ANNALS OF FACULTY ENGINEERING HUNEDOARA
- INTERNATIONAL JOURNAL OF ENGINEERING
Tome IX (Year 2011). Fascicule 3. (ISSN 1584 - 2673)

[1.] Diana TSANKOVA

NEURAL NETWORKS BASED PATH PLANNING

[1.] TECHNICAL UNIVERSITY SOFIA - BRANCH PLOVDIV, BULGARIA

ABSTRACT: The paper proposes a path-planning approach that generates a global optimal path according to a criterion, which has to guarantee „minimum path length and no collisions with obstacles". The optimization procedures reported in the literature use gradient descent techniques and suffer the local minima problem. The main contribution of the paper consists in overcoming this drawback by neural network based filling the local minimum and its vicinity. The good performance of the proposed modified algorithm is confirmed in MATLAB simulations.
KEYWORDS: path planning, neural network, gradient-based search, global and local optima

INTRODUCTION

Most of the global path-planning methods, which assume a static and completely known to the robot environment, are considered to belong to or to be variations of a few basic approaches: roadmap, cell decomposition, and potential field methods [2, 1]. Each of the path-planning methods has its own advantages and drawbacks and is more or less appropriate for use in a particular situation. Sometimes, during path following a local re-planning is needed in order to make the robot avoid collisions with unknown obstacles. By combining global and local path-planning methods mobile robots are able to operate in dynamic environments [7]. How to select the most suitable approach and how to use local path-planning in collaboration with the global one - these are still open research problems.

Some of the path-planning algorithms make use of neural networks (NNs) as a powerful tool of artificial intelligence (AI). In Ref. [3] the authors have proposed a path planner that generates a global optimal path according to a criterion, which guarantees „minimum path length and no collisions with obstacles". An appropriate gradient based method is used for minimization of the path length together with the penalties for collisions with a priori known obstacles. The penalties are produced using an approximation of obstacle-oriented repulsive potential function, made by a feed-forward neural network, trained by error back-propagation algorithm. However, due to the gradient search technique used for path length minimization, this method suffers from getting stuck in the local minima of the optimization function. The same path planner, which produces the global collision-free path between the given start and goal positions, has been used in Refs. [4,5,6] as part of a three-component system for navigation and control of mobile robots. The basic drawback mentioned above remains and decreases the reliability of the whole system.

The purpose of the paper is to propose a new approach for global path planning in 2D environment. The approach is a modification of the one, presented by Meng & Picton [3], and intends to overcome the problem with the gradient search getting stuck in the local minimum. The local minimum and its surrounding area are iteratively filled by introducing virtual obstacles into it, and the obstacles are approximated by a neural network. Criteria for starting and ceasing the iterative filling of the local minimum have been suggested. The validation of the proposed method has been carried out by means of MATLAB simulations.

THE PATH PLANNING

THE BASIC PATH PLANNING METHOD. The collision-free path planning can be stated as: Given an object with a start position, a desired goal position, and a set of obstacles, the problem is to find a continuous path from the start position to the goal position, which avoids colliding with obstacles along it. The path-planning procedure proposed in this work is based on the theoretical work of Meng and Picton [3]. The path for the object is represented by a set of N via points. The path finding algorithm is equivalent to optimizing a cost function, defined in terms of the total path length and the collision penalty, by moving the via points in the direction that minimizes the cost function. A two-layer log-sigmoid/log-sigmoid back-propagation neural network is used to produce the collision penalty. The surrounding area is divided into 2D grid cells and a binary value is assigned to each grid cell to indicate an obstacle presence: "0" means that the cell is fully unoccupied, and "1" - the cell is occupied. To take into account the robot's geometry the obstacles could be modified by growing their size isotropically by the robot's radius plus a small tolerance.

The x, y coordinates of the cells' centres and the corresponding assigned binary values are used as learning patterns for the 2-input/1-output neural network. The output of the network represents the collision penalty for the current position (x, y) of the object. The collision penalty of a path is defined as the sum of the individual collision penalties of all the via points. The energy function for the collision is defined as:

$$E_C = \sum_{i=1}^{N} C_i \ , \tag{1}$$

where C_i is the collision penalty for the i^{th} via point. The energy function for the path length is defined as the sum of the squares of all the segments' lengths connecting the via points:

$$E_L = \sum_{i=1}^{N} L_i^{\,2} = \sum_{i=1}^{N} [(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2] \, . \tag{2}$$

The total energy is

$$E = E_C + E_L \, , \tag{3}$$

The equation (3) is modified by multiplying both energy functions with positive weight coefficients k_C and k_L to express our preference for the collision criterion or for the path length criterion, respectively [4]:

$$E = k_C E_C + k_L E_L \, . \tag{4}$$

The dynamical equation for a via point is chosen to make time derivative of the energy be negative along the trajectory, because the low energy implies less collisions and a shorter path. The time derivative of E is

$$\frac{dE}{dt} = \frac{d}{dt}(k_C E_C + k_L E_L) = \sum_{i=1}^{N} (k_L \frac{\partial L_i^2}{\partial x_i} + k_C \frac{\partial C_i}{\partial x_i}) \frac{dx_i}{dt} + \sum_{i=1}^{N} (k_L \frac{\partial L_i^2}{\partial y_i} + k_C \frac{\partial C_i}{\partial y_i}) \frac{dy_i}{dt} \, . \tag{5}$$

Let

$$\begin{aligned} \frac{dx_i}{dt} &= -(k_L \frac{\partial L_i^2}{\partial x_i} + k_C \frac{\partial C_i}{\partial x_i}) \\ \frac{dy_i}{dt} &= -(k_L \frac{\partial L_i^2}{\partial y_i} + k_C \frac{\partial C_i}{\partial y_i}) \end{aligned} \, , \tag{6}$$

then

$$\frac{dE}{dt} = -\sum_{i=1}^{N} [(\frac{dx_i}{dt})^2 + (\frac{dy_i}{dt})^2] < 0 \, . \tag{7}$$

$dE/dt = 0$ if and only if $dx_i/dt = 0$ and $dy_i/dt = 0$. Therefore, all via points move along trajectories, decreasing the energy, and finally reach equilibrium positions. In (6)

$$\begin{aligned} \frac{\partial C_i}{\partial x_i} &= f'(I2) \sum_{j=1}^{S} W2_j f'(I1_j) W1^x{}_j \\ \frac{\partial C_i}{\partial y_i} &= f'(I2) \sum_{j=1}^{S} W2_j f'(I1_j) W1^y{}_j \end{aligned} \, , \tag{8}$$

where C_i is the output of the network, $I2$ is the input of the output layer neuron, $I1_j$ is the input of the j^{th} hidden layer neuron, S is the number of hidden layer neurons, $W1^x{}_j$ is the weight coefficient of the input "x" with respect to j^{th} hidden layer neuron, $W2_j$ is the weight coefficient of j^{th} hidden layer neuron's output with respect to the output layer neuron. From (6) and (8), the dynamical equations for x_i and y_i are derived as:

$$\begin{aligned} \frac{dx_i}{dt} &= -[k_L(2x_i - x_{i-1} - x_{i+1}) + k_C f'(I2) \sum_{j=1}^{S} W2_j f'(I1_j) W1^x{}_j] \\ \frac{dy_i}{dt} &= -[k_L(2y_i - y_{i-1} - y_{i+1}) + k_C f'(I2) \sum_{j=1}^{S} W2_j f'(I1_j) W1^y{}_j]. \end{aligned} \tag{9}$$

The suggested modification. When the gradient search (9) sticks in a local minimum, the resultant path comprises points, lying on an obstacle, or the segment, connecting two calculated points intersects an obstacle. The optimal path, except for going through obstacles, also (in accordance with (2)) consists of equal in size segments. Consequently, as a criterion for starting the iterative procedure, modifying the basic path-planning method, the fulfillment of at least one of the following two conditions could serve:

1. The output of the neural network generates at least once a penalty for an obstacle, when at the input the coordinates of the path points calculated by the formulae (9) of the basic algorithm are given.

2. At least two path segments exist, whose ratio is a number, bigger than a predetermined value (bigger than 1), and the neural network generates a penalty for an obstacle if the coordinates of the medium point of the bigger segment are introduced at the input.

The iterative procedure stops either after both of the mentioned above conditions are no longer met, or in case a predetermined number of iterations has been realized. After detecting a local minimum, the search procedure is modified in the following way. With the help of the above two conditions the point is defined, after which the calculated path (a path point or a path segment) intersects an obstacle. In this point a virtual obstacle is generated by means of a second neural network, having the same structure as the first one, but with continuously adjustable weights, so that the network approximates only the virtual obstacle (obstacles). Formula (4) obtains the following form:

$$E = k_V E_V + k_C E_C + k_L E_L \ , \qquad \left(E_V = \sum_{i=1}^{N} C_{V,i} \right) \ , \tag{10}$$

where E_V is the energy function for collision $C_{V,i}$ with the virtual obstacles, and k_V is its weight coefficient. The dynamical equations (9) are modified as:

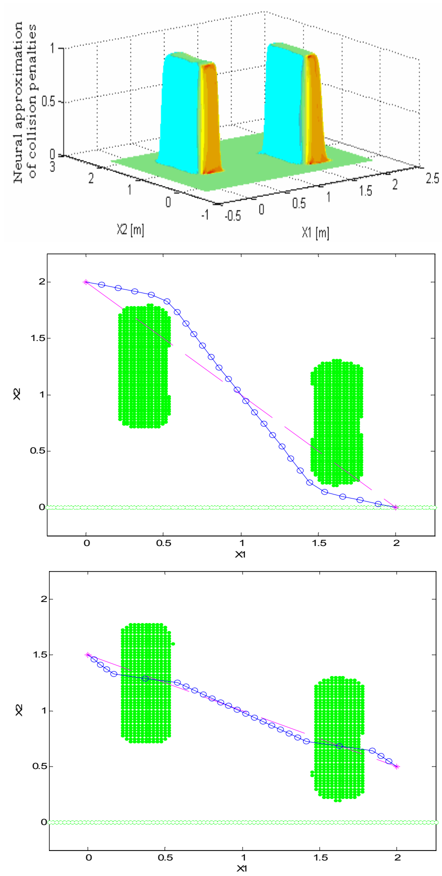
$$\begin{aligned} \frac{dx_i}{dt} &= -[k_L(2x_i - x_{i-1} - x_{i+1}) + k_C f'(I2)\sum_{j=1}^{S} W2_j f'(I1_j) W1^x{}_j + k_V f'(I2)\sum_{j=1}^{S} W2_{V,j} f'(I1_j) W1^x{}_{V,j}] \\ \frac{dy_i}{dt} &= -[k_L(2y_i - y_{i-1} - y_{i+1}) + k_C f'(I2)\sum_{j=1}^{S} W2_j f'(I1_j) W1^y{}_j + k_V f'(I2)\sum_{j=1}^{S} W2_{V,j} f'(I1_j) W1^y{}_{V,j}]. \end{aligned} \tag{11}$$

where the coefficients $W1^x{}_{V,j}$, $W1^y{}_{V,j}$, and $W2_{V,j}$ correspond to the second neural network, which is trained to approximate the virtual obstacles every time a new virtual obstacle is introduced. The virtual obstacle is assumed to have a round shape with a predetermined radius. Usually after introducing a few obstacles, the local minimum fills up and thus the probability for successful planning a path, free of collisions with obstacles, increases.

❖ RESULTS AND DISCUSSIONS

The workspace consisted of two a priori known obstacles. The neural network of the path planner was trained with the error back-propagation algorithm with momentum and adaptive learning rate. The number of neurons in the hidden layer was 25. The workspace was divided into 40×40 grid cells and the neural network used 1600 training samples. After training the neural network was used to produce collision penalties needed to solve the equations (9) and thus to find a path described by a set of $N = 25$ via points (without start and goal points). The virtual sampling time was $T_0 = 0.01\,\mathrm{s}$. Two experiments were carried out for checking the workability of the described algorithm for path planning: during the first one the the start position was $(0, 2)$, and the goal position was $(2, 0)$; and during the second one they were $(0, 1.5)$ and $(2, 0.5)$, correspondingly.

The initial path, needed for solution of the equations (9) was chosen as a straight line from the start position to the goal position (it is shown in Figure 1b, c as a dashed line). The weight coefficients in (9) were chosen to be $k_L = 1$ and $k_C = 1$. Figure 1 shows the approximation performance of the neural network ((a) the surface; (b,c) the dotted area). In the same figure, case (b), the piece-wise liner path obtained by the described above algorithm is presented by the bold line with 'o'-marks, and the case (c) shows unsuccessful path-planning result due to the local minimum problem. All distances in the figures in this article are presented in metres.



(a)
(b)
(c)

Figure 1. Results from the path-planning procedure: (a) Surface of collision penalties generated by a neural network, trained to recognize a priori known obstacles; (b) Collision-free path obtained by the path-planning algorithm; (c) Unsuccessful path-planning procedure.

The start point $(0, 1.5)$ and the goal point $(2, 0.5)$, corresponding to the unsuccessful experiment, were used again for path planning but with the help of the modified algorithm (11) this time. Weight coefficient $k_V = 1$ was chosen. While at the beginning of the experiment the basic neural network was trained a single time to approximate the predetermined static obstacles, the additional neural network is now trained permanently and alternatively with solving (11), so that it could approximate the round virtual obstacles, which are introduced to fill the local minimum. The radius of the virtual obstacles was chosen to be $r = 0.08\,\mathrm{m}$. The obtained results are presented in Figure 2, where the cases (a), (b), (c) and (d) correspond to the first, second, fifth and seventh iterations and (e) depicts the neural approximation of the real and virtual obstacles, obtained as a result of summing the outputs of both neural networks (the basic and the additionally introduced in (10) and (11)).

The area with virtual obstacles, iteratively introduced, is shown by a multitude of red points (Figure 2a,b,c,d). The good performance of the proposed modified algorithm for path planning (11) is confirmed by the considered example. The solution is obtained at the seventh iteration. During each of the iterations the additional neural network is trained to approximate the virtual obstacles, introduced up to the moment and by means of it the path is planned (11). The training of the neural network itself by the Back-propagation learning algorithm could be easier or more difficult (and longer) depending on the number and the location of the virtual obstacles. Besides, the Back-propagation learning, as a gradient method, suffers from the same problem of sticking in a local minimum in the surface of the error. These problems are solvable though, by using improvements such as 'momentum' etc. in the algorithm of the Back-propagation, as well as by developments in the calculating techniques. As a whole, with the use of the modified formulae (11) a free of collisions with obstacles path can be generated, but it cannot be guaranteed after the introduction of virtual obstacles that the resultant path will have a minimum length.

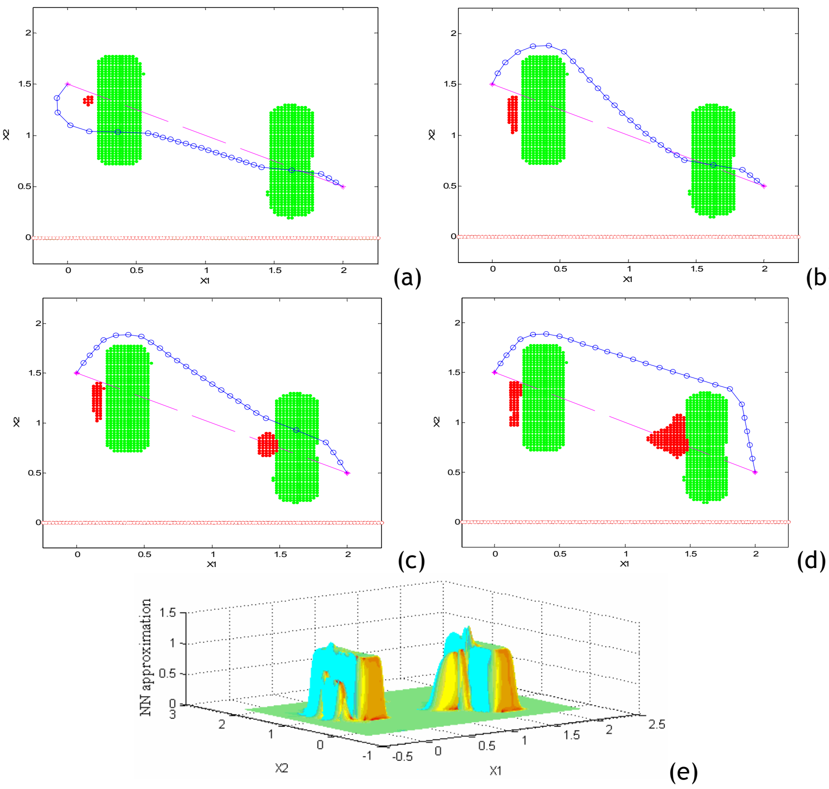


Figure 2. Results from the modified path-planning procedure: (a) I iteration; (b) II iteration; (c) V iteration; (d) VII iteration; and (e) the surface of the modified neural approximation in the VII iteration.

❖ CONCLUSIONS

The proposed modification of the path-planning method presented in [3] can overcome the local minima problem with high probability. The basic neural network in the path-planning algorithm can automatically build up the collision penalty function (as an obstacles' potential function approximation). The training procedure uses the information from the environment, no matter how many obstacles there are and how they look like, that makes the algorithm effective. By using an additional neural network, the local minima of the energy function can be filled and thus the path-planning procedure can ensure successful collision-free path. The proposed modified path-planning algorithm will be further verified on a real mobile robot.

❖ ACKNOWLEDGEMENTS

The paper presents research and development, supported by the Scientific Fund of Internal Competition of the Technical University-Sofia under the Research Projects No.102ni085-19 /2010 and No.112pd010-19 /2011.

❖ REFERENCES

[1.] Choset, H., K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, and S. Thrun, Principles of Robot Motion: Theory, Algorithms, and Implementations, MIT Press, Boston, 2005
[2.] Latombe, J.C., Robot Motion Planning, Kluwer Academic Publishers, Boston, 1991
[3.] Meng, H., and P.D. Picton, A neural network for collision-free path-planning. In: Artificial Neural Networks, Aleksander, I. & Taylor, J. (Ed.), vol.2, pp. 591-594, 1992
[4.] Topalov, A., D. Tsankova, M. Petrov, and T. Proychev, Intelligent motion planning and control for a simulated mobile robot, Proc. of the 2nd IFAC Workshop on New Trends in Design of Control Systems, pp. 338-343, Smolenice, Slovac Republic, 1997
[5.] Topalov, A., D. Tsankova, M. Petrov, and T. Proychev, Intelligent sensor-based navigation and control of mobile robot in a partially known environment, Proc. of the 3rd IFAC Symposium on Intelligent Autonomous Vehicles, pp. 439-444, Madrid, 1998
[6.] Tsankova, D., Neural Networks Based Navigation and Control of a Mobile Robot in a Partially Known Environment, Chapter 10, pp.197-222, In Mobile Robots Navigation, Alejandra Barrera (Ed.), I-tech Education and Publishing, Vienna, Austria, 2010
[7.] Yahja, A., S. Singh, and A. Stentz, An efficient on-line path planner for outdoor mobile robots. Robotics and Autonomous Systems, vol.32, pp. 129-143, Elsevier Science, 2000