weight (n) + □[ΔWeight(n-1)] + [N * Output (previous neuron) * Output (this current hidden neuron) * (1-Output (this current hidden neuron)) * Σk errork * weight kj]*

International Journal of Electrical and Electronics Engineers
Vol. No. 9, Issue No. 01, January-June 2017
ISSN (O) 2321-2055
ISSN (P) 2321-2045

III.(c) The Learning Process

Although the back-propagation algorithm looks simple, computing it is quite an intensive task as it requires a series of arithmetic operations executed for hundreds of iterations. Below are the various calculations required to train the proposed network using the back-propagation algorithm.

Initial Weights:

W00 = 0.17 ; W01 = 0.33 ; W02 = 0.1 ; W10 = 0.3 ; W11 = 0.71 ; W12 = 0.21 ; W20 = 0.15 ; W21 = 0.43 ; W22 = 0.69

W00 = 0.11 ; W01 = 0.03 ; W02 = 0.52 ; W03 = 0.41 ; W10 = 0.93 ; W11 = 0.14 ; W12 = 0.79 ; W13 = 0.66

Input Vector*:* [0 , 0]

Desired Output Vector*:* [1 , 1]

Learning parameter*:* 0.25

Biases*:* +1

Forward Pass:

Input of h0: (0*0.17) + (0*0.33) + (1*0.1) = 0.1

Output of h0: 1/ 1+exp-0.1 = 0.524

Input of h1: (0*0.3) + (0*0.71) + (1*0.21) = 0.21

Output of h1: 1/ 1+exp-0.21 = 0.552

Input of h2: (0*0.15) + (0*0.43) + (1*0.69) = 0.69

Output of h2: 1/ 1+exp-0.69 = 0.665

Input of O0: (output of h0 * 0.11) + (output of h1 * 0.03) + (output of h2 * 0.52) + (1 * 0.41) = (0.524*0.11) + (0.552*0.03) + (0.665*0.52) + (1*0.41) = 0.05764 + 0.01656 + 0.3458 + 0.41 = 0.83

Output of O0: 0.83 *1 = 0.83 (Linear activation Function)

Input of O1: = (0.524*0.93) + (0.552*0.14) + (0.665*0.79) + (1*0.66) = 0.48732 + 0.07728 + 0.52535 + 0.66 = 1.74995

Output of O1: = 1.74995 * 1 = 1.74995 (Linear activation Function)

Calculating Error for O0: desired – actual = 1 - 0.83 = 0.17

Calculating Error for O1: desired – actual = 1 - 1.74994 = 0. 74994

Back Propagation:

Starting with output Neuron Case 1 in the algorithm

For W00: weight (new) = weight (old) + (η * output(previous neuron) * error)= 0.11 + (0.25 * output (h0) * error (O0)) = 0.11 + (0.25 * 0.524*0.17) = 0.13227

For W10: 0.93 + (0.25 * 0.524 * error (O1))= 0.93 + (0.25 * 0.524 * (-0.74994)) = 0.83176 For W01: 0.03 + (0.25 * output (h1) * error (O0))= 0.03 + (0.25 * 0.552* 0.17) = 0.05346

Now dealing with the hidden Neurons Case 2 in the algorithm

For W00:

Weight (new) = weight (old) + [η *output (previous neuron) * output (this neuron) * (1 - output(this neuron)) * Σk errork * weghtkj]

Weight (n+1) = 0.17 + [0.25 * input x * output(h0) * (1-output(h0)) * (error (O0) * W10 + error (O1) * W10)] = 0.17 + [0.25 * 0 * 0.524 * (1- 0.524) *((0.17*0.13227) + (-0.74994 * 0.83176)) = 0.17 + 0 = 0.17 = W00(n+1)

International Journal of Electrical and Electronics Engineers
Vol. No. 9, Issue No. 01, January-June 2017
ISSN (O) 2321-2055
ISSN (P) 2321-2045

IV. RESULTS

The system has been implemented and the following results have been observed.

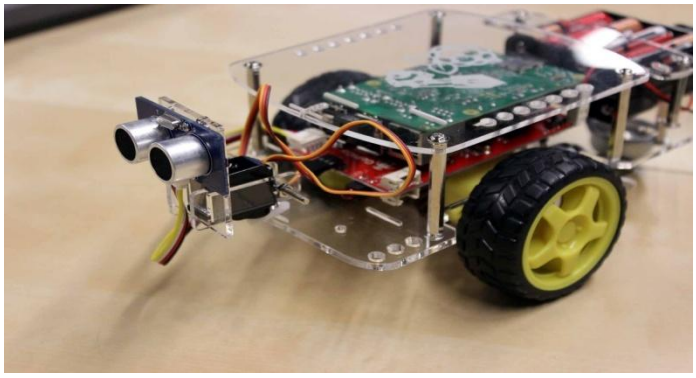

Figure 4: Structural Diagram

Another way to visualize the connections is to use the most excellent Matrix2PNG program from the Bioinformatics department at UBC. The program represents connection strengths with different colors as shown below:

Once learning is complete, we place our robot under the control of the network and set it loose among a collection of newly placed obstacles.

V. CONCLUSION AND FUTURE SCOPE

The system designed mainly aims at monitoring and surveillance at sensitive areas or unreachable areas .It will be helpful for the user who need surveillance of any place and this system provides the best results with low cost of deployment .This paper can be extended further by making the robot accessible via the internet.

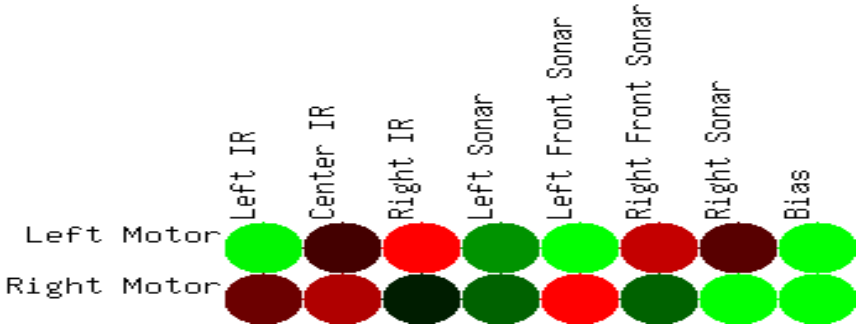

Figure 5: Color Layout

If user wants to use the location, they can use mapping algorithms to make it map the complete environment and then move autonomously after a certain periodic intervals to check everything. Also by giving it the ability to detect and recognize faces it can be made to alert us about any unknown person and take a snap of it and email us the same.

REFERENCES

[1.] Charles Severance, "Eben Upton: Raspberry Pi", Published by IEEE computer society, October 2013

[2.] Ikhankar, R.; Kuthe, V.; Ulabhaje, S.; Balpande, S.; Dhadwe, M., "Pibot: The raspberry pi controlled multi environment robot for surveillance & live streaming," in Industrial Instrumentation and Control (ICIC), 2015 International Conference on , vol. no., pp.1402-1405, 28-30 May 2015

[3.] Michael potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images", in Science direct Computer vision, graphics and image processing, vol., no., pp.1-29 October 1987

[4.] S Mukherjee and K. Das "A Novel Equation based Classifier for Detecting Human in Images", International Journal of Computer Applications (0975-8887), vol 72, no. 6, 2013

[5.] D. Yuan and R. Manduchi, "Dynamic Environment Exploration Using a Virtual White Cane", Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), University of California, Santa Cruz, pp. 1-7, 2005.

[6.] Amjed S. Al-Fahoum, Heba B. Al-Hmoud, and Ausaila A. Al-Fraihat, "A Smart Infrared Microcontroller-Based Blind Guidance System", Hindawi Transactions on Active and Passive Electronic Components,Vol.3, No.2, pp.1-7, June 2013.

[7.] S.Bharathi, A.Ramesh, S.Vivek and J. Vinoth Kumar, "Effective Navigation for Visually Impaired by Wearable Obstacle Avoidance System", International Journal of Power Control Signal and Computation(IJPCSC), Vol.3, No.1, pp. 51-53, January-March 2012.

[8.] Shraga Shovel, Iwan Ulrich, and Johann Borenstien, "NavBelt and the Guide Cane", IEEETransactions on Robotics& Automation, Vol.10, No.1, pp. 9-20, March 2003.

[9.] Pooja P. Gundewar and Hemant K. Abhyankar, "A Review on an Obstacle Detection in Navigation of Visually Impaired", International Organization of Scientific Research Journal of Engineering (IOSRJEN), Vol.3, No.1 pp. 01-06, January 2013.

[10.] Bhuvanesh Arasu and Senthil Kumaran, "Blind Man‟ s Artificial EYE An Innovative Idea to Help the Blind", Conference Proceeding of the International Journal of Engineering Development and Research(IJEDR), SRM University, Kattankulathur, pp.205-207, 2014.