Development of intelligent behavior using decision making methods based on ANN

Branko Miloradović, Svemir Popić, Ivan Stojković
Center of Robotics
Mihailo Pupin Institute
Belgrade, Serbia
branko.miloradovic@pupin.rs; svemir.popic@pupin.rs; ivan.stojkovic@pupin.rs

Abstract— ***This paper presents one of possible approaches in replacing conventional way of internal transport in manufacturing environments. New, improved solution includes the use of intelligent mobile robot for transport of materials. This is achieved by implementing motion model and using algorithms for planning and selection of optimal trajectory. Epithet "intelligent" is deserved for using artificial neural networks (ANN) for recognition of existing obstacles and making a decision about their successful avoidance. Experimental results, conducted on Khepera II mobile robot, have fulfilled expectations and shown that this kind of solution has practical use in today's modern technological environments.***

Keywords-component; intelligent mobile robot; artificial neural networks; decision making methods; visibility graph; obstacle avoidance;

I. INTRODUCTION

This paper shows the solution of engineering problems in the domain of an intelligent agent's behavior – mobile robot for modeling of internal transport of raw materials and finished parts in a manufacturing environment. Design solution is implemented on *Khepera II* mobile robot (Fig. 1). For the purpose of mobile robot's motion it is necessary to develop and implement software solutions, based on the decision-making methods, which will, with a pre-set accuracy, perform appropriate mapping, defined by the choice robot should make.

Figure 1. Khepera II mobile robot

Given the complexity of above problem which involves avoiding obstacles using only information from the infrared (IR) sensors, developed solution is based on machine learning and application of ANN. The aim is to improve the internal transport by replacing the current way of transport with a mobile robot. Robot's tasks would be to transport additional materials, work pieces, tools, etc. between machine tools, warehouses, conveyor belts and storages for components within the plant.

II. ESTIMATION OF THE MOBILE'S ROBOT POSE

A fundamental goal in mobile robotics is to have the right knowledge about the pose of the robot at each given moment. To perform a specific task, which requires the robot to move, it is necessary to solve the problem of its motion in working space. Determining the pose of mobile robot in the working environment consists of finding the parameters that determine its position and orientation. Coordinates of the mobile robot in previously defined absolute coordinate system represents a position while rotation angle around Z-axis represents orientation of the mobile robot. Number of positions and orientations robot can achieve, directly depends on the number of degrees of freedom (DOF). In the general case, a mobile robot can have 6 DOF, where the state vector is described as follows:

$$x_t = (\chi, \upsilon, \zeta, \theta, \psi, \phi) \quad (1)$$

In this paper we use a simplified case, because it is a mobile robot performing planar motion (Fig. 2), which means that the movement is performed in the X and Y plane, and the angle θ relates to the rotation of the robot around Z axis. Taking this into account, the equation is reduced to the following form:

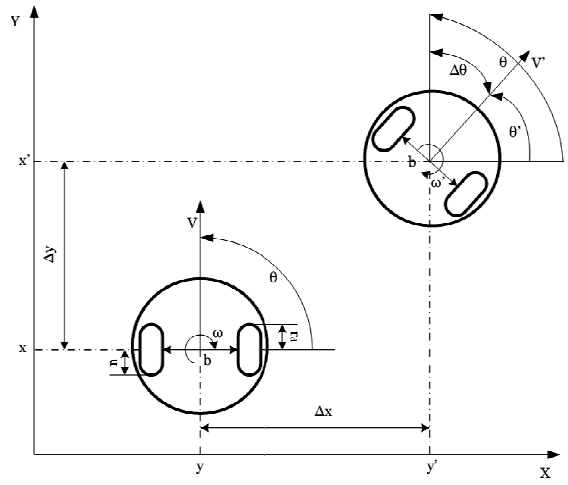

Figure 2. Representation of intelligent robot in two consecutive moments

$$x_t = (\chi, \upsilon, \theta) \quad (2)$$

Motion model, which is used in this study, calculates the position of the robot based on the distance it traveled, i.e. using odometric localization – odometry. Odometry is based on a fact that the rotation of the wheels can be translated into linear displacement relative to a flat surface on which the mobile robot moves. Mathematical model of motion, based on odometry, is presented as follows:

$$x' = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_d + \Delta s_l}{2} \cos\left(\theta + \frac{\Delta s_d - \Delta s_l}{\frac{b}{2}}\right) \\ \frac{\Delta s_d + \Delta s_l}{2} \sin\left(\theta + \frac{\Delta s_d - \Delta s_l}{\frac{b}{2}}\right) \\ \frac{\Delta s_d - \Delta s_l}{b} \end{bmatrix} \quad (3)$$

Where: Δs_d, Δs_l, b – represents distance traveled by the right wheel, distance traveled by the left wheel and distance between wheels, respectively. Knowing the encoder resolution and the wheel radius we can, at any given moment, determine the distance robot traveled. Substituting these values in (3), we obtain the values of x, y coordinates and angle θ.

III. Optimal trajectory – Planning and navigation

In the case of mobile robots, a special aspect of cognition, directly related to the mobility of robot, is navigational competence. If robot has partial knowledge about its environment, position of the goal or series of positions, navigation of mobile robots includes ability to act in accordance with the prior knowledge and sensory information in order to reach the desired position as efficiently and reliably as possible.

So far, the mobile robotics community suggested many approaches to solve this problem. All approaches, though they do not seem so, are essentially similar. The main difference is in decomposition. There are two key requirements for successfully solving the problem of mobile robot navigation. Knowing the map and the position of the goal, path planning is the identification of trajectory that will enable the mobile robot to reach the desired position, i.e., path planning of robots is the ability to generate a plan of its own motion in order to perform a specific task.

Since it required movement of the robot from start to finish, without a collision with objects and obstacles that can be found on its path, taking into account readings from sensors in real-time, obstacle avoidance involves changing trajectory in order to evade collision with objects in the environment. In order to successfully achieve this, robot has to react based on sensory readings. However, planning and reacting seems like opposite approaches but, in fact when it comes to physical systems, such as intelligent mobile, planning and reacting have strong complementarity [1]. There are two basic paradigms for path planning with obstacle avoidance:

1) A global approach which generates a path in *off-line* mode using known information about the environment.

2) The local approach is done in *on-line* mode using a small number of known information about the area, but there is an essential need for constant acquisition of new data through sensors.

We come to the conclusion that with no reaction robot cannot reach its goal, in the case of unforeseen events, and without planning only reacting cannot control overall behavior of the robot and guide it to the desired goal. In the best case scenario, if unexpected changes occur in environment, robot will change its behavior only at local level in order to correct a predefined trajectory.

First step in solving the problem is to simplify robot, which has a complex geometric shape, into the point, in the new abstract space called the configuration space [2],[3]. With this we transform the original problem into problem of trajectory planning for motion point. After that is done we do a discretization of continuous configuration space and we construct connected graph. At the end, the graph is searched in order to find the optimal trajectory for the robot to reach its goal. Block diagram of the planning process is shown in Fig. 3. The output from this process is a trajectory from initial to the final position.

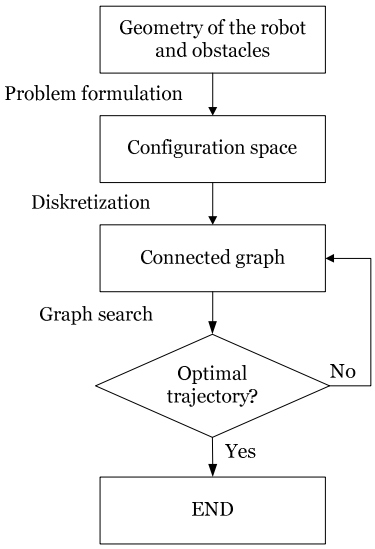

Figure 3. Block diagram of the planning process

There are many approaches to solving the problem of planning the path. One of the possible approaches is the *Road map* algorithm which represents a group of algorithms used for path planning. All approaches are based on establishing a network of possible trajectories from the start to the target position. In this paper a road map algorithm is used, namely the visibility graph. It is commonly used in two-dimensional configuration space with polygonal obstacles. It is obtained by connecting all obstacle visible vertices with the initial position of the robot and targeted position. Two vertices are mutually visible if there are no obstacles between them and the line connecting those two points can be taken as a potential part of the trajectory.

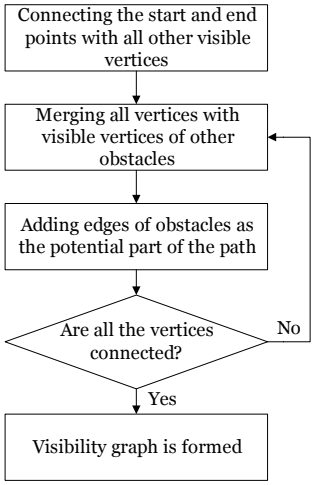

Figure 4. Block diagram of the formation of the visibility graph

After getting physically possible trajectories through which robot can move, it is necessary to determine not only the trajectory that leads from start to finish, but the trajectory must be optimal[1]. Before searching can start, it is necessary to solve the main problem of the visibility graph. The optimal trajectory obtained by applying the visibility graph (Fig. 4) tends to take the robot as close as possible to the obstacle on the way to the goal position. There are two general ways to solve this problem:

1) To increase the size of the obstacles at least for the radius of robot or more [4].

2) To perform correction of the given trajectory.

In this paper, the solution to the problem is derived by using the first method mentioned above, but the problem is further simplified as only obstacles that are relevant to the movement of mobile robot succumb to resizing. In Fig. 5, the green color is an extension of obstacles, while the red and gray represents true dimensions.

The main disadvantage of the visibility graph is that by increasing the number of obstacles, number of edges and vertices also increase and thus the number of trajectories to be examined. Also, the shape of obstacles should not be with more than 8 vertices. Model of manufacturing environment in this paper is simplified, the number of obstacles is small and generally rectangular shape which allows very easy and effcient implementation of visibility graph. More about graph theory can be found in [5].

After the formation of the graph, it must be examined in order to find the optimal path. One of possible search algorithms for the shortest path in the graph is Dijkstra's algorithm [6]. The algorithm in this paper is a slight variation of Dijkstra's algorithm. The difference is that in Dijkstra's algorithm distance, between every node[2] is calculated, while algorithm used in this paper first calculates all possible trajectories and their values and then determines which one is optimal (Fig. 5).

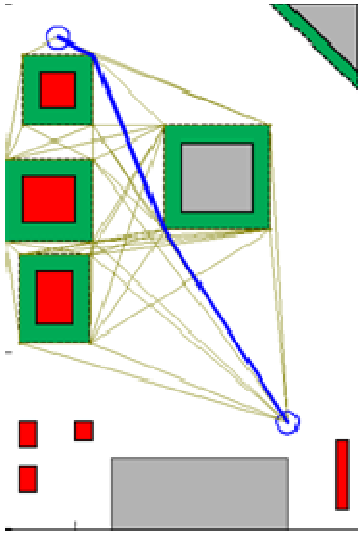

Figure 5. Model of the manufacturing environment with optimal trajectory

By defining motion model we insure we have the information about the position of the mobile robot at any time. Selecting algorithms for solving the path planning and navigation problem provided the successful planning and execution of movement of mobile robot from starting positon to the finish. Using visibility graph and altered Dijkstra's algorithm for graph search provided that selected trajectory is also optimal. Now it is necessary to ensure the robustness of the system, so that intelligent mobile robot, which has a pre-defined path and order of accessing machines, will be able to avoid obstacles in the environment and to continue performing its task.

IV. OBSTACLE RECOGNITION AND AVOIDANCE

Infrared sensors are used for object recognition. However, for decision making process it is necessary to introduce an artificial neural network, which in this case has a task to identify disturbances that occurred in the system and to determine the way to overcome them. Application of artificial neural networks have a broad representation in robotics, mainly because of their ability to find a correlation between the input and desired output data. Method used in this paper is based on machine learning, with the application of artificial neural networks, because of their ability to adapt to changes in the system caused by perturbation factors and based on that minimize the error after each subsequent iteration.

A neural network is a paradigm of artificial intelligence which is defined as the connective model of reasoning based on an analogy with the brain, while stressing the cognitive ability to learn and perform generalization of acquired knowledge [7]. Two types of neural networks are used: *backpropagation* (BP) and *radial bias function* (RBF). BP neural network has been developed to solve the problem of nonlinear mapping from the input space to the output space, while doing modification of the weights between input and hidden layer. BP is feedforward neural network which also

[1] In this paper optimal trajectory is based on the criterion of the Euclidian distance between two points.

[2] The term node, in the graph search algorithm, refers to the top of obstacles

implements a supervisory type of learning, with different activation functions and learning algorithms [8].

In this paper, standard Levenberg – Marquardt learning algorithm is used:

$$W^{(k+1)} = W^k - \mu(H - \lambda diag(H))^{-1}\nabla J(W) \quad (4)$$

Tan-sigmoid activation function was used.

In RBF networks we have to determine the function of the hidden layer, the number of processing elements and the training algorithm for finding the network parameters [9,10]. RBF are feed-forward networks and may use different activation functions, usually Gaussian, which is given by (5).

$$\Phi_j(X) = \exp\left[-\frac{1}{2}(X-\mu_j)^T\Sigma_j^{-1}(X-\mu_j)\right] \quad (5)$$

BP networks in this paper were used to identify the type of obstacle based on information from sensors, while the RBF networks are used to determine the distance to the obstacle by using the value of the infrared sensors. Since Khepera II has eight infrared sensors, consequently we have eight RBF networks, one for each sensor. During the acquisition of data required for training the BP network, 1000 measurements per sensor were made, ie. 100 measurements for each of the 10 typical cases. For the RBF networks, measurements were made at distances of 1 to 2 [cm] in steps of 2 [mm], and at a distance of 2 to 5 [cm] step was 0.5 [mm]. The total number of measurements is 9600. The networks are trained with only 50% of the collected data, while the rest is used for their testing and verification of performed training.

Given that we have different readings for different distances from the robot to the obstacle, which is normal, two ANN are introduced for making decisions regarding obstacle avoidance. One ANN is trained for the case when robot is at a distance of 2 [cm] and the other when robot is at 4 [cm] to the obstacle. The success rate of recognizing obstacles at 4 [cm] distance is not satisfactory and the system cannot rely only on the decisions that this network makes. The primary purpose of this network is the early detection of an obstacle, after which robot decreases its speed in order to be better prepared to perform readings at 2 [cm]. After these readings robot is able to make decisions and decide how to rotate to avoid obstacle that is placed in front of it. Accuracy is described in detail with decision matrix (Table 1).

TABLE I Decision making matrix

	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
Y1	83.34	3.33	0	0	0	0	0	0	0	0
Y2	13.33	93.34	0	0	0	0	0	0	0	0
Y3	0	3.33	96.67	0	3.33	0	0	0	0	3.33
Y4	0	0	0	80.01	0	0	3.33	0	0	0
Y5	0	0	0	0	96.67	19.99	3.33	3.33	0	0
Y6	3.33	0	0	0	0	80.01	0	0	0	0
Y7	0	0	0	19.99	0	0	90.01	16.66	3.33	0
Y8	0	0	3.33	0	0	0	3.33	80.01	0	0
Y9	0	0	0	0	0	0	0	0	76.68	23.33
Y10	0	0	0	0	0	0	0	0	19.99	73.34

The flow of information within the system begins by entering the start and the end coordinates of the robot. If they are physically achievable, visibility graph is formed, searched and optimal path is determined on the criterion of shortest distance. After that we determine the position and orientation of the robot at every point robot must pass through in order to reach its goal. If during the motion, the robot detects an unknown obstacle, first it checks to see if the robot is far enough from the obstacle to safely circumvent it. If this condition is not met, the robot returns back to a safe distance and then runs ANN, which should based on sensory information, make a decision how to overcome this unknown obstacle. This entire process is repeated until the robot reaches the desired position.

V. EXPERIMENTAL RESULTS

For the control of the intelligent mobile robot and training of the ANNs, MATLAB® software package was used. All codes are developed specifically for the Khepera II, while other basic functions are provided with kmatlab toolbox.

Fig. 6 shows the 10 typical cases that the BP network is trained to recognize. Table 1 shows the performance of obstacle detection for each of the 10 typical cases. For obstacle detections sensors have been used only on the front side of the robot. That is why most of the results are shown for only front six sensors.

Columns in Table 1 represent possible reaction, while rows represent number of possible types of incentives. Once these values are known decision matrix of the observed agent can be described. On its basis it is easy to see for what cases ANN gives the best results.

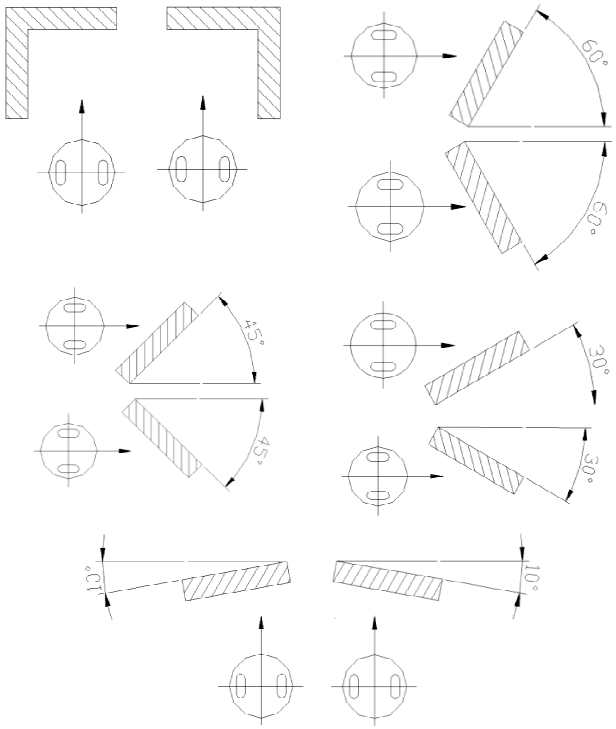

Figure 6. Sketches of typical obstacles

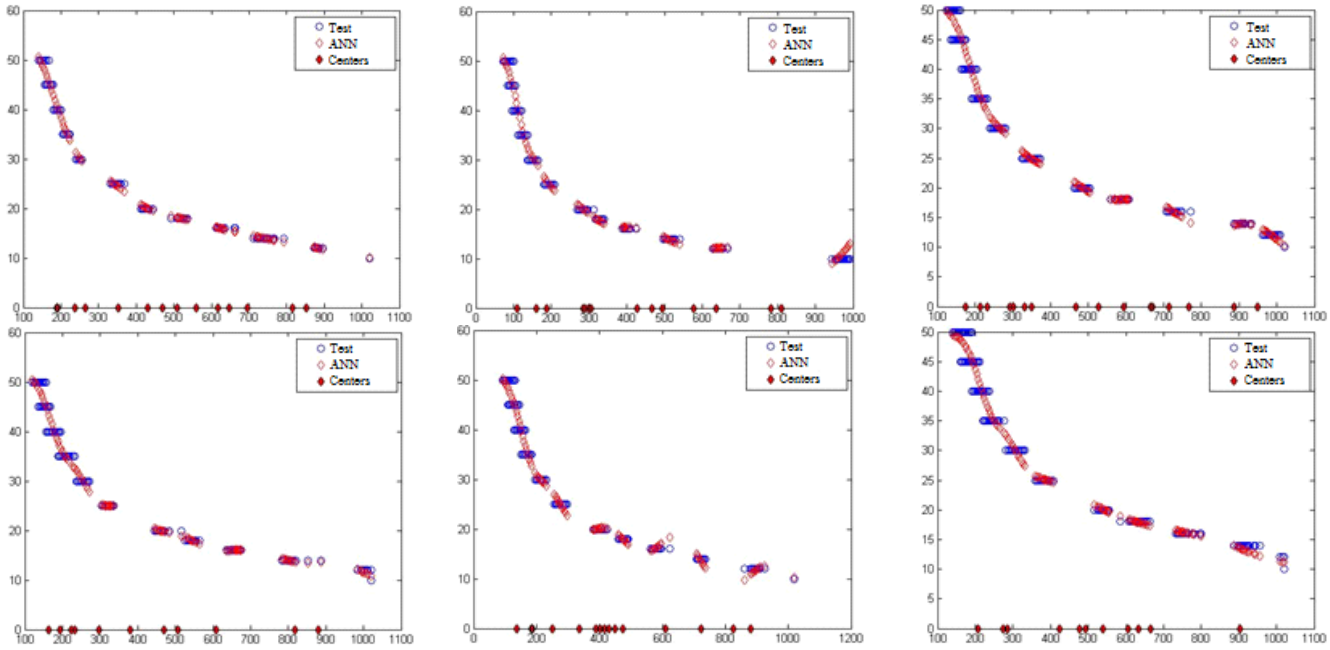

Figure 7. Graphical display of results for training with the training set and test set. X axis - readings from the sensors, and the Y axis distance in [mm]. First line sensors 1, 2, 3 and 4, 5, 6 others.

It is clearly seen that the highest percentage of failed attempts is in cases 9 and 10. It must be emphasized that this does not necessarily mean that ANN is trained wrong. As the differences between these two cases are quite small, precision of the sensors plays a big role in making the right decision. The best results are obtained when the value of learning parameter was 0.0001.

TABLE II Average root mean square error is expressed in % for all 3 distances in the experiment

Number of the sensor	12[mm]	20[mm]	40[mm]
1	1.423	6.011	16.073
2	1.427	3.351	12.851
3	1.338	5.071	11.854
4	1.045	2.508	14.517
5	0.796	4.772	10.218
6	1.283	1.969	12.574
7	1.565	9.423	12.582
8	1.489	7.282	12.524

In only 76 iterations in 15 seconds, the ANN is trained with the root mean square error $3.76 \cdot 10^{-12}$ for training set and 0.2181 for test set.

In RBF network training, it was required for the error to be less than 10^{-3} and that was accomplished. For each sensor there have been 10 ANNs trained and the one with best results was chosen. Each network had 20 neurons in hidden layer and 240 input and 12 output parameters.

Results in table 2 shows that the error varies from 0.78% to 1.56 when measuring distance to the obstacle is 12 [mm]. When measuring the distances of 20 [mm] to the obstacle, the error varies between 1.97% and 9.42%. Error of 40 [mm] varies between 10.22% and 16.07%. Error already exceeds 10% and the limitation of the sensors comes into play.

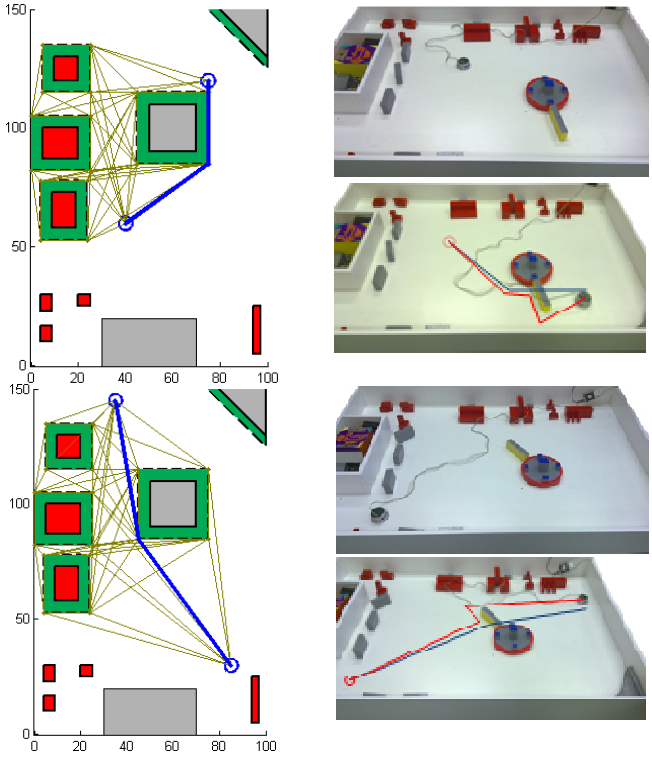

Figure 8. Experimental results for two cases

For this reason, two ANNs were trained for obstacle avoidance, one for the 20 [mm] distance and another one for 40 [mm] distance. Graphical demonstration of BP network testing, for 10 typical cases, is presented in Fig. 9 X axis represents the ten typical cases, while the Y axis represents the output of the network (rotation angle) for a given case.

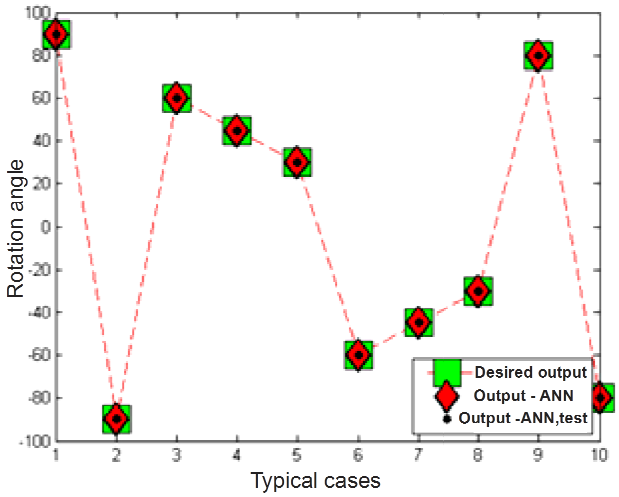

Figure 9. Graphical representation of training BP network to identify 10 typical obstacles

VI. Conclusion

The primary objective of this paper is to show the possibilities and to prove the usability of intelligent mobile robots in modern technological environments. The aim was to prove that the intelligent mobile robot, which has a pre-defined path and order of handling the machine, is able to avoid any obstacles in the environment and continue to perform its task.

In order for this to be achieved, it was necessary to develop and implement a model of motion. With this we answered to one of the major issues that intelligent mobile robot "needs" to know and that is "Where am I". Also, it is essential that the robot has the answers to the questions "Where I was", "Where is my goal" and "How to get to the goal". The answers to these questions provide algorithms for path planning, visibility graph and its search for the optimal path, while system robustness is achieved with the use of artificial neural networks.

After the performed experiments, where the robot had to get to a certain position, including avoiding of unknown obstacles, the X-axis error is between 5.5-6.5 [cm], while the Y-axis is between 0.5-1 [cm]. Given the distance traveled in terms of percentages, the X-axis error is 13-15%, while for Y only 1%. Larger error for the X-axis is partly a result of the slow communication between the computer and the robot, and it can be reduced by optimizing the code.

Further improvement can be done by introducing additional sensors, i.e. camera, which would be used for the detection of characteristic objects in the environment and thus increase the likelihood of successful performance of a task.

Acknowledgment

This work was funded by the Ministry of Education and Science of the Republic of Serbia under contracts TR-35003 III-44008 and III-44004.

References

[1] Sigwert, R., Nourbakhsh, R., Illah, Introduction to Autonomous Mobile Robots, Massachusetts Institute of Technology, 2004.

[2] Lozano-Perez, T., Wesley, M., A., An Algorithm for Planning CollisionFree Paths Among Polyhedral Obstacles, Communications of the ACM, Vol. 22, No. 10, October 1979.

[3] Velagić, J., Uvod u Mobilnu Robotiku, Elektrotehnički fakultet, Sarajevo,2009.

[4] Subir, K., Ghosh, Visibility-based Robot Path Planning, Tata Institute,Mumbai, India, 2009.

[5] Reinhard, D., Graph Theory, SpringerVerlag New York, 2000.

[6] Zhan, F., B., Noon, C., E., Shortest Path Algorithms: An Evaluation Using Real Road Networks, February, 1998.

[7] Miljković, Z., Sistemi Veštačkih Neuronskih Mreža u Proizvodnim Tehnologijama, Naučna Monografija, Beograd, 2003, VI+185 str.

[8] Miljković, Z., Aleksendrić D., Veštačke Neuronske Mreže Sa Izvodima Iz Teorije, Univerzitet u Beogradu – Mašinski Fakultet, 2009.

[9] Park, J., Sandberg, J. W. Universal Approximation Using Radial BasisFunctions Network, Neural Computation, vol. 3, pp. 246-257, 1991.

[10] Poggio, T., Girosi, F., Networks For Approximation And Learning, Proc. IEEE, vol. 78,no. 9, pp. 1481-1497, 1990