THE MOBILE ROBOT CONTROL FOR OBSTACLE AVOIDANCE WITH AN ARTIFICIAL NEURAL NETWORK APPLICATION

Victor Andreev, Victoria Tarasova

This Publication has to be referred as: Andreev, V[ictor] & Tarasova, V[ictoria] (2019). The Mobile Robot Control for Obstacle Avoidance with an Artificial Neural Network Application, Proceedings of the 30th DAAAM International Symposium, pp.0724-0732, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-22-8, ISSN 1726-9679, Vienna, Austria

DOI: 10.2507/30th.daaam.proceedings.099

Abstract

The article presents the results of the research, the purpose of which was to test the possibility of avoiding obstacles using the artificial neural network (ANN). The ANN functioning algorithm includes receiving data from ultrasonic sensors and control signal generation (direction vector), which goes to the Arduino UNO microcontroller responsible for mobile robot motors control. The software implementation of the algorithm was performed on the Iskra Neo microcontroller. The ANN learning mechanism is based on the Rumelhart-Hinton-Williams algorithm (back propagation).

Keywords: Microcontroller; Arduino; ultra-sonic sensor; mobile robot; neural network; autonomous robot.

1. Introduction

One of the most relevant tasks of modern robotics is the task of creating autonomous mobile robots that can navigate in space, i.e. make decisions in an existing, real environment. Path planning of mobile robots can be based on sensory signals from remote sensors [1]. Local planning is an implementation of this process. At present, the theory of artificial neural networks (ANNs) is increasingly being used to design motion control systems for mobile robots (MR).

Artificial neural networks (artificial neural networks, ANN) - a set of interacting artificial neurons or a set of mathematical and algorithmic methods for solving a wide range of problems. Artificial or formal neuron (FN) (artificial neuron) is a mathematical abstraction, a simplified model of a biological neuron.

Neural networks are used in many areas where tasks such as:

- recognition of visual, auditory images;
- associative seek information and the creation of associative models, speech synthesis, the formation of a natural language;

- the formation of models and various nonlinear and difficult to describe mathematically systems, forecasting the development of these systems over time;
- application in production, forecasting the development of cyclones and other natural processes, forecasting changes in exchange rates and other financial processes;
- predictive control and regulation systems, control of robots, and other complex devices;
- a variety of machines: public service systems and switching systems, telecommunication systems;
- decision-making and diagnostics, excluding a logical conclusion, especially in areas where there are no clear mathematical models: in medicine, forensics, financial sphere.

A unique feature of ANN is versatility. For all of the above problems, there are effective mathematical methods for solving and despite the fact that ANNs lose to specialized methods for specific problems, due to their universality and perspectiveness for solving global problems, for example, constructing artificial intelligence and modeling the thinking process, they are an important area of research that requires careful studying.

The advantages of ANN are:

1. **Learning.** By creating a network and performing the learning algorithm, you can train the network to solve the desired problem.
2. **The ability to generalize.** After training, the network becomes insensitive to small changes in the input signals (noise or variations of the input images) and gives the correct result at the output.
3. **Ability to abstract.** If you give the network with several distorted variants of the input image, then the network itself can create an ideal image at the output that it has never met.

In the article [2] it was shown that the analysis of the main results of research conducted around the world, in the field of robotics, made it possible to form the following groups for intelligent control:

- intelligent management of industrial facilities and production systems;
- creation of intelligent control systems for mobile objects for various purposes and vehicles of ground-based, underwater and air-based;
- development of tools and methods for controlling intelligent robots in special, industrial, medical, domestic and other applications;
- development and creation of specialized hardware for intelligent control systems.

In [3], an ANN based on the neurobiological model of Valtsev VB [4] is used to control a mobile robot (a guide robot). Three signals are received at the network input - from an ultrasonic obstacle sensor (ultrasonic sensor), a battery charge level sensor and a sensor that determines the accuracy of robot positioning in the lane. It is assumed that the signals have different priorities or significance (the signal from the obstacle sensor has the highest priority - safety requirement). It is necessary to determine what task the MR should solve at the moment (reaction to an obstacle, positioning on the line or starting the reaction procedure to reduce the battery discharge), i.e. dynamic planning of robot behavior. The experiments carried out in [3] show the fundamental possibility of using ANNs to solve control problems, but as indicated in the article, this method has technical difficulties associated with the need for quick modeling and the laboriousness of the process of creating and configuring a network.

In [5], [6], a mobile platform is used with an IP camera placed on it (for determining the direction of the target point), ultrasonic and infrared (IR) sensors (detection of obstacles), a magnetometer and accelerometer (orientation on the ground). The Arduino Duemilanove microcontroller is responsible for controlling the movement of the platform, receiving commands from the controlling personal computer (PC) via the Bluetooth module. A neural network in the form of a multilayer perceptron is implemented on a PC and calculates the direction of motion of the MR based on the data received from the camera. The direction information is sent to the Arduino via the Bluetooth communication channel, where, based on this information and information from the sensors, using the fuzzy logic methods, the microcontroller calculates the motion parameters and generates control commands. This algorithm has been successfully implemented, but its performance depends on the stability of the communication channel.

In [7], the authors propose associating a ready-made ANN library containing training using the back-propagation error algorithm with Arduino to implement climate control in the incubation chamber, which has shown its effectiveness and ease of implementation.

2. Using a neural network to integrating data from a group of ultrasonic distance sensors

In [8] [9] an approach was considered in which not a single control and sensory information processing center is used, but a distributed control system in which the general task is divided into sub-tasks according to the functional-modular

principle, distributing them between the modules. This approach provides a significant reduction in the computing power requirements of the multiprocessor system calculators and an increase in the overall system speed.

The purpose of this work: to investigate the possibility of using an ANN to perform data integrating from ultrasonic sensors for avoid obstacles by a mobile robot provided that a distributed control system is used: one microprocessor is responsible for the operation of the ANN, and the second for the operation mode of the wheeled platform electric motors. In this case, the ANN is included in the feedback of the MR motion control system, which distinguishes this solution from the standard when a special filter is used in the feedback of the control system.

Used equipment.

For experiments, we used a wheeled platform with a sensor system consisting of three simultaneously firing ultrasonic sensors HC-SR04. The neural network is implemented on the Iskra Neo microcontroller [10]. Data from the sensors arrives at the input of the neural network in the form of an input vector $X(x_1, x_2, x_3)$, where x_1, x_2, x_3 – these are the values obtained from the first, second and third ultrasonic sensors, respectively. At the output of the neural network, control signals are generated that are transmitted via the serial UART interface to the microcontroller Arduino Uno [11], responsible for engine management. To train the neural network, a vector of output values $Y(y_1, y_2, y_3)$, is created, which is taken depending on the readings of the ultrasound sensors. The following combinations of output vector values were used:

- (1, 0, 0) – right turn;
- (0, 1, 0) – movement straight;
- (0, 0, 1) – turn left.

In these experiments, training was selected using Rumelhart-Hinton-Williams algorithm (back propagation) [12]. It was proposed in various variations in several scientific papers; there are also many improved versions of the algorithm.

The back-propagation method is a way to quickly calculate the gradient of the error function. The calculation is made from the output layer to the input one according to recurrence formulas and does not require recalculation of the output values of neurons. Knowing the gradient, one can apply many methods of optimization theory using the first derivative. Quasi-Newtonian methods are also applicable, with the help of which a matrix of second derivatives is constructed on the basis of several successive gradient values. A quick calculation of the gradient is necessary in many optimization methods (training), so the value of the backpropagation algorithm in the ANN theory is large.

The advantages of the backpropagation method are as follows:

- Quite high efficiency, only information about connected neurons passes through each neuron, so back-propagation is easily implemented on computing devices with parallel architecture
- High degree of generality: the algorithm is easy to write for an arbitrary number of layers, an arbitrary dimension of outputs and outputs, an arbitrary loss function, and arbitrary activation functions, possibly different for different neurons
- The ability to apply in conjunction with various gradient optimization methods: the fastest descent, conjugate gradients, Newton-Raphson, etc.

But the method inherits the known disadvantages of the gradient adjustment of weights in the perceptron, problems of slow convergence arise, such as:

- Network paralysis: if one of the weights during training gets too large a value, then with the usual values of this input, the neuron output will be saturated, i.e. will be close to the limit value of the activation function. The output of the neuron will depend little on the weight coefficient w , and therefore, the derivative $\frac{\partial \delta}{\partial w} \approx 0$. Training on this weight will be very slow because the change in weight is proportional to the derivative. The output signal of the neuron will depend little not only on the weight, but also on the input signal x_i of the given neuron, and the derivative with respect to x is involved in the back propagation of the error. Consequently, preceding neurons will also train slowly. This slowdown in learning is called network paralysis
- The step size: a poor choice of stride length can result in:
 - to inaccuracy of training: being in the vicinity of a local minimum, when small step lengths are required for fine-tuning the parameters, the algorithm with a large step will give inaccurate parameter values
 - to slow learning: if the step is too small, learning can become unacceptably slow
 - lack of convergence, network paralysis and other problems with a very long stride length
- Local lows: like any gradient algorithm, the back propagation method “gets stuck” in the local lows of the error function, because the gradient near the local lows tends to zero

The general solution algorithm is as follows [13]:

1. Determine what will constitute the input vector X of the neural network. The input vector must contain a formalized condition of the problem, i.e. sufficient information necessary to obtain the desired response. In our task of integrating data from sensors to provide control signals for MR movement, the input vector represents the readings from three simultaneously firing ultrasonic sensors
2. For training a neural network, it is necessary for each input vector X to select the desired reaction corresponding to it in the form of an output vector Y . In this task, depending on the readings of ultrasound sensors, one of three combinations of values of the output vector was chosen:
 - $(1, 0, 0)$ – right turn
 - $(0, 1, 0)$ – movement straight
 - $(0, 0, 1)$ – turn left
3. Select the type of nonlinearity in neurons (activation function). Most often, the type of nonlinearity does not fundamentally affect the solution of the problem. However, a good choice can reduce training time by several times. The choice of activation function is determined by:
 - specifics of the task
 - ease of implementation on a computer, in the form of an electrical circuit or in another way
 - learning algorithm: some algorithms impose restrictions on the type of activation function, they must be taken into account

In order to be able to use the back propagation method of error (which is used in our task to train ANN), the activation function of neurons must be differentiable. As an activation function, a logistic function or an S-shaped function (sigmoid) was used:

$$f(x) = \frac{1}{1 + e^{-ax}}. \quad (1)$$

The sigmoid function has a simple expression for the derivative of the function (this can significantly reduce the computational complexity of the error back propagation method, making it applicable in practice).

$$f'(x) = f(x) \cdot (1 - f(x)) \quad (2)$$

4. Select the number of layers and neurons in the layer. There is no strictly defined procedure for choosing the number of neurons and the number of layers in the network. The larger the number of neurons and layers, the wider the network capabilities, but the slower the network learns and works and the more non-linear the input-output dependence can be. In this experiment, three layers of ANN were used (input, hidden and output), where 3 neurons were used in each of the layers
5. Also, for training the network, it is necessary to assign the initial values to the weights. The initial values should not be large so that the neurons are not saturated (on the horizontal section of the activation function), otherwise training will be very slow. The initial values should not be too small so that the outputs of most of the neurons are not equal to zero, otherwise learning will also slow down
6. Training, i.e. select weights so that the problem is solved in the best way. At the end of the training, the network is ready to solve the problems of the type with which it is trained
7. Check network operation. Submit the vector X to the network input, and check the resulting output vector Y

Thus, in this experiment, the following neural network architecture was used (Fig. 1):

- Three layers (input, hidden and output), 3 neurons in each of the layers
- The neurons of each layer are connected according to the principle of "each with each"
- At each of the three inputs of the neural network, readings are received from one of the three ultrasonic distance sensors, respectively
- The output of a neural network is the direction vector of further movement

Fig. 1. The structure of the used ANN

To train a neural network, preparation of a training set is necessary. Data supplied to the network input and taken from the output must be properly prepared. If the readings come from various sensors, it is necessary to bring the data into the acceptable range using, for example, scaling. In this experiment, only ultrasonic sensors are used, therefore, additional reduction of data to one range is not necessary. Data collection for training was as follows. Figure 2 shows the MR sensor system, consisting of ultrasound sensors located as follows: the distance between the sensors is $d=3$ cm, the distance to the object L is variable. Sensors start simultaneously.

Fig. 2. a) an arrangement of ultrasonic sensors; b) the location of the sensors on the MR

Data was taken at the following MR and object positions (the object in the experiment was a cardboard box):

1. Before the MR at a distance L we place the object (Fig. 3). Change L in the range:
 - 10 - 18 cm with step 4, for each L we measure 50 measurements - at these values, the MR should turn right
 - 20 - 145 cm with step 25 cm, for each L we measure 50 measurements - at these values the MR should go straight

Fig. 3. Object location in front of sensors

2. We position the object to the left of the MR at a distance L (Fig. 4). Change L in the range:
 - 3 - 18 cm with step 5 cm, for each L we measure 50 measurements - at these values, the MR should turn right
-

- 20 - 70 cm with step 10 cm, for each L we measure 50 measurements - at these values the MR should go straight

Fig. 4. Location of the object to the left of the sensors

3. We position the object to the right of the MR at a distance L (Fig. 5). Change L in the range:
 - 3 - 18 cm with step 5 cm, for each L we perform 50 measurements - at these values, the MR should turn left
 - 20 - 70 cm with step 10 cm, for each L we carry out 50 measurements - at these values the MR should go straight

Fig. 5. The location of the object to the right of the sensors

4. We position the object at an angle to MR at a distance L (Fig. 6). Change L in the range:
 - 3 - 18 cm with step 5 cm, for each L we perform 50 measurements - at these values, the MR should turn left
 - 20 - 50 cm with step 10 cm, for each L we carry out 50 measurements - at these values the MR should go straight

Fig. 6. Object angle

5. We position the object at an angle to MR at a distance L (Fig. 7). Change L in the range:
 - 3 - 18 cm with step 5 cm, for each L we perform 50 measurements - at these values, the MR should turn right
 - 20 - 50 cm with step 10 cm, for each L we carry out 50 measurements - at these values the MR should go straight

Fig. 7. Object angle

ANN learning algorithm:

1. Arbitrary generated weights for a multilayer network are set in the range $[-0,5; 0,5]$
2. The input vector X is randomly input to the input. The direct propagation through the network is calculated (weighted sum s and activators $f(s)$ for each cell are determined), the network output vector is $Y_{out}(y_{out,1}, y_{out,2}, y_{out,3})$ is

compared with the output vector $Y(y_1, y_2, y_3)$ corresponding to the input vector X . Evaluation of the network is performed by the standard error on the output s :

$$s = 0,5 \cdot \sum_{j=1}^n (y_{out,j} - y_j)^2 \quad (3)$$

where j – is the serial number ($j= 1..n$), n – is the number of output cells. If $s < 0,001$, then go to step 5 , otherwise, perform the next step (p. 3)

3. Starting from the outputs, the back error δ moves through the cells of the output and intermediate layer:
 - for output cells:

$$\delta_{0j} = \sum_{j=1}^n y_{Bылx,j} - y_j \quad (4)$$

where δ_0 – is the error at the output cell.

- for hidden cells:

$$\delta_i = \sum_{j=1}^n \sum_{i=1}^k (w_{ij} \cdot \delta_{0j}) \cdot s_i \cdot (1 - s_i) \quad (5)$$

where δ – is the error in the hidden cell, s – is the state of the neuron, k – is the number of neurons in the hidden layer.

4. Net weights are updated as follows:

- weight of the connections between the hidden layer and the output:

$$w_{i,j} = w_{i,j} + \rho \delta_{0j} s_j \quad (6)$$

- weight of connections between hidden layer and input:

$$w_{i,j} = w_{i,j} + \rho \delta_i s_i \quad (7)$$

where ρ – is the learning coefficient (or step length). There are several methods for choosing this option.

The *first* way is to start the educational process with a large value of the parameter ρ . During the correction of weights, the parameter is gradually reduced. The *second* is a more complex learning method, it starts with a small parameter ρ . In the learning process, the parameter increases and then decreases again at the final stage of training. The beginning of the educational process with a low value of the parameter ρ allows us to determine the sign of the weight coefficients. In this experiment, the parameter ρ was constant, $\rho = 0,01$.

5. Go to p. 2.
6. ANN is trained until the global error in all samples ceases to be significant. The obtained weighting coefficients are substituted into each example of the training sample in series and the total network error δ_1 , which tends to a minimum, is calculated, in this work, when $\delta_1 < 70$ the cycle ends, when $\delta_1 > 70$ we go to p. 2.

After completing the training cycle, we obtain 18 weighting coefficients, which are set in the microcontroller Iskra Neo neural network.

The algorithm of the ANN on the microcontroller:

1. Weights are set in microcontroller as variables of type float
2. Measurements from ultrasound sensors come to the input of the ANN
3. Direct distribution through the network is calculated
4. From the output values, the value with the largest sum of weights is selected according to the principle “the winner gets everything”: the cell with the largest sum of weights is the “winner” of the group, the remaining cells are zeroed and at the output we get the output vector (turn right, turn left, right), on the basis of which control signals are sent to the Arduino Uno.
When turning, the MR turns until the control signal changes at the output of the neural network. Values are taken from the sensors every 50 ms
5. The cycle repeats from point 2

3. The results of the experiments

The purpose of the experiments described below is to experimentally prove the possibility of using a neural network to detect relatively simple objects, with a minimal sensor system consisting of only a few simultaneously firing ultrasonic distance sensors. The results of experiments on the detection of objects using ultrasonic sensors were described in articles [14], [15].

The following objects were used as the simplest forms: a flat wall and objects at an angle. external (internal) corner formed by walls.

1. *Detection of an object like "flat wall".*

The results are shown in Figure 8.

Fig. 8. Robot trajectory

The robot came too close to the obstacle before starting to turn. This can be changed by setting the requirement in the training set: start turning at a greater distance to the object.

Conclusion: MR with ANN detected an obstacle and successfully avoided a collision.

2. *Detection of an object of the "flat wall" type, oriented at a certain angle to the direction of movement of the robot.*

The results are shown in Figure 9.

Fig. 9. Robot trajectory

Conclusion: sometimes the MR turns in the wrong direction, but this can be adjusted by changing the training sample.

3. *Detection of an object with small dimensions.*

The results are shown in Figure 10.

Fig. 10. Robot trajectory

Conclusion: MR detects and avoids an obstacle.

4. Conclusion

The use of ANN in the feedback circuit of the motion control system of an autonomous mobile robot allows solving the problem of safe MR movement in a non-deterministic environment. When developing a conventional logical filter, it is not possible to foresee all possible options for the interaction of MR with the environment. Using ANN makes it possible to train a management system on decision making, even if not optimal. Experimental studies have shown that the use of ANNs to integrating the readings of ultrasonic distance sensors combined into a group successfully solves the problem of detecting obstacles for MR. The use of a group of simultaneously firing ultrasound sensors and the use of ANN not only increases the reliability of obstacle detection, but also solves the problem of detecting obstacles with small dimensions. In addition, training ANN allows you to automatically develop rules for avoiding obstacles. Separation of the MR control system by functional feature allows you to use a modular approach to implement functions on individual microprocessors of low productivity. In other words, this approach makes it possible to create a distributed control system, thereby parallelizing the general MP control algorithm.

5. Summary

In the future, it is planned to complicate the neural network by adding additional remote sensors and internal state sensors. Implement a learning algorithm without a teacher for automatic selection of coefficients depending on the speed of the robot and the environment. Go to more complex networks to build a map of the area.

6. Acknowledgments

The Russian Foundation for basic research supports this research: Grant 19-07-00892a.

7. References

- [1] Ogur, M.G. (2015). The main problems of planning the behavior of mobile handling robots. Actual problems of modern science, Vol. 2, № 4, pp. 189-192
- [2] Lokhin, V.M. (2015). The development of intelligent control technologies at the Institute of Cybernetics MIREA. Bulletin of MSTU MIREA, Vol. 1, № 3, pp. 30-40, 2313-5026
- [3] Karpov V. E & Valtsev V. B (2009). Dynamic planning of the behavior of robots based on a network of "smart" neurons, Artificial Intelligence and decision making, №2, pp. 58-69
- [4] Valtsev, V. B.; Grigoriev, V. R.; Nikonov V. G.; & Kobzar, I. V. (1996). Brainputer and Neural Architecture of Neurocomputers, Vol. 1, № 1, pp. 54-59
- [5] Zotin A. G. & Sayupin A. V. (2016). Educational distributed control system for a mobile wheeled platform using video and sensory information. Programmnye produkty i sistemy, № 1, pp. 146-151
- [6] Zotin A. G. & Sayupin A. V. (2013). Development of a semi-autonomous research system using video data and sensor evices. Cshctnevskie chteniya, Vol. 2, № 17, pp. 203-204
- [7] Kurinka V. S. (2017). Synthesis of a neural network on an arduino microcontroller for efficient control in a micro-incubation chamber. Innovation-2017, pp. 64-68
- [8] Andreev, V. & Poduraev, Y. (2016). Network-Based Design of Heterogeneous Modular Mobile Robotic Systems, Proceedings of the 27th DAAAM International Symposium, pp.0004-0009, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-08-2, ISSN 1726-9679, Vienna, Austria. DOI:10.2507/27th.daaam.proceedings.001
- [9] Andreev V. (2018). Control System Mobile Robots with Modular Architecture as a Multi-Agent System with a Hierarchical Topology, Proceedings of the 30th DAAAM International Symposium, pp.xxxx-xxxx, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-xx-x, ISSN 1726-9679, Vienna, Austria.
- [10] <http://wiki.amperka.ru/%D0%BF%D1%80%D0%BE%D0%B4%D1%83%D0%BA%D1%82%D1%8B:iskra-neo> (2019). Iskra-neo, Accessed on: 2019-09-25
- [11] <http://arduino.ru/Hardware/ArduinoBoardUno> (2019). Arduino Uno, Accessed on: 2019-09-25
- [12] Zaentsev, I.V. (1999), Neural networks: basic models, Voronezh
- [13] Vasilenkov D.V. (2007). Training methods for artificial neural networks. Computer Tools in Education, No. 1
- [14] Andreev V. & Tarasova V. (2017). Determination of the Form of Obstacles by a Mobile Robot Using Scanning Angular Movements of Ultrasonic Sensor. Mechatronics, Automation, Control, Vol. 18, No. 11, pp. 759 –763, DOI: 10.17587/mau.18.759-763
- [15] Andreev, V. & Tarasova, V. (2018). Identification of the Obstacle Shape Using the Ultrasonic Sensors Module of Modular Mobile Robot, Proceedings of the 29th DAAAM International Symposium, pp.1001-1009, B. Katalinic (Ed.), Published by DAAAM International, ISBN 978-3-902734-20-4, ISSN 1726-9679, Vienna, Austria. DOI: 10.2507/29th.daaam.proceedings.143