These data is possible to connect with input neurons.

The output data of the system (y^s_k) are suggestions of operating commands to the production equipments (e.g. open/close a gate, switch on/off a supercharger):

1. Suggestions to switch on/off a supercharger.
2. Suggestions to open/close gates to airway, circulating silt and superchargers.
3. Suggestions of the percent, gates to circulating silt to be opened.

The system output data are considered as output neurons.

The signals will be flowed in real time to inputs of the neural network in the processed form from the automated control system of technology processes (ACS TP) of biological refinement in aerotanks and the signals from the output layer flow back in ACS TP. The developed decision-making system will be directly involved in the given ACS TP and will work as its component under the remote equipment control mode. Also the given system will be possibly used as an independent program.

The back-propagation algorithm has been chosen as a training method for the neural network [2]. It is the most appropriate method because there is enough big database containing a lot of information about biological refinement processes gathered for many years of sewage disposal plants work. After minimization of the mean-square error of the network and getting of rather correct weights the network is good for work in real environment.

The neural network for the decision-making system will be realized and tested with MATLAB by means of the Neural Network Toolbox specially intended for neural network training and simulation.

The given decision-making system will have not only theoretical, but also practical importance. In future the system is planed to apply to Minsk aeration station No. 1.

The special program is developed in Scada-system Trace Mode for demonstration of decision-making system operation for biological refinement in aerotanks. The given program allows visually to present results of simulated system operation on a set of initial data, and also to trace changes in biological refinement after execution of control commands. It allows to estimate system performance on pseudo-real conditions.

Economic profit on this decision-making system setup can be gained not at once, but it can pay back even during one, two years that is a definite advantage of this setup.

References

1. Bondarev A.V. Manual. Lectures on sewage disposal plants: efficiency estimation of separate cleaning units. Technology calculations. Minsk, 2004.
2. Golovko V.A. Neurointellect: theory and applications. Book 1. Organization and training of feedforward and feedback neural networks. BPI, Brest, Belarus, 1999.
3. http://www.tvp.ru/vnizd/mathem1.htm.

ROBOT’S MOTION CONTROL ALGORITHMS ON A TASK OF LINE-FOLLOWING

Dzmitry Kaliukhovich
Brest State Technical University, Brest

Keywords: task of line-following, algorithms “Avoidance of the line”, “Hold the aim in the center”, “The best approximation with an arc”.

At the System Engineering Laboratory, Hochschule Ravensburg-Weingarten, Germany the mobile robot “MAX” was developed and realized [1] (see Fig. 1). There is a rich set of control commands for it. It has an incorporated web-camera to observe space in front of. So it was a natural step to improve it to follow under the defined route – it’s so called line-following task.

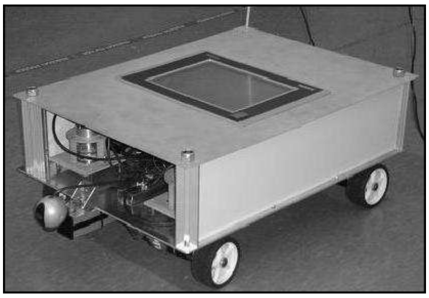

Fig. 1. Mobile robot “MAX”

To solve this task three empirical algorithms were developped: “Avoidance of the line”, “Hold the aim in the center”, “The best approximation with an arc” [2]. Before shortly to describe algorithms it’s important to notice that the line is represented as a sequence of “median” points lying on it in the order of y-coordinate increase. To prevent loosing the line tracing of commands is used. If the robot loose the line he takes the opposite motion direction to previous one.

The situation of line absence is occurred when the number of the detected “median” points is less than defined one by the user of the program.

The “Avoidance of the line” algorithm consists in permanent detection of line existence and absence and making a choice to move in direction opposite to previous one. It’s similar to the preventing of loosing the line technique but after finding one the robot has to loose it again (see Fig. 2). Actually in this algorithm robot follows under one border of the line. It works well at low speeds and gets bad at high ones.

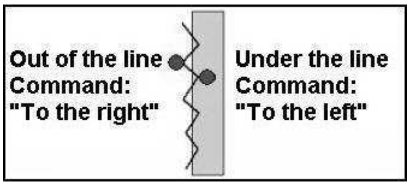

Fig. 2. “Avoidance of the line” algorithm

The “Hold the aim in the center” algorithm uses definitions of “Aim” and “Central window”. The “Central window” is the central part of a shot from the web-camera with the defined dimensions by the user. The “Aim” is a point in the shot with the “median” coordinates of all points lying in the “central window”. The algorithm works according to:

1. Find all points lying in the “central window”.
2. Calculate the coordinates of the “aim”.
3. If the “aim” point lies on the right side out of the “central window” so move to the right.
4. If the “aim” point lies on the left side out of the “central window” so move to the left.
5. If the “aim” point lies in the “central window” so move forward (see Fig. 3).

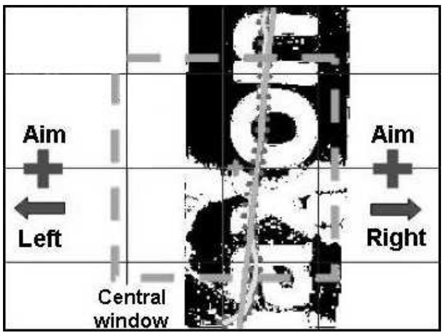

Fig. 3. “Hold the aim in the center” algorithm

The “Best approximation with an arc” is more complicated than previous ones. It’s based on reflection from the approximate turning radius of the line into one for the robot. While the robot can move to the right and to the left the best arc has the center either in the *(0; 0)* or in the *(w; 0)*, where w is a shot width.

$$j = 0 ... w/2, \tag{1}$$

$$R = j, \text{ if } \sum_{i=1}^{N} (\sqrt{x_i^2 + y_i^2} - j)^2 = \min_j, \tag{2}$$

where R is the radius of the best approximating arc.

The turning radius for the robot is calculated according to:

$$R_{ROBOT} = a + \frac{2 \cdot R}{w} \cdot b \quad (3)$$

where *a* and *b* are the algorithms parameters which set by the user.

Because of the robot's construction has constraints on the turning radius by default *a = 50* and *b = 100*. These parameter values corresponds to the robot's minimal and maximum turning radius while *R = 0* and *R = w/2* respectively (see Fig. 4).

This algorithm provides more high-quality and precise motion control in comparison with the previous ones thank to taking into consideration geometric parameters of the line to be followed.

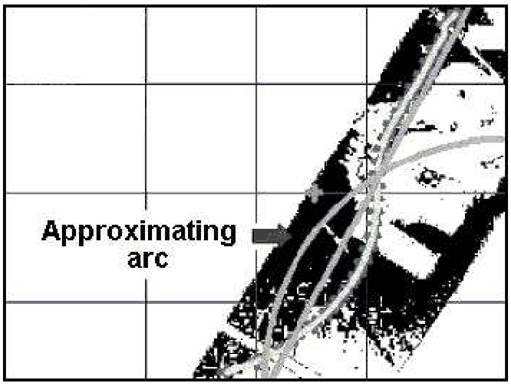

Fig. 4. "The best approximation with an arc" algorithm

The algorithms described above were implemented with C++ Builder v. 5. For tests the special route was created. It pre-sented the closed one-color (red) line made with an insulation tape on the one-color (brown) background. It consisted of different radius turnings and straight lines (see Fig. 5).

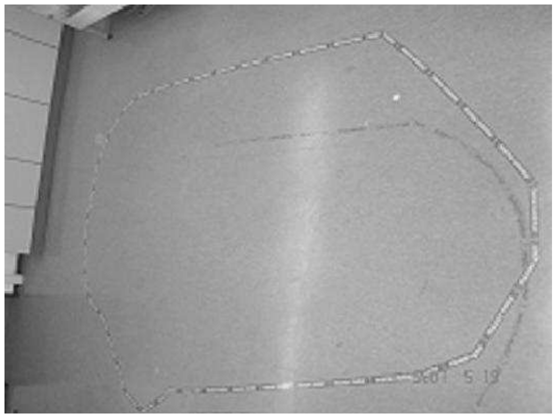

Fig. 5. Test route

The control program run under MS Windows XP, Pentium 4 2GHz and RAM 128 Mb. The speed was 4 cm/s (41% of the possible speed the robot is under control). The other parameters were by default. The tests showed that the motion was robust and rather precise. When the robot lost the line thanks to tracing of last commands it returned to the line. It should be noticed that the robot's motion was like "sinusoid" because of the small number of processing shots (approx. 2.3 per second) from the incorporated web-camera per second.

This work has done in cooperation of the Intelligent Information Technologies Department, Brest State Technical University, Belarus and the System Engineering Laboratory, Hochschule Ravensburg-Weingarten, Germany under common supervision of Prof. Dr. Vladimir Golovko and Prof. Dr. Andreas Paczynski respectively and was supported by the Landesstiftung Baden-Württemberg gGmbH.

References

1. Zajac M., Paczynski A., Stetter R., Ucinski D. Sensor/Actuator network concept for robust control of a four-wheel-steering mobile robot. Paper. Hochschule Ravensburg-Weingarten, Weingarten, Germany. 2006.

2. Dzmitry Kaliukhovich. Design and comparative analysis of robot's motion control algorithms on a task of line-following. Diploma paper. Brest State Technical University, Brest, Belarus. 2007.