ISSN 1330-3651 (Print), ISSN 1848-6339 (Online) https://doi.org/10.17559/TV-20200727095821
Original scientific paper

Neural Network Driven Automated Guided Vehicle Platform Development for Industry 4.0 Environment

János SIMON, Monika TROJANOVÁ*, Alexander HOŠOVSKÝ, József SÁROSI

Abstract: This work presents the development of a new "Two Wheel" Automated Guided Vehicle (AGV) platform for education, competition, and research that is based on sensor fusion and neural networks (NN) with machine learning for Industry 4.0 applications. The method that is described in the paper deals with intelligent path planning and navigation of an AGV that should move safely in an unknown environment. The unknown environment may have obstacles of arbitrary shape and size that can move. Also is described the approach to solving the navigation problem in AGV navigation using a neural networks-based technique based on various types of input sensors. The neural network determines the safe direction for the next point section of the path in the environment while avoiding the nearby obstacles. Simulation examples of the generated path with proposed techniques will be presented.

Keywords: Automated Guided Vehicle (AGV); Industry 4.0; Machine learning; Neural networks (NN); Sensor fusion

1 INTRODUCTION

Industry 4.0 represents a period of digitization and transforms production and control systems in almost every industry. The Fourth Industrial Revolution represents a connection of people through mobile devices and other computing technology. The basic properties of Industry 4.0: unrestricted access to information and knowledge; and all will be supported by high data storage and computing power. Emerging technical discoveries will be reinforced by knowledge from the field of robotics, IoT, artificial intelligence, 3D printing technology, autonomous vehicles, and much more.

Mobile robotics is a relatively young and not yet explored field of mechatronics. Its roots are intertwined and include engineering and scientific disciplines such as mechanical engineering, electrical engineering, computer science, cognitive science, and the social sciences. Each discipline contributes to its share and directs researchers to design advanced prototypes. Robots include fundamental implemented components such as DC motors, stepper motors, sensors, and various others. However, in electronics and industry, robots require extreme precision, which is why microprocessors and other components are made that allow the existence of laptops and mobile phones. Therefore, we desire to improve the mobility of different types of robots.

Autonomous guided vehicles form a significant area of mobile robotics, which is of considerable interest. In rough terrain, where the wheels of devices cannot be moved, or the movement is limited, robots come to the fore. The person coordinates the goal position of the robot, but the robot controls the movement. For example, Pulstech's walking robot provides automatic coordination of the legs of the robot, which means that the person chooses only the location or direction of travel [1, 2]. HelpMate is a mobile robot used in hospitals for transportation tasks. It has a variety of sensors that allow it to move around corridors autonomously. We know several different patents and designs for autonomous robots. For example, HelpMate is oriented by light and following pre-determined wires that are laid on the ground and have sensors adapted for this purpose [3]. Various cleaning robots take advantage of this property to detect the geometry of the space and thus orient themselves in space. There are always questions and desires about how to locate and control robots. This currently represents one of the largest mobile robot markets in the world. Although mobile robots have a wide range of applications, we need to be aware of the facts that apply to almost every successful robot. The design should include a wealth of knowledge as it is an interdisciplinary science. To successfully solve the problem of robot motion, we need to know the mechanics, kinematics, dynamic properties of the robot, and control theory [4]. For the system to function to our liking, it must correctly use signal analysis. Localization and navigation require knowledge of computer algorithms, computer science, art intelligence, and probability theory [5].

AGV represents the way how to improve mobility, or how to streamline the material/goods transfer process (for example, implementing an AGV system is described in [6]). Manipulation is one of the fundamental factors that affect the total costs of the company, and thus the resulting profit. Therefore, it is necessary to think about safety, efficiency, accuracy, cost, time, and others when handling. AGV research began in the middle of the 20th century (Barnet), and since then, it has found application mainly in areas of industry where the transport of material on the repetitive trajectory is expected [7]. That is why, if the fourth industrial revolution is mentioned, AGVs are certainly among the elements that deserve attention in the field of further research and development (increasing demands on their size, energy savings, materials used, etc.).

One of the features of modern companies, and especially modern factories, is the integration of elements that increase automation and production efficiency. Such an element is the use of AGVs. Their task is mainly to locate objects and transport objects. In their development, design, and construction, it is necessary to keep in mind that the operating conditions are different lighting, ambient temperature, dust, size of the workspace, and others [8]. AGVs need to be adapted to this. In this article, the authors focused on creating an AGV vehicle where an artificial neural network was chosen to determine the rotation angles of robot joints. The ability of the AGV to traverse the final

(desired) pathway depends on several factors - e. g. from the degree of learning ability of the neural network, input data, training phase, using sensors. All this can affect the resulting required vehicle path (Fig. 1 - red curve). In the absence of a certain factor mentioned above (or their combination), this would mean a deviation from the desired path (Fig. 1 - blue marked space). Such a deviation could cause the AGV to deviate from its safe zone, and at the same time disrupts the safe zone of personnel or equipment and disrupts the safe operation of the company.

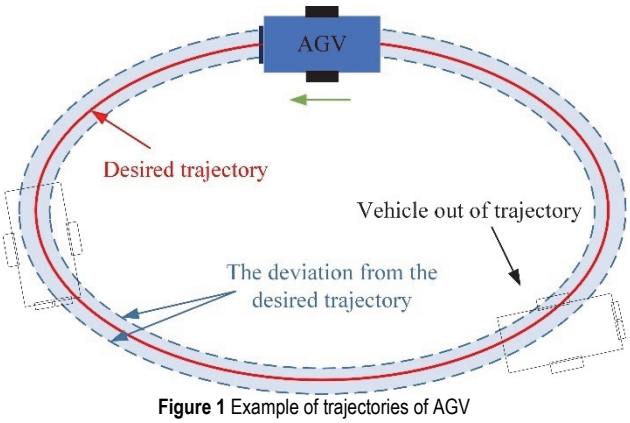

Figure 1 Example of trajectories of AGV

The authors of the article tried to implement IoT elements in the design of the AGV. The ESP32 is a 2.4 GHz Wi-Fi chip and Bluetooth designed with ultra-low power TSMC 40 nm technology. This chip is designed to deliver the best Radio Frequency (RF) performance and performance, demonstrating flexibility, versatility, and reliability in a variety of applications and profiles. ESP32 is designed for mobile, portable, and Internet-of-Things (IoT) applications. The important features are low-power surfaces including more power and dynamic performance reduction. For example, in the low-power IoT sensor application scenario, ESP32 will wake up periodically and only when a specific condition is found. The low chip duty cycle minimizes power consumption. The output of the power amplifier is also adjustable, contributing to the optimum level of communication range, data rate, and power consumption.

2 TECHNICAL BACKGROUND OF DEVELOPED AGV PLATFORM

The general structure of mobile robots consists of the following parts. The robot senses changes in the environment and its events with its sensors. The on-board computer unit, the "brain" of the robot processes the acquired information and controls the propulsion system and, through it, the actual actuators. There are two main approaches to designing mobile robots [9]. On the one hand, we can create a wheeled mobile robot, which can be used efficiently on pre-built roads, and on the other hand a two-legged or multi-legged mobile robot standing on and moving on some legs, which is more suitable for the terrain.

We can use different types of sensors on a robot. We can measure the distance with an ultrasound sensor, with a laser rangefinder (1D, 2D, 3D), or with an infrared sensor [2, 10]. The visual input of the environment can be obtained with cameras [11]. In direct physical interaction, it senses the robot's environment with collision, pressure, or even tactile sensors [12].

With a thermometer, we can detect not only the condition of the outside environment but also the state of some of its internal parts. Additional internal sensors can measure voltage and power, and transducers measure the rotation of the wheels before and after transmission on the axles [13]. For measuring the position and movement of the robot, an accelerometer, a gyroscope, or a magnetometer, can be used, called an inertial measurement unit (IMU). GPS and compass are also widely used to help with positioning. These tools allow us to create a 3D view and model of the world, while the robot is measuring its position in the 3D world with sufficient accuracy [14, 15].

Based on the developed concept of hierarchical intelligent control for AGV, an integrated hardware-software experimental system was implemented. An artificial NN with backward propagation of the error (BP NN) was chosen to determine rotation angles in robot wheels, as it can map the input layer (object position and orientation) to the output one (robot wheel rotation angles). A learning algorithm based on a generalized delta rule was used to teach BP artificial NNs [16].

In Fig. 2, you can see the developed architecture of the AGV platform, where the basic part is the yellow highlighted ESP32 IoT platform with the specified specifics. Other parts are connected to it: the orange part is the power supply (energy source), the blue part is the drives, and the green part consists of sensors and a camera.

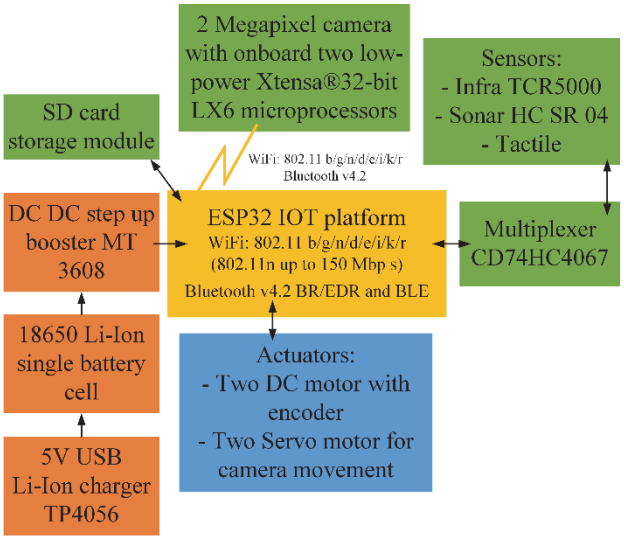

Figure 2 Developed AGV platform architecture

Tab. 1 contains the basic specifications of the individual components used in the developed AGV by authors. The AGV`s Samsung 18650 3.8 V 3000 mAh Li-ion has an autonomy of ca. 180 minutes of operation. Full Charge in 90 minutes with TP4056 charger. The robot has a payload capacity of ca. 800 g weight, which is suitable for carrying loads like a small robotic arm, the camera with a pan-tilt mechanism, and other small equipment and electronics.

Table 1 AGV specifications

Component / parameter name	Characteristic
Power supply	Li-ion 3000 mAh 3.8V 18650 cell
Power management	DC-DC SX1308 Step-Up 5V and 6V
USB charger	Micro USB 5 V 1 A 18650 TP4056 Lithium Battery Charger
Controller board	ESP32S
Analog / digital I/O	CD74HC4067 16-Channel Multiplexer
Sensors	IR Infrared Sensor Module; Sonar Module; Tactile sensors
Motor driver	L9110S
Motors	DC Gear Motor with encoders
Data storage	SD card module
Sound	Active buzzer
Camera	ESP32 Cam

Fig. 3 shows a photograph of a realistical AGV constructed by the authors using the components listed in Tab. 1, fasteners, and construction. Most parts of the construction were made using 3D printing technology (the approximate cost of this AGV's construction is ca. 100EUR). The use of this technology makes it possible to design elements of the desired shape, low weight, and reduced costs, which is certainly suitable as an alternative in the construction of prototypes or atypical shapes. The figure shows the location of the most important parts, such as sensors, ESP 32 drive. Dimensions of the AGV (in m) are 0.240 × 0.150 × 0.155 ($L \times W \times H$).

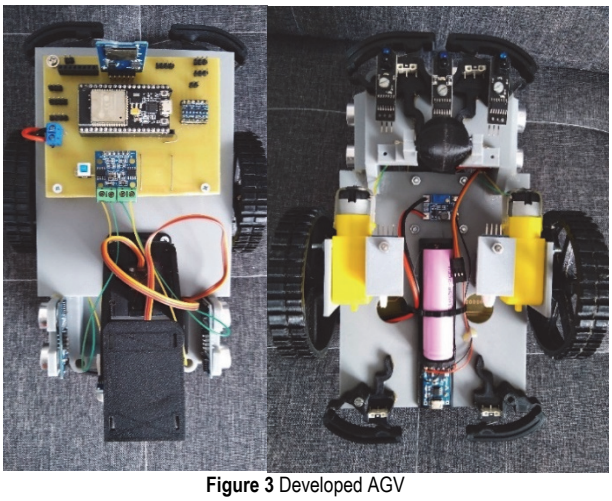

Figure 3 Developed AGV

3 SENSOR FUSIONS

Sensor fusion is the process of merging data from multiple sensors such as to reduce the amount of uncertainty that may be involved in a robot navigation motion or task performing [17]. Sensor fusion technologies allow us to improve the experience, utilization, and combination of sonar sensors, infra, and tactile sensors. Each of these types of sensors provides unique functionality [18].

Sensor fusion is a combination of data, which is especially important when combining data of different types (as in the case of autonomous guides). The AGV is a combination of information from a camera whose task is human vision and sensors that carry information about the distance of the AGV from objects (obstacles) [19].

When all these technologies are combined, sensor fusion takes the input from multiple sensors simultaneously, processes the input, and produces an output. Sensor fusion using algorithms and filtering techniques eliminates the shortcomings of each sensor [20]. The most important aspect of sensor fusion is the ability to analyse the context of the situation [21]. If the AGV does not know the context of the situation, it cannot provide relevant information for real-time spatial orientation. The ideal case is when the context is reached in real-time without the need for human action. Sensor fusion includes methods and algorithms, including the Central limit theorem, Kalman filter, Bayesian networks, and others. Detailed description of multisensor fusion and consensus filtering is in [22].

Fig. 4 shows how the sensor fusion method works - data from several sensors are clustered and processed by sensor fusion processes, then they are cumulated into one process, the data processing results in an output from the system (some action, e. g. movement of AGV). Fig. 5 shows what Sensor fusion looks like from the camera image and sonar.

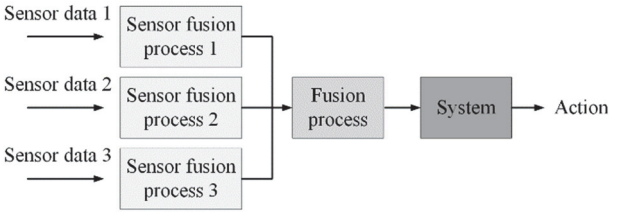

Figure 4 Sensor fusion method

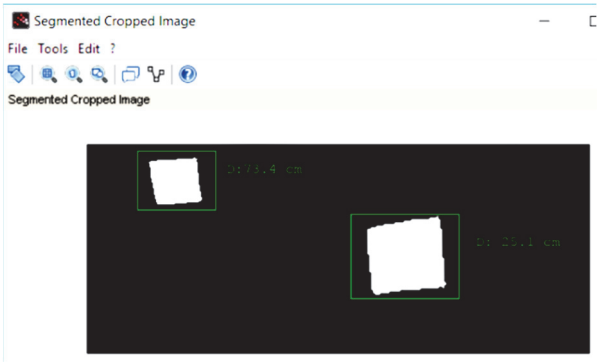

Figure 5 Sensor fusion from camera image and sonar

4 NEURAL NETWORK AS TOOL FOR TRAJECTORY TRACKING OF AGV´S

The multilayer perceptron neural network (MLP NN) represents one of the best known and one of the most used types of architecture of artificial neural networks. The basic units of a network-neurons (also called perceptrons) are located on one or more hidden layers. The role of NN is the transformation of input to output, while the key role is played by the activation function (the most typical activation functions of MLP NN are given in Tab. 2). The main task of the neural network is the approximation of a nonlinear function [23, 24].

For a NN it is necessary to work out the right weights and edge, limits that make certain operations as desired. For years this was a hard question especially for a many-level NN, where it was not known the relation between the error to the weights of the different levels. Today exist very good algorithms for NN weight selection and tuning. For example, the backpropagation training algorithm is one of

the greatly respected algorithms because of its simplicity and power [25, 26].

Table 2 Activation functions

Name of the activation functions	
Linear threshold	$y = x$
Sigmoid (logistic curve)	$y = \frac{1}{1+e^{-x}}$
Symmetric sigmoid	$y = \frac{1-e^{-x}}{1+e^{-x}}$
Hyperbolic tangent	$y = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$

Fig. 6 shows the general scheme of a multilayer perceptron neural network, where on the input layer are inputs signal $u_1 - u_p$ (stimulus). The network in the figure has one hidden layer with M neurons (perceptrons). Each perceptron has a threshold (w_{M0}). On the output layer is one neuron with output signal y (response). Its bias is marked as w_0.

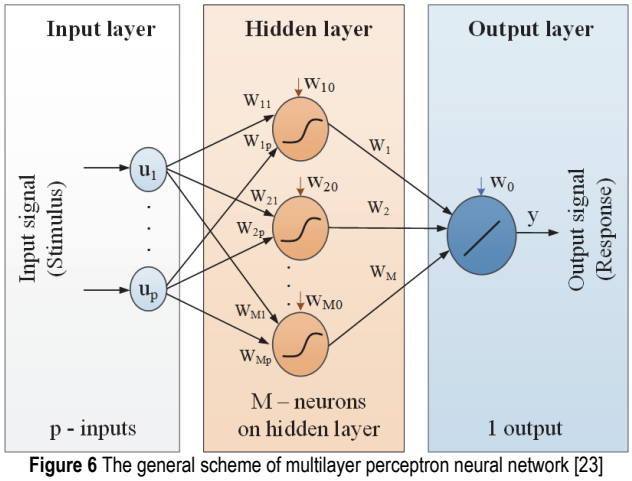

Figure 6 The general scheme of multilayer perceptron neural network [23]

The output of the output layer of the multilayer perceptron NN can be defined as follows [23]:

$$y = \sum_{i=0}^{M} w_i \phi_i \left(\sum_{j=0}^{p} w_{ij} u_j \right) \phi_o(\cdot) = 1;\ u_0 = 1 \tag{1}$$

where:

- u_j: inputs of the neural network,
- y: output of the neural network,
- ϕ_i: activation function,
- w_i: synaptic weight of the output layer,
- w_{ij}: synaptic weight of the hidden layer,
- M: the number of perceptrons of hidden layer,
- p: the number of inputs.

The Backpropagation algorithm is a method whose task is to calculate the gradients of the MLP network according to weights. It is a method of calculating derivatives according to the chain rule with application to neural networks. The output derivatives of the MLP network of the i-th weight of the output layer can be determined based on Eq. (2), where the meaning of the individual symbols in the following equations is the same as in Eq. (1). This derivative of the output by weight is equal to the basic function ϕ_i.

$$\frac{\partial y}{\partial w_i} = \phi_i;\ \phi_0 = 1;\ i = 0, ..., M \tag{2}$$

The derivatives of the output of the MLP network according to the weights of the hidden layer can be determined based on Eq. (3), which applies to the weight of the connection between the j-th input and the i-th neuron in the hidden layer. The function $g(x)$ represents an activation function, e.g. tan$h(x)$, where after substituting this form of the function into Eq. (3), the relationship takes the form of Eq. (4). It follows that the activation function ϕ_i depends on two parameters, namely the inputs and weights of neurons on the hidden layer [23].

$$\frac{\partial y}{\partial w_{ij}} = w_i \frac{\mathrm{d}g(x)}{\mathrm{d}x} u_j;\ u_0 = 1;\ i = 1, ..., M, j = 0, ..., p \tag{3}$$

$$\frac{\partial y}{\partial w_{ij}} = w_i \left(1 - \phi_i^2\right) u_j;\ u_0 = 1 \tag{4}$$

The error signal $e_i(k)$ of the neuron i, that is an output node, is defined by Eq. (5) [27]:

$$e_i(k) = d_i(k) - y_i(k) \tag{5}$$

where $d_i(k)$ is the response desired for neuron i at iteration k; $y_i(k)$ is the output of the neuron i at iteration k.

The total output error can be calculated [27]:

$$E(k) = \frac{1}{2} \sum_{j \in C} e_i^2(k) \tag{6}$$

where C includes neurons of the output layer of the NN.

Therefore, if we express the calculation of the weight of neurons on the output layer w_M in the iteration $k + 1$, then we obtain the relation:

$$w_M(k+1) = w_M(k) - \eta \frac{\partial E(k)}{\partial w_M} \tag{7}$$

where $w_M(k)$ is the weight of neurons on the output layer at iteration k and $E(k)$ is output error. Parameter η is the learning rate. Similarly, we express the weight of hidden layer neurons on the iteration $k + 1$ using the weights of neurons from the hidden layer in the iteration k and the error $E(k)$.

$$w_{Mp}(k+1) = w_{Mp}(k) - \eta \frac{\partial E(k)}{\partial w_{Mp}} \tag{8}$$

Thus, in general, the delta rule is used to determine the correction $\Delta w_{ij}(k)$ in Eq. (9), whereas the designation of the individual parameters is identical to the previous equations [27].

$$\Delta w_{ij}(k) = -\eta \frac{\partial E(k)}{\partial w_{ij}(k)} \tag{9}$$

The frequently used BP algorithm can be modified by adding momentum, resulting in an algorithm called BP with momentum (BPM) and it can be defined by Eq. (10) [28].

$$\Delta w_{ij}(k) = -\eta \frac{\partial E(k)}{\partial w_{ij}(k)} + \alpha \cdot \Delta w_{ij}(k-1) \tag{10}$$

where α is momentum factor ($0 < \alpha \leq 1$). Usually is used value of the factor $\alpha = 0.9$ (or 0.95). Momentum introduces a memory effect so that the NN does not only react to the local gradient, but also the recent trends in the surface of the error. The network can get stuck in the local minima without momentum but adding momentum can help the network ride through the local minima. The above-said algorithm was used in implementing the mobile robot trajectory tracking controller to train the appropriate NN [28].

5 RESULTS OF DEVELOPMENT AND IMPLEMENTATION OF NEURAL NETWORK FOR AGV´S TRAJECTORY TRACKING

To create a NN to approximate a general multi-input multi-output function, an algorithm should be used to initialize the NN, use the backpropagation algorithm to train the network, and use the NN's approximation error to view its results.

In this work is used the following terminology to describe various parameters of the NNs implemented:

- N_p: number of input data patterns,
- N_c: number of cycles to train the NN,
- W_{ij}: weights - hidden layer to input layer,
- W_i: weights - output layer to hidden layer,
- N_i: number of inputs to the NN,
- N_o: number of outputs of the NN,
- N_h: number of neurons in the hidden layer,
- η: learning rate,
- α: momentum rate.

The general algorithm used to construct the NN using backpropagation is depicted in Fig. 7.

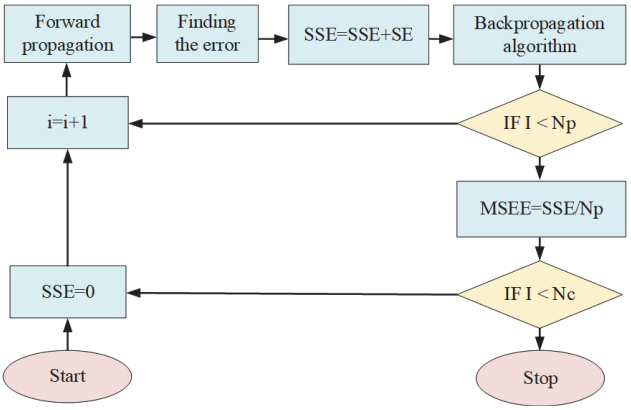

Figure 7 The general algorithm for implementation of NN training using backpropagation

The first step of the algorithm is to configure the neural network, which is to specify the number of inputs, the number of outputs, and the number of neurons in the hidden layer, and the level of learning rate and momentum rate.

The next step of the configuration of the network is to identify the activation function to use for the functions of the hidden layer and the output layer. The next step is to build the portion of learning and training composed of the following parts:

- An iteration for the number of cycles we want to repeat the same set of input data to train the network and reach the desirable error rate. The number of iterations of this loop is N_c and the loop index is j.
- An iteration for the number of patterns of data we have in each cycle. This loop will read all of the data in one cycle. The number of repetitions of this loop is N_p and the loop index is i.

As can be seen from Fig. 7, the loop reading the information patterns in each cycle is inside another loop repeating the entire data for some cycles. Until doing the backpropagation for each data set, the first thing we need to do is do the forward propagation and find the error between the network output and the desired value.

We can calculate the error for each input data pattern after doing the forward propagation and determine the sum of the error squared by using the following equation:

$$SSE = \sum_{i=1}^{Np} E_p^2 \tag{11}$$

In this equation are used the following parameters: Np is the number of patterns of input data in one cycle and E_p is the pattern approximation error. We have one SSE with one output data cycle by the start of the data pattern chain. The next step is to find the mean square sum error ($MSSE$) based on the next equation:

$$MSSE = \frac{\sum_{i=1}^{Np} E_p^2}{Np} \tag{12}$$

Calculations for the entire loops and patterns are achieved by finishing the data cycle process and we can test the network performance by looking at the $MSSE$ curve and comparing the NN inputs with the real function.

As an early test step, the Scilab code built to impel the above algorithm was used. An example of how to approximate a two-input one output function using this algorithm is defined as follows:

- Number of hidden layers: $N_h = 10$
- Learning rate: $\eta = 0.11$
- Momentum rate: $\alpha = 0.96$
- Number of cycles: $N_c = 100$
- Approximated function: $f(x_1, x_2) = \sin(\pi \cdot x_1)\sin(\pi \cdot x_2)$
- Activation function: $f(x) = \frac{1-e^{-x}}{1+e^{-x}} \log - sigmoid$

The inputs of the network are the information from four tactile sensors, three infra sensors, sonar sensors (placed on the left, the front, and the right side of the AGV),

and the camera image. Sensors are activated according to the proximity of an obstacle in that guidance. The NN will output commands like left, right, forward, backward, which will allow correcting the position and thus, avoiding the obstacle. NNs are excellent tools applicable in AGV systems for evading obstacles, which can work with imprecise information. The approximated output of the NN and the real output is depicted in Fig. 8 and represents the torque to the left wheel versus NN approximated model output. From the figure can be seen very well the performance of the approximation algorithm. The implementation of the algorithm will be used in designing the controllers which need the approximation property of a NN.

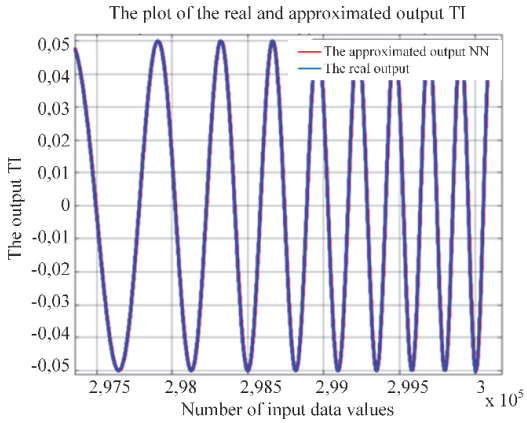

Figure 8 The neural network output

The result of the simulation presenting that it is easy for a mobile robot to perform the required motion from the starting position to the target is depicted in Fig. 9. Fig. 9 also shows the movement of the mobile robot in the workspace in a manner that follows the reference trajectory. The accuracy during trajectory navigation was 0.05 m at a speed of 0.2 m/s. Ideal monitoring of the reference trajectory is visible, which means that there will be no large deviations of the current position of the mobile robot from the reference position.

From Fig. 9 is seen the uniform motion of the mobile robot towards the target, as confirmed next in Fig. 10 where there are no sudden jumps of control magnitudes. One of the most important questions in mobile robotics is whether it is possible to implement simultaneous localization and mapping (SLAM) simultaneously if the location of the robot is unknown and its surroundings are unknown. Then the robot step-by-step enlarges and builds a coherent map of its surroundings while determining its own.

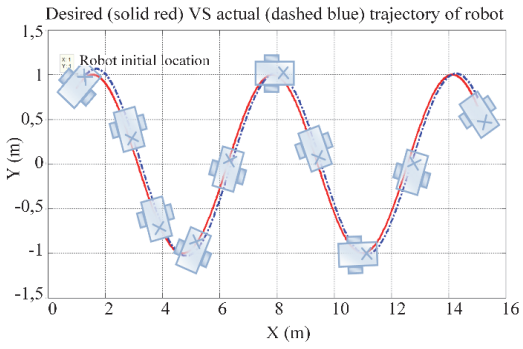

Figure 9 Simulation setup

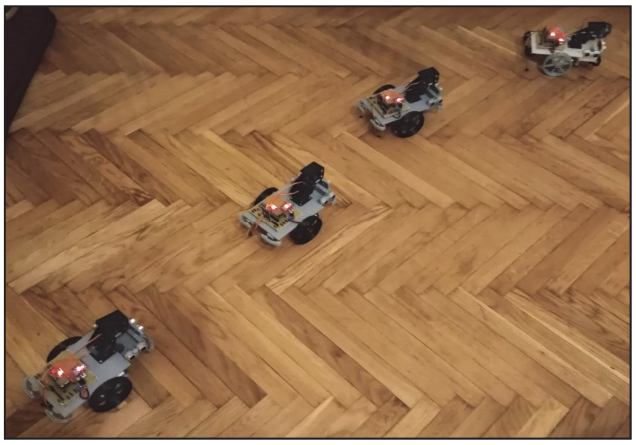

Figure 10 Real system test

6 CONCLUSION

The article described the design and implementation of a new trajectory tracking controller for the differential drive of AVG on a developed robot platform. The physical AGV platform can be improved by the addition of one heading sensor for precise measurement of the heading angle of the robot. This feature will eliminate the drifting error of the robot due to encoder readings and highly improves the tracking performance. Thus, the experimental results obtained confirm the justification hypotheses about the robot's mechatronic system ability to learn, using multi-sensory information obtained from an external sensor-camera and artificial NN systems.

In the paper was given the neural network's architecture and implementation for the approximation of a general multi-input multi-output function. The authors discussed and analysed the choice of training algorithm and neural network parameters. This control algorithm can handle disturbances and unstructured unmodeled dynamics that are part of any environment that AGV wants to navigate. To compensate for the nonlinear terms in the system is needed no knowledge of the system dynamics especially when a newly developed AGV platform with unknown parameters and inner layers is used for the tracking performance.

Acknowledgement

The project was supported by the Autonomous Province of Vojvodina, Provincial Secretariat for higher education and scientific research Project number: 142-451-2446/2019-02/2 and by the Project VEGA no. 1/0393/18 - Research of Methods for Modeling and Compensation of Hysteresis in Pneumatic Artificial Muscles and PAM-actuated Mechanisms to Improve the Control Performance Using Computational Intelligence.

7 REFERENCES

[1] Theunissen, J., Xu, H., Zhong, R. Y., & Xu, X. (2018). Smart AGV System for Manufacturing Shopfloor in the Context of Industry 4.0. *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*, 1-6. https://doi.org/10.1109/M2VIP.2018.8600887

[2] Kim, J. Y., Kashino, Z., Colaco, T., Nejat, G., & Benhabib, B. (2018). Design and implementation of a millirobot for

swarm studies mROBerTO. *Robotica*, *36*(11), 1591-1612. https://doi.org/10.1017/S0263574718000589
[3] Carvalho, F., Endler, M., & Silva, F. S. (2019). Leveraging Application Development for the Internet of Mobile Things. *Open Journal of Internet of Things (OJIOT)*, *5*(1), 12.
[4] Mester, G. (2010). Intelligent Mobile Robot Motion Control in Unstructured Environments. *Acta Polytechnica Hungarica*, *7*(4), 13.
[5] Agmell, S. & Dekker, M. (2019). *IR-Based Indoor Localisation and Positioning System* (1st ed.).
[6] Correia, N., Teixeira, L., & Ramos, A. L. (2020). Implementing an AGV System to Transport Finished Goods to the Warehouse. *Technology and Engineering Systems Journal*. https://doi.org/10.25046/aj050231
[7] Kumar Das, S. (2016). Design and Methodology of Automated Guided Vehicle-A Review. *IOSR Journal of Mechanical and Civil Engineering*, *03*(03), 29-35. https://doi.org/10.9790/1684-15010030329-35
[8] Ruiz-Carrizal, J., Orta, P., Vargas-Martínez, A., Urbina, P., & Ramirez-Mendoza, R. A. (2020, February 14). *Smart AGV development and testing*. Virtual Concept Workshop 2020 - SMART 4.0: Technology and Innovation Disruption, Monterey, Mexico.
[9] Sarieva, M., Yao, L., Sugawara, K., & Egami, T. (2019). Synchronous Position Control of Robotics System for Infrastructure Inspection Moving on Rope Tether. *Journal of Robotics and Mechatronics*, *31*, 317-328. https://doi.org/10.20965/jrm.2019.p0317
[10] Vis, I. F. A. (2006). Survey of research in the design and control of automated guided vehicle systems. *European Journal of Operational Research*, *170*(3), 677-709. https://doi.org/10.1016/j.ejor.2004.09.020
[11] Oleksiuk, K. (2019). *The use of Bluetooth BLE technology for the localization of an autonomous vehicle* [Doctoral disseration]. Instytut Pojazdów.
[12] Jung, J., & Jung, J. (2018). *Various Swarm Behavior Inspired by Nature Using Vibration Locomotion Robots* [Undergraduate Thesis]. Yonsei University.
[13] Denisov, A., Iakovlev, R., & Lebedev, I. (2019). Mathematical and Algorithmic Model for Local Navigation of Mobile Platform and UAV Using Radio Beacons. In A. Ronzhin, G. Rigoll, & R. Meshcheryakov (Eds.), *Interactive Collaborative Robotics*, 53-62. Springer International Publishing. https://doi.org/10.1007/978-3-030-26118-4_6
[14] Mekki, K., Bajic, E., & Meyer, F. (2019). Indoor Positioning System for IoT Device based on BLE Technology and MQTT Protocol. *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 787-792. https://doi.org/10.1109/WF-IoT.2019.8767287
[15] Xiao, B.-X., Qi, D.-L., Liu, H.-X., & Li, S.-S. (2006). *AGV path planning in the dynamic environment based-on fuzzy neural network*. *18*, 2401-2404.
[16] Vanneschi, L. & Castelli, M. (2018). Delta Rule and Backpropagation. In *Reference module in Life Sciences*. https://doi.org/10.1016/B978-0-12-809633-8.20340-3
[17] Arun, M., Jenson, D., Shree, V. J., Nandhini, V. M., & Abinaya, B. (2019). Smart Grid Robot Exclusively Designed for High Power Transmission Lines. *2019 5th International Conference on Advanced Computing Communication Systems (ICACCS)*, 652-654. https://doi.org/10.1109/ICACCS.2019.8728307
[18] Muchtar, F., Singh, P. K., Kumar, Y., Ariffin, A. H., Fadilah, S. I., & Yusoff, Mohd. N. (2018). ToMRobot: A Low-Cost Robot for MANET Testbed. *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, 54-59. https://doi.org/10.1109/PDGC.2018.8745838
[19] Kocic, J., Jovičić, N., & Drndarevic, V. (2018). *Sensors and Sensor Fusion in Autonomous Vehicles*. 420-425. https://doi.org/10.1109/TELFOR.2018.8612054
[20] Su, H. R. & Chen, K. Y. (2019). Design and Implementation of a Mobile Robot with Autonomous Door Opening Ability. *International Journal of Fuzzy Systems*, *21*(1), 333-342. https://doi.org/10.1007/s40815-018-0557-5
[21] Foltýnek, P., Babiuch, M., & Šuránek, P. (2019). Measurement and data processing from Internet of Things modules by dual-core application using ESP32 board. *Measurement and Control*, 002029401985774. https://doi.org/10.1177/0020294019857748
[22] Li, W., Wang, Z., Wei, G., Ma, L., Hu, J., & Ding, D. (2015). A Survey on Multisensor Fusion and Consensus Filtering for Sensor Networks. *Discrete Dynamics in Nature and Society*; Hindawi. https://doi.org/10.1155/2015/683701
[23] Nelles, O. (2001). *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*. Springer-Verlag. https://doi.org/10.1007/978-3-662-04323-3
[24] Gholami, M., Cai, N., & Brennan, R. W. (2013). An artificial neural network approach to the problem of wireless sensors network localization. *Robotics and Computer-Integrated Manufacturing*, *29*(1), 96-109. https://doi.org/10.1016/j.rcim.2012.07.006
[25] Azenha, A., & Carvalho, A. (2009). A neural network approach for AGV localization using trilateration. *2009 35th Annual Conference of IEEE Industrial Electronics*, 2699-2702. https://doi.org/10.1109/IECON.2009.5415429
[26] Židek, K., Lazorík, P., Piteľ, J., & Hošovský, A. (2019). An Automated Training of Deep Learning Networks by 3D Virtual Models for Object Recognition. *Symmetry*, *11*(4), 496. https://doi.org/10.3390/sym11040496
[27] Haykin, S. S. (1999). *Neural Networks: A Comprehensive Foundation*. Prentice Hall.
[28] Du, K. L. & Swamy, M. N. S. (2006). *Neural Networks in a Softcomputing Framework*. Springer-Verlag. https://doi.org/10.1007/1-84628-303-5

Contact information:

János SIMON, PhD. Dr. Eng.
Department of Technology,
Faculty of Engineering, University of Szeged,
6724 Szeged, Hungary
E-mail: simon@mk.u-szeged.hu

Monika TROJANOVÁ, PhD. Ing.
(Corresponding author)
Department of Industrial Engineering and Informatics,
Faculty of Manufacturing Technologies with a Seat in Prešov,
Technical University of Košice,
Bayerova 1, 080 01 Prešov, Slovakia
E-mail: monika.trojanova@tuke.sk

Alexander HOŠOVSKÝ, PhD. doc. Ing.
Department of Industrial Engineering and Informatics,
Faculty of Manufacturing Technologies with a Seat in Prešov,
Technical University of Košice,
Bayerova 1, 080 01 Prešov, Slovakia
E-mail: alexander.hosovsky@tuke.sk

József SÁROSI, PhD. Dr. Eng.
Department of Technology,
Faculty of Engineering, University of Szeged,
6724 Szeged, Hungary
E-mail: sarosi@mk.u-szeged.hr