RESEARCH PAPERS
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA
SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA

10.1515/rput-2016-0013 2016, Volume 24, Number 39

# ADDRESSING THE MOVEMENT OF A FREESCALE ROBOTIC CAR USING NEURAL NETWORK

Dušan HORVÁTH, Peter CUNINKA

SLOVAK UNIVERSITY OF TECHNOLOGY IN BRATISLAVA,
FACULTY OF MATERIALS SCIENCE AND TECHNOLOGY IN TRNAVA,
INSTITUTE OF APPLIED INFORMATICS, AUTOMATION AND MECHATRONICS,
ULICA JÁNA BOTTU 2781/25, 917 24 TRNAVA, SLOVAK REPUBLIC
e-mail: dusan.horvath@stuba.sk, peter.cuninka@stuba.sk

**Abstract**

*This article deals with the management of a Freescale small robotic car along the predefined guide line. Controlling of the direction of movement of the robot is performed by neural networks, and scales (memory) of neurons are calculated by Hebbian learning from the truth tables as learning with a teacher. Reflexive infrared sensors serves as inputs. The results are experiments, which are used to compare two methods of mobile robot control - tracking lines.*

**Key words**

*control subsystem of mobile robot, PID controller, neural networks, Hebbian learning*

## INTRODUCTION

To ensure reliable movement of the mobile robot on the guide line, fast and accurate data processing from input sensors with the smallest possible deviation of expected data from calculated data is essential. Therefore, it is necessary for learning of neurons to achieve the slightest learning error, so the robot can pursue the guide line as accurately as possible.

### Overview on the topic

At present, the robot control subsystem software implements the following methods:

- evaluating the data from the input sensor by different software and conditions (if, switch) by which the input data (input vector) from sensors on the robot are associated with desired outcomes (e. g. motor control);
- by PID controllers, where the input data (input vectors) are compared with the expected values and the difference of these values determines the regulation intervention (1-4);
- using the neural network, which comprises of weights of neurons (memory of neurons) depending on the input data from the sensors (input vector) and the expected outputs calculated during the learning process. During the operation of the robot, the control

subsystem calculates the robots response to the input vector based on the weights and the input vector.

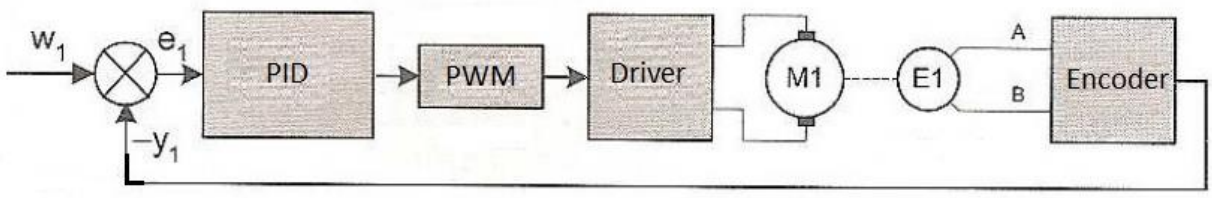



***Fig. 1*** *Example of electric motor control using PID controller (1)*

To follow the guide line a reflexive infrared or single-line camera are used. Quicker movement along the guide line is assured by a method of drawing up the map of the environment (5). To calculate the distance traveled, the motion sensor subsystem - encoders is used. Motion sensor subsystem is also used for robot navigation. The most common method of navigation is odometry (1, 6-7). Using odometry, the robot updates its location and records it in the map of the environment. Using environment maps a robotic car is capable of moving at a higher speed along the guide line.

The development of a control system for Freescale robotic cars is actively engaged at the Faculty of Electrical Engineering and Information Technology, Slovak University of Technology in Bratislava. In the European round of the prestigious technology competition, Freescale Cup, they have won gold medals repeatedly (8-10).

**Addressing the logical gate AND**

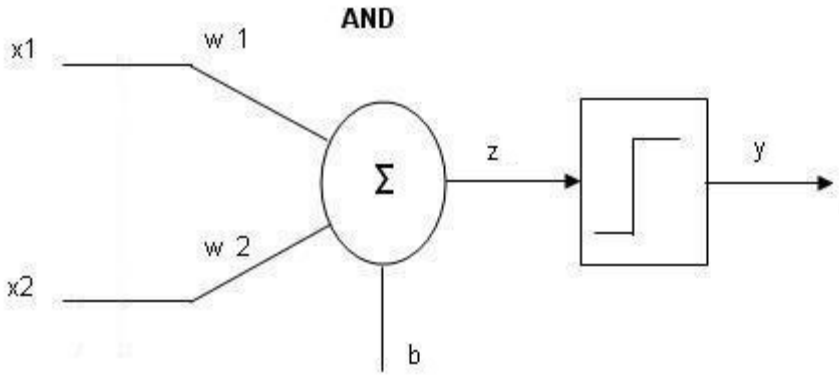



***Fig. 2*** *Example of dual-input neuron*

**w**=[0.2, 0.3], **b**=0,1 - random initialization of weights and bias.

**Table 1** Input vectors and expected outputs of **AND** function

| AND | | |
|---|---|---|
| **x2** | **x1** | **O** |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Self-calculation of the weights (adaptation) and the bias continues until the delta is equal to zero (or minimal) for all input vectors **X** (four in our example).

**Table 2** Weights calculation

| Logical function AND | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| line | x[0] | w[0]=b | x[1] | w[1] | x[2] | w[2] | output | z | y | delta Δ |
| 1 | 1 | 0.1 | 1 | 0.2 | 1 | 0.3 | 1 | 0.6 | 1 | 0 |
| 2 | 1 | 0.1 | 0 | 0.2 | 1 | 0.3 | 0 | 0.4 | 1 | -1 |
| 3 | 1 | -0.4 | 1 | 0.2 | 0 | -0.2 | 0 | -0.2 | 0 | 0 |
| 4 | 1 | -0.4 | 0 | 0.2 | 0 | -0.2 | 0 | -0.4 | 0 | 0 |
| 5 | 1 | -0.4 | 1 | 0.2 | 1 | -0.2 | 1 | -0.4 | 0 | 1 |
| 6 | 1 | 0.1 | 0 | 0.7 | 1 | 0.3 | 0 | 0.4 | 1 | -1 |
| 7 | 1 | -0.4 | 1 | 0.7 | 0 | -0.2 | 0 | 0.3 | 1 | -1 |
| 8 | 1 | -0.9 | 0 | 0.2 | 0 | -0.2 | 0 | -0.9 | 0 | 0 |
| 9 | 1 | -0.9 | 1 | 0.2 | 1 | -0.2 | 1 | -0.9 | 0 | 1 |
| 10 | 1 | -0.4 | 0 | 0.7 | 1 | 0.3 | 0 | -0.1 | 0 | 0 |
| 11 | 1 | -0.4 | 1 | 0.7 | 0 | 0.3 | 0 | 0.3 | 1 | -1 |
| 12 | 1 | -0.9 | 0 | 0.2 | 0 | 0.3 | 0 | -0.9 | 0 | 0 |
| 13 | 1 | -0.9 | 1 | 0.2 | 1 | 0.3 | 1 | -0.4 | 0 | 1 |
| 14 | 1 | -0.4 | 0 | 0.7 | 1 | 0.8 | 0 | 0.4 | 1 | -1 |
| 15 | 1 | -0.9 | 1 | 0.7 | 0 | 0.3 | 0 | -0.2 | 0 | 0 |
| 16 | 1 | -0.9 | 0 | 0.7 | 0 | 0.3 | 0 | -0.9 | 0 | 0 |
| 17 | 1 | -0.9 | 1 | 0.7 | 1 | 0.3 | 1 | 0.1 | 1 | 0 |
| 18 | 1 | -0.9 | 0 | 0.7 | 1 | 0.3 | 0 | -0.6 | 0 | 0 |
| 19 | 1 | -0.9 | 1 | 0.7 | 0 | 0.3 | 0 | -0.2 | 0 | 0 |
| 20 | 1 | -0.9 | 0 | 0.7 | 0 | 0.3 | 0 | -0.9 | 0 | 0 |

Calculated values of weights and bias: **w**=[0.7, 0.3], **b**= 0.9
x1=0,9/0,7=1,28
x2=0,9/0,3=3

The last four lines of Table 2 show, that the delta = 0 for all combinations of input vectors and the expected outputs. Learning of the network is completed. The result of the calculated values is one of the possible options, since the initiation of weights is random.

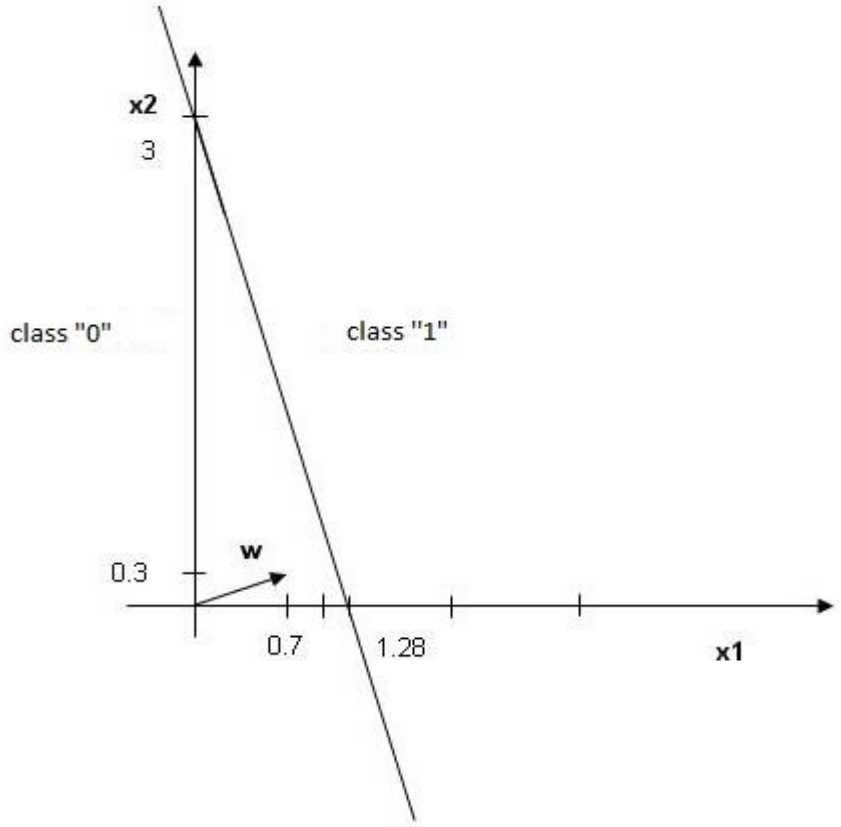



***Fig. 3*** *Distribution of the plane of the class of "0" and "1" according to Table 2*

An example of eight input neurons - dealing with the movement of the Freescale robotic car using linear activation functions.

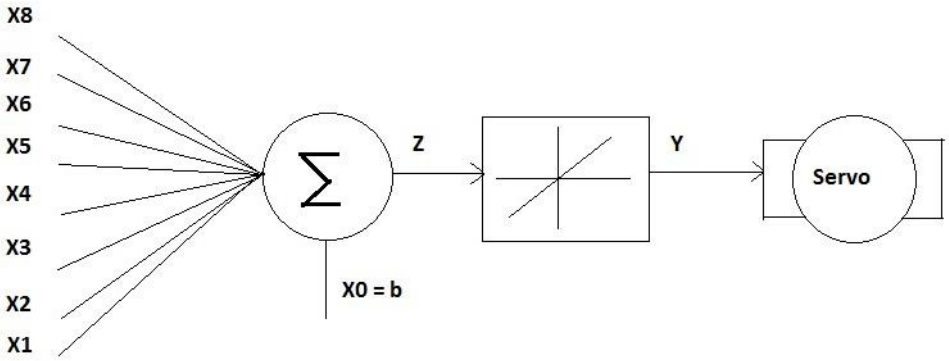
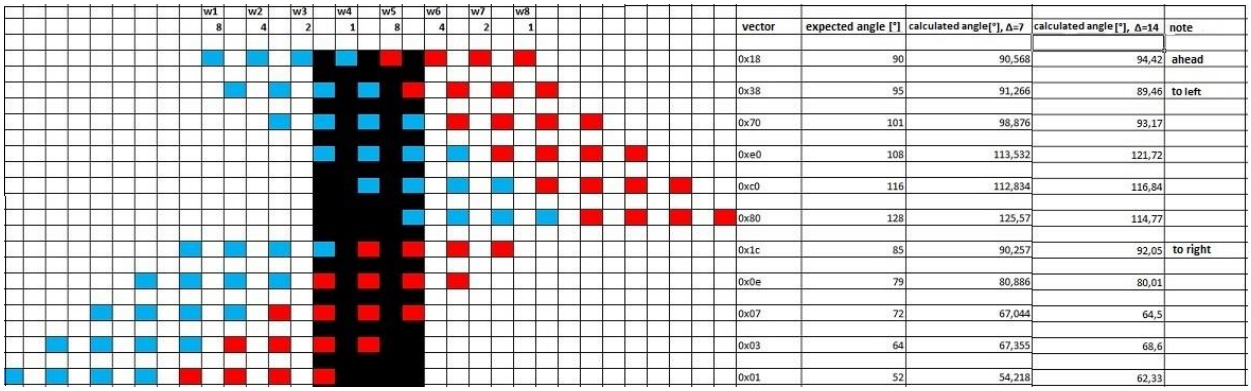



***Fig. 4*** *Diagram of the neural network with activated Signum function*

As seen in Fig. 4, to control the servo motor a single neuron with eight inputs is used. The table design of input vectors and the expected movement of the robot for a linear activation function is shown in Fig. 5. The calculated values of weights and bias are put into the control subsystem of a robot.




| vector | expected angle [°] | calculated angle[°], Δ=7 | calculated angle [°], Δ=14 | note |
|---|---|---|---|---|
| 0x18 | 90 | 90,568 | 94,42 | ahead |
| 0x38 | 95 | 91,266 | 89,46 | to left |
| 0x70 | 101 | 98,876 | 93,17 | |
| 0xe0 | 108 | 113,532 | 121,72 | |
| 0xc0 | 116 | 112,834 | 116,84 | |
| 0x80 | 128 | 125,57 | 114,77 | |
| 0x1c | 85 | 90,257 | 92,05 | to right |
| 0x0e | 79 | 80,886 | 80,01 | |
| 0x07 | 72 | 67,044 | 64,5 | |
| 0x03 | 64 | 67,355 | 68,6 | |
| 0x01 | 52 | 54,218 | 62,33 | |

***Fig. 5*** *Used input vectors for motion of robot cars along the track*

**Table 3** Calculated weights for learning error Δ = 14 and Δ = 7

| | w8 | w7 | w6 | w5 | w4 | w3 | w2 | w1 | w0 |
|---|---|---|---|---|---|---|---|---|---|
| Δ=14 | -15.413 | 4.887 | -1.098 | 0.991 | 15.255 | -0.612 | 1.818 | 39.211 | 75.560 |
| | | | | | | | | | |
| Δ=7 | -34.188 | 13.137 | -0.311 | -20.346 | 22.508 | 0.698 | -12.736 | 37.164 | 88.406 |

To verify a neuron with a linear transfer function a Freescale robotic car is used. For scanning the black guiding line, eight pieces of reflective infrared sensors will be used. To control the direction of movement of the robotic car, an Arduino microprocessor unit - Duemilanove eight-bit microprocessor, which controls the servo is used. The servo is mechanically connected to the front axle of the robot car and controls the direction of movement of the robotic car.

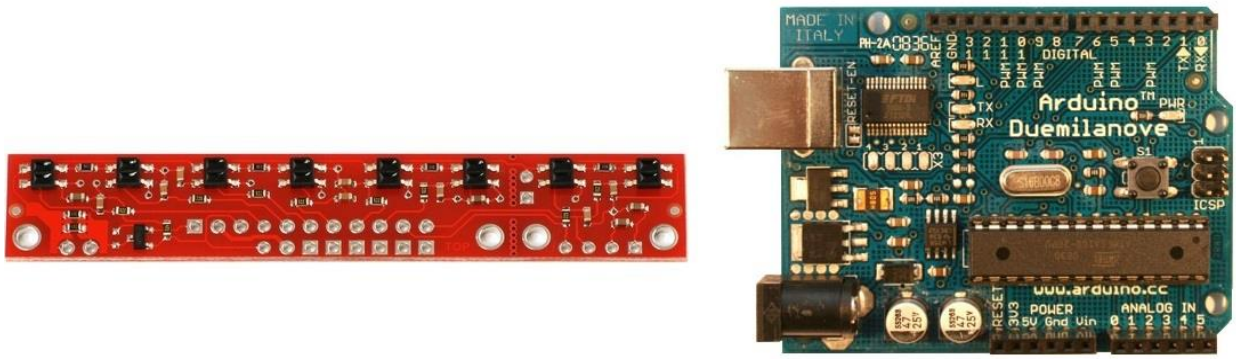



***Fig. 6*** *QTR-8RC sensor rail, microprocessor unit Arduino Duemilanove (11-12)*

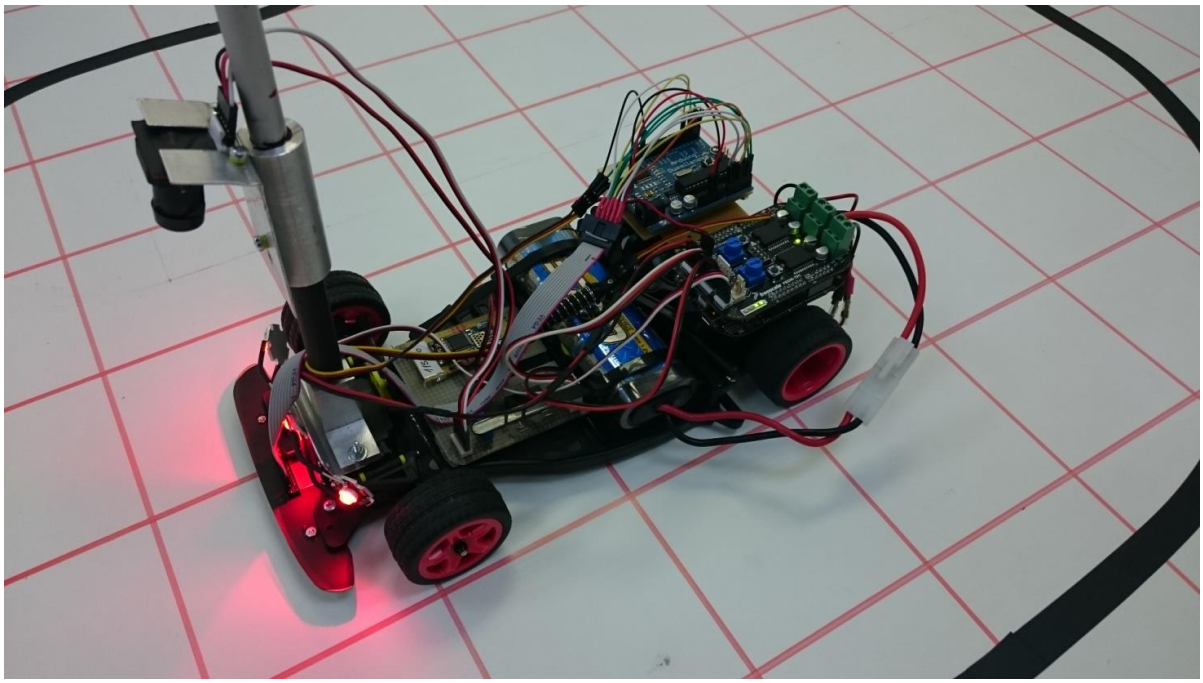


***Fig. 7*** *Freescale robotic car*

During its movement along the black line, the Robotic car captures the reflection from the surface underneath with their infrared sensors. As the reflection of infrared light from the black areas differs from the white surface, a different voltage is at the output of the infrared sensors. The voltage from the infrared sensors are digitized and represent the input vector to the table of the expected movement of the robot. The input vector and the weights are prepared from the equation $\mathbf{z(i)= \sum(x(i) * w(i)) + b}$. This results in values that go directly to the motion control subsystem of a robot car. The result of this action is to move the robot around a track.

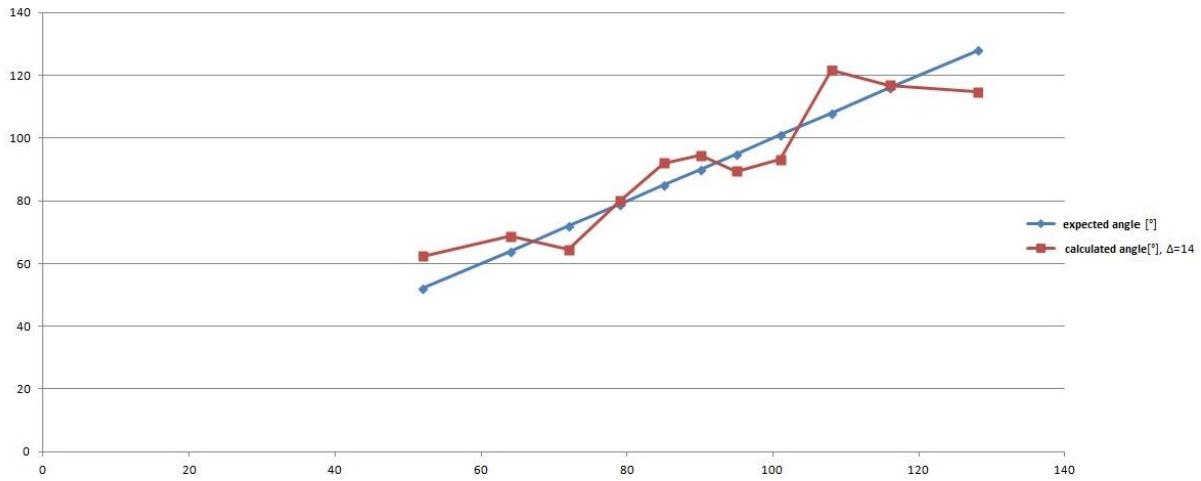



***Fig. 8*** *Graph of a moving Freescale robotic car with learning error Δ = 14*

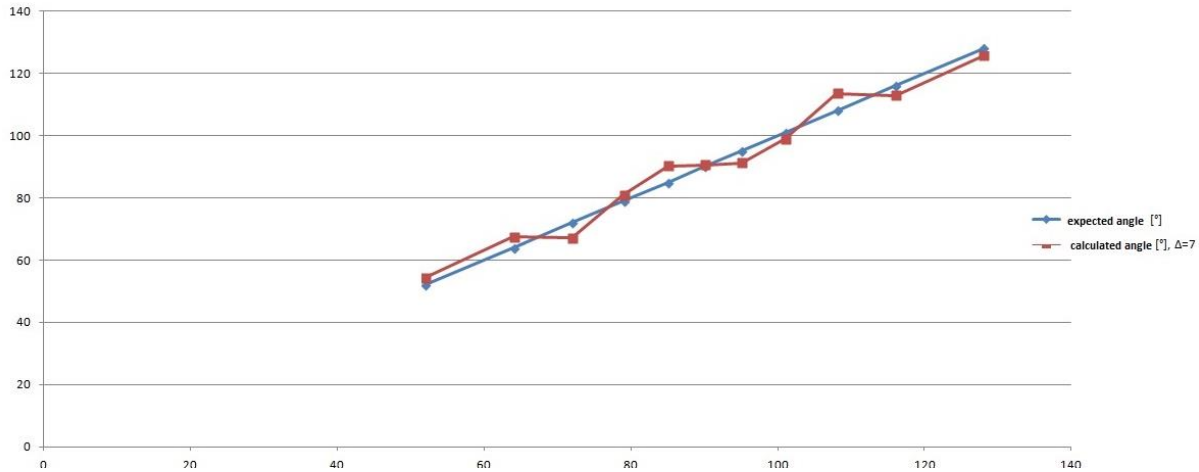



***Fig. 9*** *Graph of a moving Freescale robotic car with learning error Δ = 7*

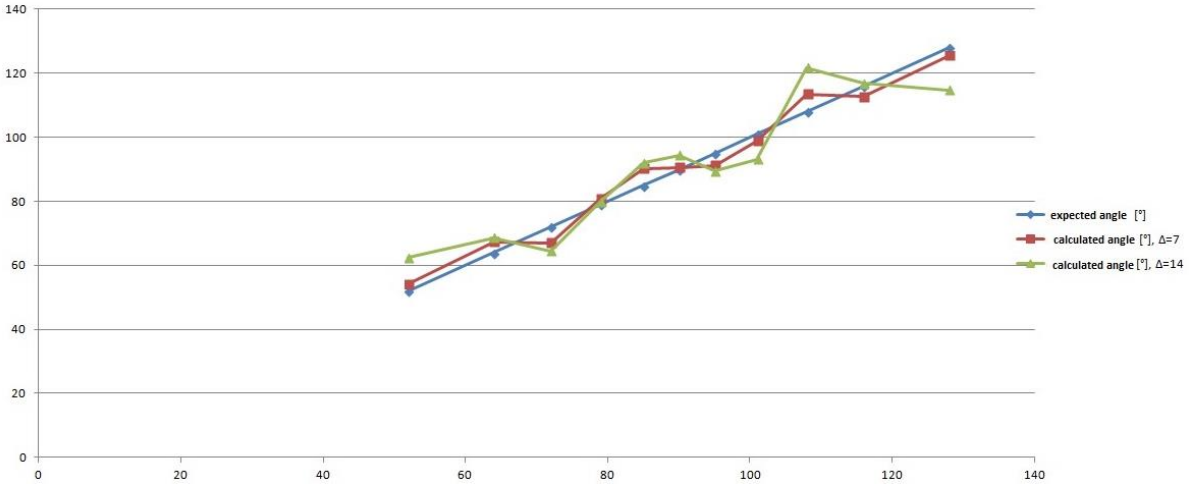



***Fig. 10*** *Graph of the movement of a Freescale robotic car - comparison of expected and calculated values of learning*

The blue line shows the expected angles of rotation of servos controlling the front axle of the robotic car in a traditional way of control (if, switch). The red line shows the servo steering angles calculated by Hebbian learning with learning error Δ = 7. The green line shows the angles of servo rotation with learning error Δ = 14. Figure 10 shows that the movement of the robot car along the guide line is not smooth, but the car is moving with certain variations. With learning error Δ = 14 variations are moving sideways so much that the robotic car runs out of track. The minimum deviation of the servo steering angle with the learning error Δ = 7 is 0.51 % (expected angle of 90°) and the maximum 5.97 % (expected angle of 108°). The minimum deviation of the servo steering angle with learning error Δ = 14 is 0.79 % (expected angle of 79°) and the maximum 16.93% (expected angle of 128°).

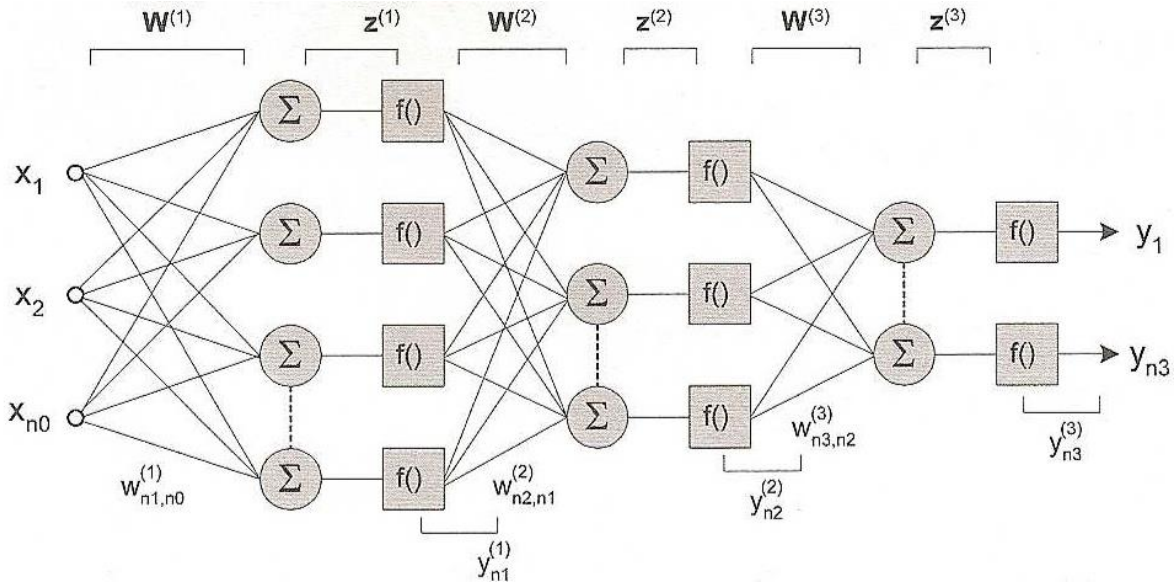



***Fig. 11*** *Multi-layer network (1)*

## CONCLUSION

The abovementioned neuron learning error $\Delta = 7$ works well for the robot car tracking lines on a simple track (round, oval). To move robot cars on complex pathways, where the robot car control subsystem must address not only turns, but also reducing cornering speeds so that the car stays on the track, speed bumps, intersections and hills, a use of a multi-layer network is essential (Figure 11). In the multi-layer networks with a large number of neurons in layers and greater number of inputs, it is not possible to manually define the parameters of such a network and therefore a use of computational algorithms is crucial. The most famous is the algorithm of "back - propagation". This is at least a double layer network consisting of neurons with differentiable activation function. As the activation function a sigmoid function of the form $y=1/(1+e^{-\lambda z})$ is most often used

Neural networks are used in various areas. From voice recognition to space robots control.


## Acknowledgement

This publication is the result of implementation of the project: "UNIVERSITY SCIENTIFIC PARK: CAMPUS MTF STU - CAMBO" (ITMS: 26220220179) supported by the Research & Development Operational Program funded by the EFRR.


**References:**


1. NOVÁK, P., 2005. *Mobilní roboty*. (*Mobile robots*). Prague: 247 p. ISBN 80-7300-141-1.
2. http://www.nxp.com/support/online-academy/the-nxp-cup-lecture-3-control-design:WBNR_FSLCUP_LECT3UMANAND?tab=otheng&type=traing&site_preference=normal&uc=true&lang_cd=en
3. https://www.pololu.com/docs/0J21/7.c
4. https://www.pololu.com/docs/0J26/all
5. http://www.atpjournal.sk/buxus/docs//casopisy/atp_plus/plus_2006_2/plus63_67.pdf
6. JURIŠICA, L., 1988. *Aplikovaná robotika* (*Applied robotics*). Bratislava: SVŠT, 904 p.

7. SIEGWART, R., NOURBAKHSH, I. R., 2004. *Introduction to Autonomous Mobile Robots, Massachusetts Institute of Technology*.
8. http://www.urpi.fei.stuba.sk/sk/content/freescale-cup-2014
9. http://www.stuba.sk/sk/diani-na-stu/prehlad-aktualit/studenti-stu-obhajili-zlato-na freescale-cup.html?page_id=8218
10. http://www.atpjournal.sk/vzdelavanie/skoly-a-institucie/vysoke-skoly/slovenska-technicka-univerzita-v-bratislave/fakulta-elektrotechniky-a-informatiky/ustav-riadenia-a-priemyselnej-informatiky/po-dramatickom-finale-striebro-pre-stu-nafreescale-cup.html?page_id=15057
11. https://www.pololu.com/product/961
12. https://www.arduino.cc/en/Main/ArduinoBoardDuemilanove