A New Wave Neural Network Dynamics for Planning Safe Paths of Autonomous Objects in a Dynamically Changing World

DMITRY V. LEBEDEV, JOCHEN J. STEIL, HELGE RITTER
University of Bielefeld
AG Neuroinformatik, Faculty of Technology, University of Bielefeld
GERMANY

Abstract: - We consider the problem of finding a safe path for a robot/manipulator in a dynamic environment and propose a novel neural network model for solving this task. The network has discrete time-dynamics, is locally-connected, and is, hence, computationally efficient. No preliminary information about the current world status is required for the planning process. Path generation is performed via the neural-activity landscape, which forms a dynamically-updating potential field over a distributed representation of the configuration space of an object. The network dynamics guarantees local adaptations, and includes a set of strict rules for determining the next step in the path of an object. According to these rules, planned paths tend to be optimal in a L_1 metric. We present here the description of the model, and evaluate simulation results for various types of environmental changes.

Key-Words: - path planning, neural networks, wave expansion, robotics, autonomous navigation

1 Introduction

Capabilities of planning safe paths independently are crucial for autonomous robots/manipulators. Real-world environments, where such robots have to operate, are often unknown to them and may have a complex structure. Moreover, the status of the operational environment may change in time due to sudden appearance or/and disappearance of other objects.

In the last decade the problem of dynamical path planning has been in the research focus of many scientists, and a lot of papers, devoted to this non-trivial problem, have been published. Most of the existing approaches (e.g [1]) require full knowledge about the world.

Several neural network approaches have been proposed for solving the problem of path planning ([2]-[11]). The authors of [6] use a self-organizing Kohonen net with nodes of two types. The work [5] contains a description of a network with oscillating behavior, that solves the problem of path planning for an object with two degrees of freedom (DOFs), formulated as a dynamic programming task. An algorithm, proposed in [9], uses a set of intermediate points, connected by elastic strings. Gradient forces of the potential field, generated by a multi-layer neural network, minimize the length of the strings, forcing them at the time to round the obstacles. Authors of [2] used a multilayer feed-forward network to perform real-time path planning. A neural network for path finding, described in [4], has three layers of neurons with recurrent connections in the local neighborhoods. The dynamics of the network emulates the diffusion process.

All of the approaches, mentioned above, are applied, however, only to stationary environments. Besides that, the optimality of the path is often left out of consideration.

A biologically plausible neural network for dynamical trajectory generation, described in [10], has recently been improved in [11], but additional efforts are required for tuning the network parameters.

In this paper we present a novel neural network dynamics for finding a path in a dynamic world. The model is based in interweaving manner on three paradigms, (a) the notation of configuration space as a framework for a flexible object representation [12]; (b) a potential field building, which we state as a generic and elegant method for formation/reconstruction of a path; and, (c) a wave expansion mechanism, that guarantees an efficient construction of the potential field.

The model has been tested for various types of dynamical changes, and has demonstrated efficient and effective path generation capabilities.

The rest of the paper is organized as follows. In section 2 we give a formal problem definition. Section 3 contains the description of the proposed neural network model. We evaluate simulation results in section 4 and conclude with a discussion of results in section 5.

2 Problem Definition

Without loss of generality, we can define the configuration space $C \subset \Re^d$ to be a regularly discretized hypercube, where d is the number of DOFs of an object. For an object in C the starting and the final configurations S and T are denoted. Suppose at the time t_k, there is the number N_k of obstacles (i.e. of forbidden configurations) in C. At that moment of time, positions of all obstacles in C form the obstacle region $O_k = \{O_i^k\} = \{(X_{i_1}^k, ..., X_{i_d}^k)\}$, where the obstacle coordinates in C are denoted by vectors $(X_{i_1}^k, ..., X_{i_d}^k)$, $1 \le i \le N_k$. Let $\tau(t_k) = (p_1(t_k), .., p_d(t_k))$ define the configuration of the object in C at the time t_k. The task is to find a safe (i.e. a collision-free) path in C from the start to the goal, i.e. a path τ, that satisfies the conditions: $\tau(t_s) = S$, $\tau(t_g) = T$, $\tau(t_k) \cap O_k = \emptyset$, $t_s \le t_k \le t_g$, where t_s and t_g are the time of starting and the time of reaching the goal, respectively.

3 The Model Description

3.1 The General Idea

We realize in our approach the idea of generating a numerical potential field over a discretized representation of the configuration space. Therefore, the state space of the neural network is the discretized C of an object. We use a wave-expansion mechanism to form the desired potential field. The idea of neural wave expansion was proposed and applied for a stationary domain in [13]. According to this idea, the activity is spread around the source of excitation, and the minimum value of the generated potential field always stays at the excitory point, which, in turn, attracts an object. An example of a simple wave expansion in the plane is depicted in Fig. 1.

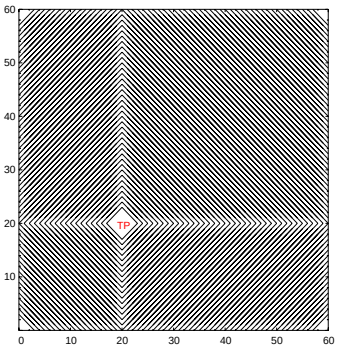

Fig. 1. Wave fronts expansion around a target point in a 2D environment.

For planning paths in a non-stationary domain, we propose a dynamics for the wave expansion, which makes an effective combination of (1) repetitive wave expansions, and (2) rules for indication of the object's next route step in a dynamically-updating potential field. These rules ensure that object moves only along a safe route. Such combination is achieved by using a set of simple functions, that define the network dynamics.

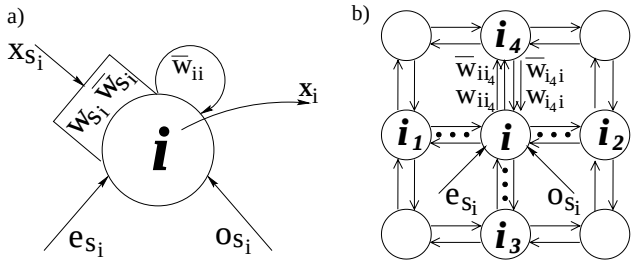

Fig. 2. a) The neuron model; b) Network neighborhood structure in the plane.

We feed a regular excitation source at the target, resulting in a new wave of neural activity in the network field at each time step. Neural activity, therefore, propagates first through the network field, and then changes locally, adapting to the dynamical status of the environment. Since in the case of a stationary environment, wave fronts yield paths, which are optimal in a L_1 metric, dynamical paths, which are generated by the proposed network, tend also to keep such optimality. Consequently, longer paths to the target are automatically cut out of consideration.

3.2 Neural Network Architecture

The proposed neural network has a parallel locally-connected structure of cellular type. Depending on the dimensionality of C the network may consist either of a single layer (for a 2D configuration space), or a set of layers with locally-connected neurons. The arrangement of the neurons coincides with the discretized structure of C, i.e. each discrete position in C is associated with a neuron in the network field. Figure 2a contains the neuron structure.

The i-th neuron is connected with $n = 2d$ immediate neighbors, where d is the dimensionality of C. We will denote a set of neighbors of the i-th neuron as s_i, i.e., $s_i = \{i_1, ..., i_n\}$, and the relative order of neurons in the local neighborhoods is fixed. A neuron neighborhood for an example 2D configuration space is depicted in Fig. 2b.

The network can be viewed as a discrete-time dynamical system, which can be fully described by a set of neuron state vectors $X_i = [x_i, \widehat{W}_{s_i}] \subset \Re^{2n+2}$. The first element x_i of vector X_i is the activity level, or the output of the i-th neuron, which is a real scalar quantity. The second element, vector $\widehat{W}_{s_i} = [w_{i_1 i}, ..., w_{i_n i}, \bar{w}_{ii}, \bar{w}_{i_1 i}, ..., \bar{w}_{i_n i}]$ consists of two sets

of connection weights, defining the synaptic strengths of the connections between neuron i and its immediate neighbors. Notice, that w_{ji}, $\bar{w}_{ji}$, w_{ij} and $\bar{w}_{ij}$ are *four* connection weights between neurons i and j, and weights w_{ji} and $\bar{w}_{ji}$ follow different learning rules.

Activity levels of the neurons in the neighborhood of the neuron i comprise vector $X_{s_i} = [x_{i_1}, x_{i_2}, ..., x_{i_n}]$. The neuron i is active, if $x_i > 0$, and inactive, otherwise.

Additionally, the activity of the i-th neuron can be influenced by excitory and inhibitory inputs of the neighborhood neurons, which form accordingly vectors $e_{s_i} = [e_{i_1}, ..., e_{i_n}, e_i]$ and $o_{s_i} = [o_{i_1}, ..., o_{i_n}, o_i]$, whose elements could be of value zero or one only.

3.3 Path Planning Process

In this section we present the underlying idea of the path planning process along with its formal description.

State equations (1)-(3) contains the rules, providing the activity evolution of the neuron i and associated with the latter connection weights.

$$x_i^{t+1} = e_i + (1 - e_i)\times \\ r\left(\sum_{j\in s_i} w_{ji}(x_j + 2 + e_j \cdot \bar{w}_{jj})\right). \quad (1)$$

$$w_{ji}^{t+1} = U\left(P_k(X_{s_i})\right), \quad j \in s_i, \quad (2)$$

$$\bar{w}_{ji}^{t+1} = e_j \cdot (\bar{w}_{ji}^t + 1) + (1 - e_i)\times \\ \bar{\delta}_{ji} \cdot (x_j + e_j \cdot \bar{w}_{jj}), \quad j \in \{\{s_i\} + i\}, \quad (3)$$

Functions $r(.)$, $U(.)$ and $M_k(.)$ are described as $r(\tau) = \begin{cases} [\tau], & \tau \geq 0 \\ 0, & \tau < 0 \end{cases}$, where $[\tau]$ denotes whole part of τ, $U(\tau) = \begin{cases} 1, & \tau > 0 \\ 0, & \tau \leq 0 \end{cases}$, and $\bar{\delta}_{ki} = \begin{cases} 1, & i \neq k \\ 0, & i = k \end{cases}$.

Initially, activity levels of all neurons and all connection weights are zero. Available information about the current status of the environment is applied to the external neuron inputs. So, the neurons, corresponding to the stationary or dynamical obstacles in C receive one on their inhibitory inputs o_i. The neuron associated with a target position in C receives one on its excitory input, and initiates, therefore, spreading of activity in the neural field. As can be seen from Eq. (1), the activity level of the target neuron is equal to one during the network evolution, that corresponds to the minimum value of the potential field. From the same formula, outputs of other neurons depend on the current state of the neurons in the local neighborhoods. Eqs. (2) guarantee the positive value of only the connection weight from the neighboring neuron, chosen by the priority function $P_k(.)$ (4).

$$P_k(x_{s_i}) = \begin{cases} \Omega_1, & k = 1 \\ \Omega_k + I(\sum_{l=1}^{k-1} \Omega_l), & 1 < k \leq n \end{cases}. \quad (4)$$

This function queries the neighboring neurons in accordance with a priority, assigned to them, and selects the only neuron, which satisfies to a positive value of the expression (5). The priority could be given, e.g., by assigning a numerical label. The neighboring neurons will be queried then according to the increase of this label.

$$\Omega_k = F(\gamma_k) \cdot U(x_k^t) \cdot E(x_k^t, \bar{w}_{ki}^t)\times \\ \left(U(x_i^t) \cdot D(x_i^t, x_k^t) + F(x_i^t)\right). \quad (5)$$

Here functions $I(.)$, $E(.)$, $D(.)$ and $F(.)$ are given by $I(\tau) = \begin{cases} -c, & c >> 0, \tau > 0 \\ 0, & \tau \leq 0 \end{cases}$ [13], where c is a big positive constant, $E(\tau_1, \tau_2) = \begin{cases} 1, & \tau_1 \neq \tau_2 \\ 0, & \tau_1 = \tau_2 \end{cases}$, $D(\tau_1, \tau_2) = \begin{cases} 1, & \tau_1 - \tau_2 \geq 0 \\ 0, & \tau_1 - \tau_2 < 0 \end{cases}$, and $F(\tau) = 1 - U(\tau)$.

As follows from the formula (5), the connection weight w_{ji} will receive the positive value of one if, and only if the following conditions are fulfilled: a) the neuron j has changed its activity level during the considered time step; this means, that the neuron j belongs to the shortest path; b) the neuron j is active; i.e., the rules a), b), c) and d) are fulfilled also for this neuron; c) it has zero on its inhibitory input; i.e., the neuron j does not correspond to an obstacle at the current time step; and, if the i-th neuron already has during the considered time step non-zero activation, d) the activity level of the neuron j is not greater than those of the i-th neuron; i.e., an object will move only along the shortest route. These rules ensure, therefore, that an object always moves along the safe and the shortest path. If one of these rules is false, the corresponding weight w_{ji} either receives zero value, or stays zero. Connection weights $\bar{w}_{ji}$ (Eqs. (3)) are responsible for storing of activity values of the neighboring neurons, calculated at the previous time step. For the neuron, corresponding to a target position, whose activity stays always at one, the necessary condition of alteration of the activity level is done via incrementing the associated weight $\bar{w}_{jj}$ by one. This fact guarantees, that the immediate neighbors of the target neuron will evolve properly.

Permanent excitation of the target neuron via its external input e_i leads to generation at each time step of a new wave of neural activity in the network field. These waves carry an updated information about the environment status. The dynamical activity landscape, therefore, accounts for the changes in the environment and adapts to them.

An object starts moving as the first wave of neural activity has reached its initial position. Due to the

strict rules, which are included into the network state equations and provide a proper potential field building, the next step rule for an object becomes rather simple: it should move into the direction of the positive weight from the neighboring neuron, or, formally, $\tau(t_s + n) = \{p_j : w_{ji} > 0, j \in s_i\}$, where p_j is the next position of an object in C, associated with the neuron j, t_s is the starting time, and $n > 0$ denotes the n-th discrete time step.

4 Simulation Results

In this section we evaluate simulation results for various types of dynamical changes in the workspace. We demonstrate our experiments for a point object with 2 DOFs, that doesn't restrict general applicability of the model.

We show stationary obstacles in the workspace in black, and denote dynamical obstacles in a light-gray color. Paths of the object are represented by continuous curves. Empty squares denote randomly-appearing obstacles. SP and TP stand for Start and Target Position.

We used for our experiments a network field, consisting of 61×61 neurons over the discretized workspace representation. The borders of the workspace were also treated as obstacles. For test simulations we chose the neighboring neurons labelled as shown in Fig. 2a. This doesn't lead to a loss of the model functionality. Therefore, when possible, the object prefers first to move horizontally.

4.1 The "open gate" situation

This model situation is shown in Fig. 3-4. The workspace is cluttered with static obstacles. After 50 path steps of the object, dynamic obstacles (at the right border of the workspace) start to move in the direction, shown by the arrow, as depicted in Fig. 3a. Figures 3b and 3c contain two further intermediate episodes during the traverse. The dynamic obstacles

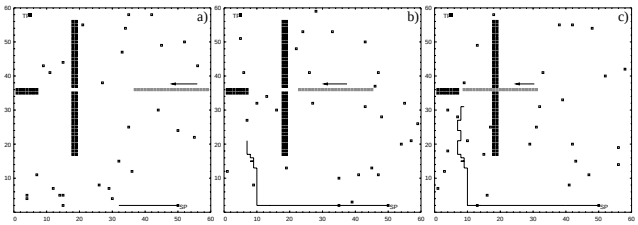

Fig. 3. Path planning in the dynamic environment: a),b),c) - three intermediate stages during the navigation.

stop then at the position, shown in Fig. 4a, leaving a small gate open. The object traverses through the gate, while avoiding 30 random obstacles, appearing in the

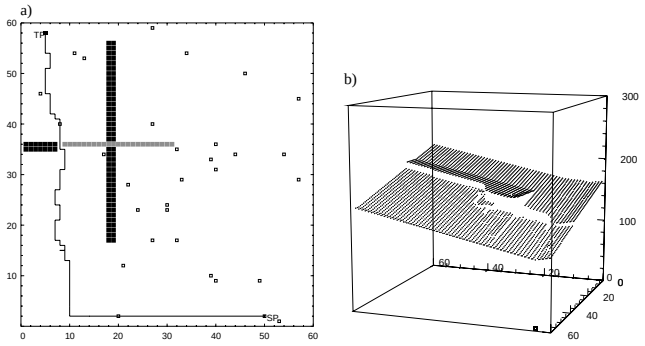

Fig. 4. The "open gate" situation: a) Moving obstacles do not influence the planned path; b) Activity landscape at the moment of reaching the target in Fig. 4a).

workspace at each time step. The resulting path of the object and the activity landscape, corresponding to the time of reaching the target are illustrated in Fig. 4a and 4b, accordingly.

4.2 The "closed gate" situation

The initial setup for this test example is very similar to the "open gate" situation, and the object goes through the same intermediate stages, as shown in Fig. 3. But in the new situation, the moving obstacles stop at the position, as illustrated in Fig. 5a, and close the passage before the object has come through the gate. The object must then dynamically react to environmental changes, and it found another route. The final path and the activity landscape at the moment of reaching the target are depicted in Fig. 5a and 5b, respectively.

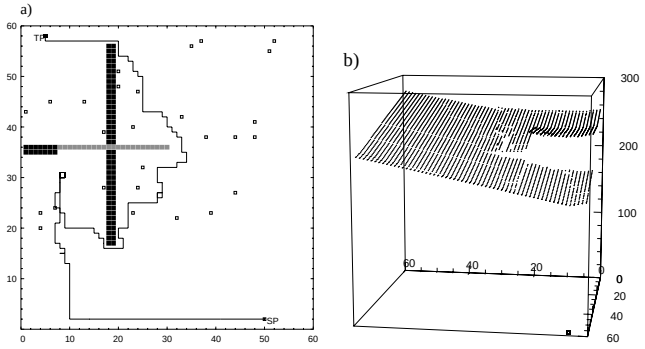

Fig. 5. The "closed gate" situation: a) The object dynamically follows another route; b) Activity landscape at the moment of arriving to the goal in Fig. 5a).

4.3 Disappearing dynamic obstacles

In this model situation the dynamic obstacles began the simulation as depicted in Fig. 6a, and started to move in the direction of the arrows. The object began to traverse with account of dynamical changes, and as it went 30 path steps, the obstacles froze at the initial positions. The activity landscape then adapted quickly

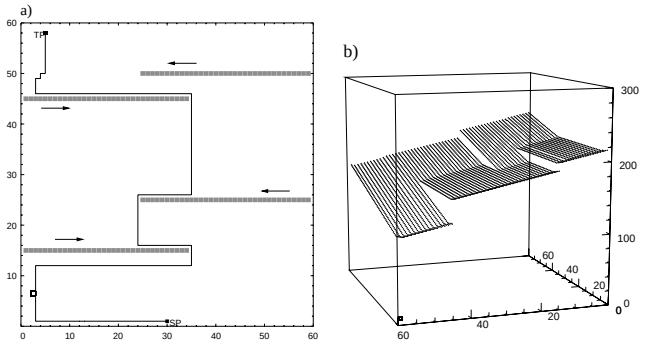

Fig. 6. Disappearing dynamic obstacles: a) Finding a route after the obstacles stopped moving; b) Activity landscape at the moment of reaching the target in Fig. 6a).

to the status of the world. The resulting activity landscape in Fig. 6b reflects the potential field structure for the static workspace at the moment of arrival to the goal. The path of the object is depicted in Fig. 6a.

4.4 Emerging dynamic obstacles

In this example situation the start and the target positions for the object are as in the previous example. The obstacles appeared first as shown in Fig. 6a. The object started moving, and as it made 30 path steps, the obstacles began drifting in the directions of the arrows. The object dynamically adapted to this complex situation and successfully approached to the goal. The resulting route and the activity landscape at the arrival to the target are illustrated in Fig. 7a and 7b, accordingly.

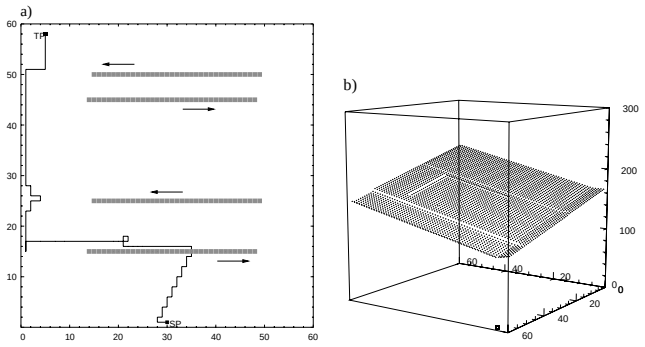

Fig. 7. Emerging dynamic obstacles: a) Planning a path while adapting to sudden changes in the environment; b) Activity landscape at the moment of arrival to the target in Fig. 7a).

4.5 Workspace occupation by random obstacles

We performed with our model a series of experiments with randomly-appearing obstacles. Some examples for 10, 150 and 250 obstacles, emerging randomly at each time step, are illustrated in Fig. 8. These model examples clearly demonstrate the tendency of the path to be optimal in L_1 metric. The path with 10 obstacles in Fig. 8a is near-optimal (see Table 1 for results of another experiments).

The number of obstacles	Path length
empty workspace	101(the optimal length)
10 obstacles	107
50 obstacles	121
100 obstacles	133
150 obstacles	119
200 obstacles	139
250 obstacles	209

Table 1. Path lengths in L_1 metric in the experiments with random obstacles, appearing at each time step in the workspace. The start and the target are as in Fig. 8.

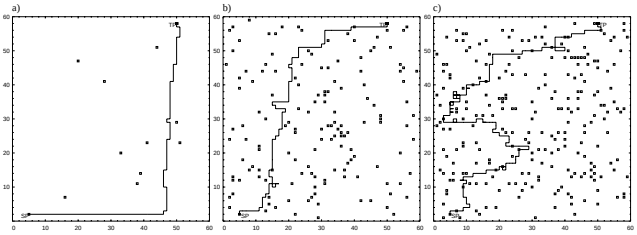

Fig. 8. Planning paths avoiding randomly-appearing obstacles in the workspace: a) With 10 obstacles; b) With 150 obstacles; c) With 250 obstacles.

4.6 Target pursuit

Our network is also capable to dynamically track a moving target. We show two examples here, for the empty workspace, and for the workspace with 50 random obstacles, emerging in the workspace at each time step. In both situations, depicted in Fig. 9a and 9b, the target started to move after 10 path steps of the object. The path of the target is shown by empty circles. To reach the target, the object must choose the optimal route. In both examples the object caught the moving target and demonstrated a reliable pursuance.

5 Conclusions

We presented in this paper a novel neural dynamics to solve the problem of finding a path in a changing environment. The proposed dynamics in efficient manner combines the wave expansion mechanism with a set of rules for detection of the next candidate path step. Due to local interactions between neurons and a regular excitation at the target neuron, the neural activity landscape adapts and accounts for environmental changes, providing proper formation of the potential

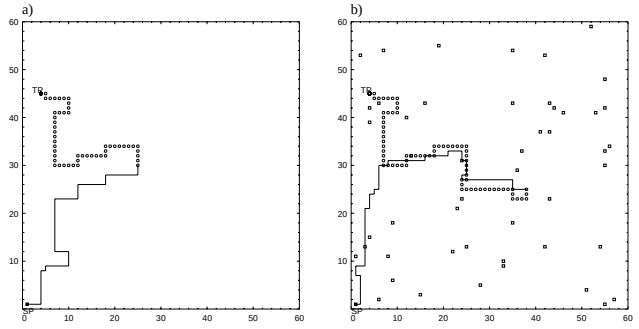

Fig. 9. Target persuasion: a) In an empty workspace; b) With 50 randomly-appearing obstacles.

field. The target point stays always at the minimum value of the potential field and attracts an object to the goal.

We focus in the following list on the main properties of the proposed model:

▷ no *a priori* knowledge of the environment status is needed;

▷ no learning process is required;

▷ network is locally-connected;

▷ computational complexity grows linearly in the number of neurons in the field;

▷ tendency to gain optimal paths in L_1 metric;

▷ due to fast activity propagation real-time planning is possible;

The proposed dynamics has been tested on various types of complex dynamical changes, including appearance and disappearance of obstacles, avoidance of random obstacles occupying the workspace and tracking a moving target. It has shown both the capabilities of fast adaptation to the dynamical changes and a fast activity stabilization in the absence of the latter. The planned paths are safe and have the tendency to the optimality in a L_1 metric.

Due to dynamically-updating potential field an object navigates actively, without waiting until the environment presents ”good-traversal opportunities”. Hence, one can consider the proposed approach as a compromise between tendency to path optimality and active reaction on environmental changes.

The described approach could be applied for planning paths of both mobile autonomous systems and robotic manipulators.

6 Acknowledgments

The work of the first author has been done with support from the DFG (German Research Society), the grant GRK256-2. The first author would like to thank also Patrick McGuire and Robert Haschke for their helping.

References:

[1] B. Baginski, The Z^3-Method For Fast Path Planning in Dynamic Environments, *Proc. of IASTED Conf. Applications of Control and Robotics*, 1996, pp. 47-52.

[2] J. Park, S. Lee, Neural Computation For Collision-free Path Planning, *Proc. IEEE Conf. on Neural Networks*, Vol. 2, 1990, pp. 229-232.

[3] Ashraf A. Kassim, B.V.K. Vijaya Kumar, Path Planning for Autonomous Robots Using Neural Networks, *Journal of Intelligent Systems*, Vol. 7(1-2), 1997, pp. 33-56.

[4] Th. Kindermann, H. Cruse, K. Dautenhahn, A fast, three-layer neural network for path finding, *Network: Computation in Neural Systems*, Vol. 7, 1996, pp. 423-436.

[5] M. Lemmon, 2-Degree-of-freedom Robot Path Planning using Cooperative Neural Fields, *Neural Computation*, Vol. 3, 1991, pp. 350-362.

[6] Jules M. Vleugels, Joost N. Kok, Mark H. Overmars, Motion Planning Using a Colored Kohonen Network, *Technical Report*, Department of Computer Science, Utrecht University, 1993, Report RUU-CS-93-38.

[7] G. Bugmann, J.G. Taylor, M. Denham, Route finding by neural nets, In: *Neural Networks* (ed. J.G. Taylor), Alfred Waller Ltd, 1995, pp. 217-230.

[8] L. Tarassenko, M. Brownlow, G. Marshall, J. Tombs, A. Murray, Real-time autonomous robot navigation using VLSI neural networks, In: *Advances in Neural Information Processing Systems 3*, 1991, pp. 422-428.

[9] S. Lee, G. Kardaras, Collision-Free Path Planning with Neural Networks, *Proc. Int. Conf. on Robotics and Automation*, 1997, pp. 3565-3570.

[10] R. Glasius, A. Komoda, S. Gielen, A biologically inspired neural net for trajectory formation and obstacle avoidance, *Biological Cybernetics*, Vol. 74(6), 1996, pp. 511-520.

[11] Simon X. Yang, Max Meng, An efficient neural network approach to dynamic robot motion planning, *Neural Networks*, Vol. 13, 2000, pp. 143-148.

[12] T. Lozano-Perez, Spacial planning: a configuration space approach, *IEEE Transactions on Computers*, 1983, pp. C-32:108-120.

[13] Ashraf A. Kassim, B.V.K. Vijaya Kumar, The wave expansion neural network, *Neurocomputing*, Vol. 16, 1997, pp. 237-258.