1-1-2006

Neural Network Approach for Obstacle Avoidance in 3-D Environments for UAVs

Vivek Yadav

Xiaohua Wang

S. N. Balakrishnan
Missouri University of Science and Technology, bala@mst.edu

Recommended Citation

V. Yadav et al., "Neural Network Approach for Obstacle Avoidance in 3-D Environments for UAVs," *Proceedings of the 2006 American Control Conference*, Institute of Electrical and Electronics Engineers (IEEE), Jan 2006.
The definitive version is available at https://doi.org/10.1109/ACC.2006.1657288

Neural Network Approach for Obstacle Avoidance in 3-D Environments for UAVs

Vivek Yadav , Xiaohua Wang, and S.N. Balakrishnan, *Member, IEEE*

***Abstract*—In this paper a controller design is proposed to get obstacle free trajectories in a three dimensional urban environment for Unmanned Air Vehicles (UAVs). The controller has a two-layer architecture. In the Upper layer, vision-inspired Grossberg Neural Network is proposed to get the shortest distance paths. In the bottom layer, a model predictive control (MPC) based controller is used to obtain dynamically feasible trajectories. Simulation results are presented for to demonstrate the potential of the approach.**

I. INTRODUCTION

WITH the increasing capabilities of the UAVs, several different applications are being identified [1] for UAVs. Apart from the applications in combat scenario, UAVs can also be used for providing services for example, public transportation and shipping cargos. For the UAVs to work as buses and trucks of future, the UAVs must be capable of moving in urban environments without flying into any of the buildings. Thus it becomes extremely important to have collision free path planning methods for the UAVs in complex urban environments.

Most of the research done earlier assumes that the UAVs fly at a constant altitude [2-6]. There are several schemes developed for obstacle avoidance in two dimensional (2D) [4-6]. These methods though give a collision free path but not an optimal way of solving the problem. The assumption that the UAVs fly at a constant altitude is reasonable in certain applications like searching an unknown terrain. However this assumption is not valid for path planning in 3-D environments. Due to assumption of constant altitude, the shortest distance paths may not be generated always; also it is not necessary that the destination of the UAV is at the same altitude.

In [7] a method for path planning in 3D was presented. Receding horizon control is used to get the paths in 3D environment. In path planning phase, the nodes (vertices) are selected as the vertices and midpoints of the edges and shortest distance is computed from each node to target. However the path generated are not the shortest distance paths due to selection of nodes. By choosing the nodes properly, a shortest (or close to shortest) distance path in a 3-D environments can be obtained.

In [8] a Grossberg Neural Network based approach was presented for achieving obstacle avoidance in a 2-D environment. To get the shortest path, entire work space (2-D environment) is descretized by a mesh and a neuron is placed at every node of the mesh. However, the size of network becomes large as the size of the work space increases. In earlier work of authors [5], a neural network based approach was presented which used the geometric properties of the work space to achieve obstacle avoidance.

In this paper, a controller design is proposed that would generate dynamically feasible trajectories for UAVs to move through a complex three dimensional environment. To simplify the computations, it is assumed that the obstacles are cuboids emerging from ground whose faces are parallel to x- and y- axis. Since the obstacles are cuboids, they can be specified by knowing the location of only two vertices i.e., each obstacle can be represented by a 6 by 1 vector consisting of minimum and maximum values of the co-ordinates. See Fig. 1 for an example.

Vivek Yadav is with the Department of Mechanical and Aerospace Engineering, University of Missouri – Rolla, MO 65401 USA. Email: vyb5b@umr.edu

Xiaohua Wang is with the Department of Mechanical and Aerospace Engineering, University of Missouri – Rolla, MO 65401 USA. Email: wxw98@umr.edu

S. N. Balakrishnan is with the Department of Mechanical and Aerospace Engineering, University of Missouri – Rolla, MO 65401 USA. , Email: bala@umr.edu, Tel: +1-573-341-4675, Fax: +1-573-341-4607

1-4244-0210-7/06/$20.00 ©2006 IEEE

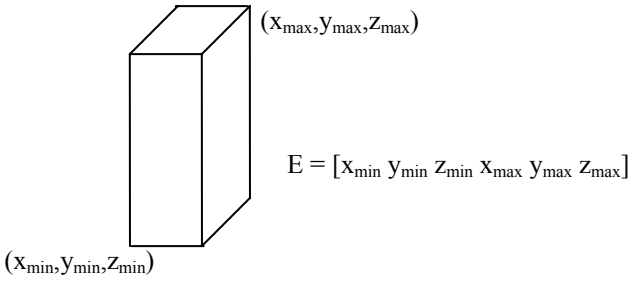

Fig. 1. Specifying an Obstacle

This paper is organized as the follows. In section II, the controller architecture is presented. The subsection A of section II presents the path planning method adopted in this paper and subsection B presents the method used to get dynamically feasible trajectory. Simulation results are presented in section III. Concluding remarks are provided in section IV.

II. Controller Architecture

This section presents the controller structure proposed to generate dynamically feasible trajectories in an urban environment. The main difference between a 2-D and a 3-D problem of getting the shortest distance path is that in 2-D environments, the shortest distance between any two points is along the vertices and edges. This is not true for 3-D environments. It is known that getting the shortest distance path in 3-D environments is very difficult and is computationally intensive [9].

In this paper, a vision based Grossberg Neural Network is used to obtain shortest distance paths in 3-D environments. The most important step is to obtain the set of vertices from which the algorithm gives the sequence of vertices to be taken by the UAVs, which is obstacle free. To get the vertices, the obstacles are sectioned with planes that contain the UAV's position and the destination and are perpendicular to faces. To reduce the computation complexity, only two planes are considered. First plane is one which is perpendicular to the top face and other plane is one which is perpendicular to the front face. The vertex where the UAV must next go is obtained by a Grossberg Neural Network whose activities indicate the closeness to target.

Once the obstacle free paths are generated, a MPC based controller is used to get the dynamically feasible path. A higher level of flexibility is achieved in the study with the use of MPC. Classical optimal control requires the differential equations to be solved and the information of the reference trajectory known prior to generating control laws for the UAVs. MPC changes tracking problems to parameter optimization problems and can deal with changes in reference conditions during the operation. State and control constraints can be handled by converting them to linear equality or inequality constraints. A lot of research has been done on nonlinear MPC and robust MPC. [10] can be looked up for more information on MPC strategies. In this paper, a GPC (Generalized Model Predictive Control) scheme is used for MPC tracking of the trajectories of the UAVs.

A vehicle with second order dynamics is considered to generate the reference trajectories.

The controller architecture is shown in Fig. 2.

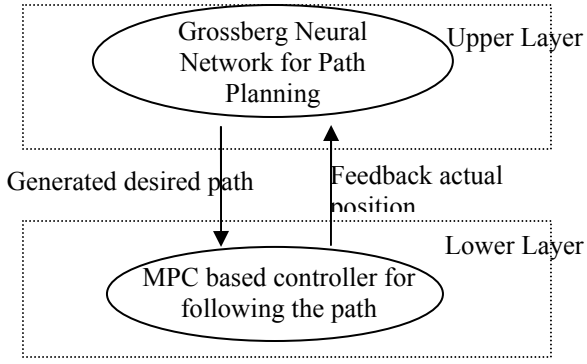

Fig. 2. Controller Architecture

A. Upper Layer

The upper layer generates paths that the UAVs have to follow for obstacle avoidance. This section presents the algorithm for getting the shortest distance paths in the upper layer. First the set of vertices are obtained from which the shortest path can be chosen. Once the vertices are obtained, a neuron is assigned to each of the vertices and the activity is obtained at each of the neurons. The activities of the neurons are used to get the path for the UAVs. The algorithm is described in detail below.

1) Getting the Vertices.

The first step is getting the vertices i.e., points on the obstacle. To do so, first a buffer zone is created around the obstacle. This buffer zone is created to account for the fact the UAV is not a point. The size of the buffer zone is determined by the dimension of the UAV. The vertices are obtained as the points of intersection of the plane containing the initial position and the target which is perpendicular to the faces with the obstacles. The vertices thus obtained are the candidates among which the next waypoint is to be

selected. There would be six vertices generated for each obstacle as the candidates for the next point. Fig. 3 shows these vertices generated for a simple case.

2) Visibility Graph

Visibility Graph is created for the topology i.e. geography of the space. Visibility Graph is defined as a map showing the lines joining each of the vertices to all the vertices that can be seen from the vertex considered. Fig. 4 shows visibility graph for a simple 2-D case.

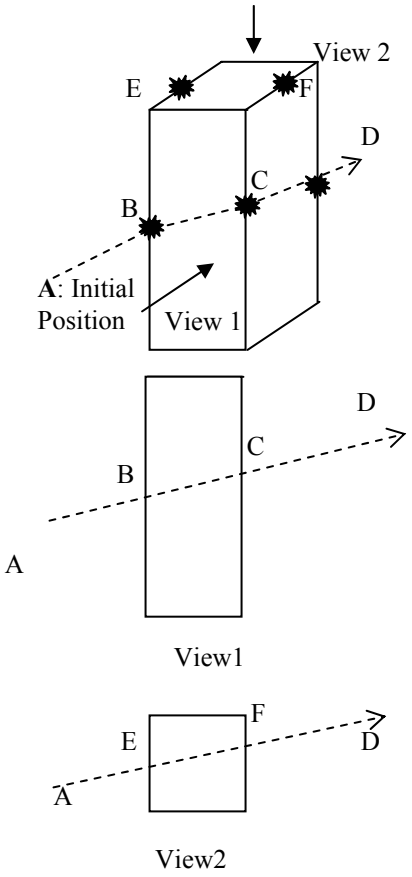

Fig. 3. Getting the Vertices

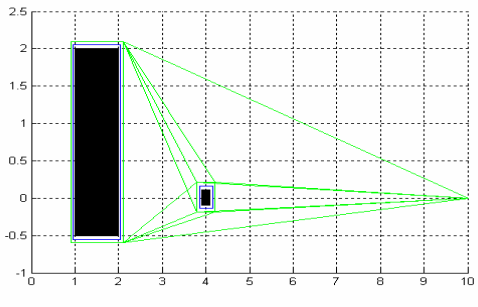

Fig. 4. Visibility Graph

To get the visibility graph, the linear programming based approach given in [7] is used.

3) Grossberg Network

Grossberg proposed a model to describe how the vision system works [18]. He proposed a shunting equation with neurons distributed in the space. Each neuron has an activity that varies with time in accordance to a shunting equation. The activity in a Grossberg network of each neuron is defined by the shunting equation as following,

$$\dot{x}_i = -Ax_i + (B - x_i)S_i^e(t) - (D + x_i)S_i^i(t) \qquad (1)$$

Where i is the neuron index, x_i is the neuron activity. S_i^e is the excitation input. S_i^i is inhibitory input. A, B and D are positive, as time decay constants. A Grossberg Network has bounded activities, and the individual neuron activities reach a steady state also by choosing the parameters properly, the rate at which the steady state is reached can be adjusted.

For the network used in this paper, the shunning equation for the activity of the neurons is,

$$\frac{dx_i}{dt} = -Ax_i + (B - x_i)\left(E + \sum_{j=1}^{k} w_{ij}x_j\right) \qquad (2)$$

Where, E the excitation input to each of the neuron is given by (3).

$$E = E_1 + E_2, \quad E_1 = \frac{\alpha}{d_{per}}, \quad E_2 = \begin{cases} 100 & \text{if vertex is the target} \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

$\alpha > 0$, d_{per} is the perpendicular distance of the vertex from the line joining the UAV and target. j is the index describing the neighboring neurons i.e. the neighboring vertices that can be seen from the i^{th} vertex and the vertex where the UAV can be at next step. w_{ij} is the weight of connection defined as μ / d_{ij}, where d_{ij} is the distance

between the i^{th} and j^{th} vertex. μ indicates how much of activity has propagated to the neighboring neurons. A, B are positive time decay constants and can be adjusted to attain the steady state faster. Once a new obstacle comes, new vertices are added to the existing map and neurons are placed at these vertices.

The activities of the neurons are generated every instant, thus the effect of changing environment is reflected by the change in the activity of the neurons.

The next point is chosen as the vertex which has the neuron with maximum activity. The time taken for the neurons to attain the steady state can be understood as the reaction time of the network to the environment. It must be noted that as the activities are generated every time and also as the effect of initial conditions does not affect the steady state value, the effect of noise in the previous stages is not carried to the path planning in the future. Also there is no effect of irrelevant obstacles on path planning.

It is important to note that the activities thus generated are stable and converge to the steady state value. Detailed analysis of stability of the Grossberg Neural Network is given in [5].

B. Bottom Layer

The bottom layer is used to generate the smooth trajectory to follow the path generated by the upper layer. The UAV is modeled as a vehicle with unit mass with the accelerations as the control inputs. This section first presents the vehicle model considered and then describes the controller.

1) Vehicle Model

The dynamics of the UAV is modeled as a double integrator given by (4)

$$\begin{aligned} \ddot{x} &= u_x \\ \ddot{y} &= u_y \\ \ddot{z} &= u_z \end{aligned} \tag{4}$$

where *x, y* and *z* indicate the position of the UAV in the X, Y and Z direction and u_x, u_y and u_z are respective control inputs. To incorporate actual vehicle dynamics, total acceleration and total velocity are constraint as $v_{\min} \le v = \sqrt{v_x^2 + v_x^2 + v_z^2} \le v_{\max}$ and $u_{\min} \le u = \sqrt{u_x^2 + u_x^2 + u_z^2} \le u_{\max}$. By taking the position and velocity of the vehicle as states, the dynamics of the system can be written as,

$$\dot{X}(k+1) = AX(k) + Bu(k) \tag{5}$$

where $X = \begin{bmatrix} x & y & z & v_x & v_y & v_z \end{bmatrix}^T$ is the state and $u = \begin{bmatrix} u_x & u_y & u_z \end{bmatrix}^T$ is the control input. The matrices A and B are given by (6).

$$A = \begin{bmatrix} I_{3\times3} & dt \times I_{3\times3} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} B = \begin{bmatrix} 0_{3\times3} \\ dt \times I_{3\times3} \end{bmatrix} \tag{6}$$

2) MPC Controller

After the trajectory is generated by the upper layer, a MPC based controller is used to track the trajectory. MPC based controllers use a discrete system model which gives the future states of the system based on future inputs and past states. This can be expressed in the form of the equation given below assuming all the velocity and positions are measurable.

$$z(k) = \psi x(k) + \Gamma u(k-1) + \Theta \Delta u(k) \tag{7}$$

and objective function is,

$$J = \sum_{k=1}^{Hz} \left\{ \left[z(k) - w(k) \right]^T Q \left[z(k) - w(k) \right] + \Delta u(k)^T R \Delta u(k) \right\} \tag{8}$$

where H_z is the predict horizon. Quadratic programming is used to obtain solution that satisfies the constraints and the control obtained is executed for certain steps (H_u the execution horizon). By this method it can be ensured that the constraints are satisfies at least at the sampling instants.

$z(k) = \begin{bmatrix} x(k \mid k) & \cdots & x(k + H_z - 1 \mid k) \end{bmatrix}^T$ is the predicted future states, $\Delta u(k) = \begin{bmatrix} \Delta u(k \mid k) & \cdots & \Delta u(k + H_u - 1 \mid k) \end{bmatrix}$ is the control increments.

$w = \begin{bmatrix} r(k \mid k) & \cdots & r(k + H_z - 1 \mid k) \end{bmatrix}$ is the future reference. Q, and R are chosen as positive symmetric weight metrics with corresponding dimensions. ψ, Γ and Θ are system parameters listed as the follows.

$\psi = \begin{bmatrix} I & A & A^2 & \cdots & A^{Hz-1} \end{bmatrix}$

$\Gamma = \begin{bmatrix} 0 & B & AB + B & \cdots & \sum_{i=0}^{Hz-2} A^i B \end{bmatrix}$

$$\Theta = \begin{bmatrix} 0 & & & \\ B & \ddots & & \\ AB+B & B & \ddots & \\ \vdots & \ddots & \cdots & 0 \\ \sum_{i=0}^{Hz-2} A^i B & \cdots & \cdots & \sum_{i=0}^{Hz-Hu-1} A^i B \end{bmatrix}$$

As $x(k)$ and $u(k-1)$ are known (7) can be written as

$$\begin{aligned} z(k) &= \Theta \Delta u(k) + f \\ f &= \psi x(k) + \Gamma u(k-1) \end{aligned} \tag{9}$$

Combining control constraints and state constraints, the optimization problem can be reformulated as the following quadratic programming problem,

$$\min_{\Delta u(k)} J = \sum_{k=1}^{Hz} \Delta u(k)^T H \Delta u(k) + G^T \Delta u(k) + C$$

$$subject\ to:$$

$$\begin{aligned} z_{\min} \le z \le z_{\max} \\ \Delta u_{\min} \le \Delta u \le \Delta u_{\max} \end{aligned} \tag{10}$$

where, $H = \left(\Theta^T Q \Theta + R\right)$, $G = 2\Theta^T Q\left(f - w(k)\right)$, and $C = \left(f - w(k)\right)^T Q\left(f - w(k)\right)$

For more details on numerical method to solve the above quadratic programming problem, please refer to [12]. The next section presents the simulation results for an urban environment.

III. RESULTS

The algorithm is tested for different configurations. The airplane constraints comes from airplane test-bed

Table. 1. UAV Parameter list

Max Velocity	35.7 m/s
Min Velocity	17.8 m/s
Max Acceleration	4.9 m/s^2

It can be seen that E is a 6 by 4 matrix. Each one of the rows indicates the obstacle configuration. The values of parameter chosen for the network are, E_2=100, A=10, B=1, $\alpha = .05$ and $\mu = .1$. The UAV is required to move at a constant velocity of 30 m/s.

The upper layer generates the shortest distance path. It returns a sequence of waypoints the UAV must visit in order to achieve the shortest distance path from the initial position to the target. A vehicle is made to follow the path at constant velocity. The path generated by the upper layer has sharp changes and the path is not dynamically feasible. Dynamically feasible paths are generated by the lower level MPC based controller. To demonstrate the effect of the MPC based controller and to show that the path generated is dynamically feasible, the position and velocity of the vehicle are shown in figures 5 – 9.

Figure 5 shows the trajectories generated by the by the upper layer. The path generated by the upper layer has sharp turns. The trajectory obtained by the MPC based controller is shown in figure 6. It can be seen that the MPC based controller gives a smoothened trajectory.

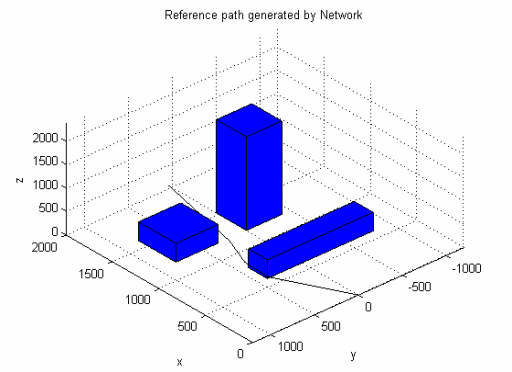

Figure 5. Trajectory generated by Upper Layer

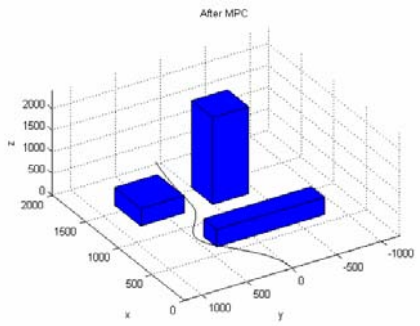

Figure 6. Trajectories generated by MPC

To compare the trajectory generated by the upper layer and the MPC based controller, the position of the UAV is plotted in figure 7. Figure 7 shows the position of the UAV as it moves towards the target following the upper layer trajectory (dotted line) and the MPC (solid) respectively. It can be seen from y vs. time plot of the two figures that there is a sharp turn at the first waypoint in the path generated by the upper layer. This sharp turn is eliminated by the MPC based controller.

To show that both velocity and acceleration constraints are met, total velocity and total acceleration of the UAV are plotted in figures 8 and 9. It can be seen that velocity is between the maximum and minimum value and the acceleration is below 4.9 m/s^2 and so both velocity and acceleration constraints are satisfied by the controller. Thus the paths generated by MPC are dynamically feasible.

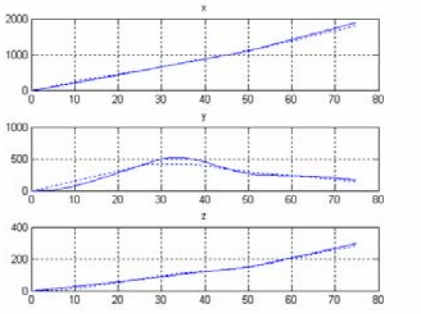

Figure 7. Trajectory generated by MPC (solid) and by the upper layer (dash)

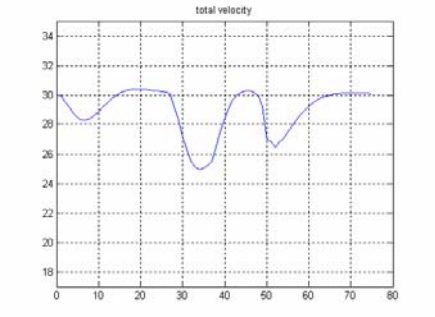

Figure 8. Total Velocity

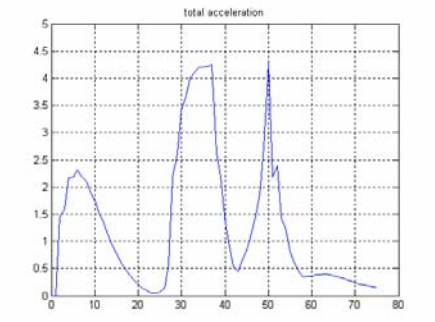

Figure 9. total acceleration

IV. CONCLUSIONS

A controller with two layer hierarchy is proposed for UAVs moving in a three dimensional environment. Simulation results were presented to show that the upper layer generates the shortest (or close to shortest) distance paths. The effectiveness of the MPC based controller was established. It was shown that the paths generated are dynamically feasible.

The computation time for the tracking controller is very small (less than .05s for quadprog in MATLAB), thus the algorithm proposed can be implemented as a two step process by assuming that the environment is known apriori. The first step being the path generation and the second step being the use of MPC based controller to follow the path. The assumption that the environment is known apriori is valid in an urban setting.

ACKNOWLEDGMENT

This research was supported by NSF grants 0201076 and 0324428.

REFERENCES

[1]. Office of the Secretary of Defense, "Unmanned Aerial Vehicles Roadmap," Tech. rep., December 2002

[2]. Suresh Jeyaraman, Antonios Tsourdos, Rafał ˙ Zbikowski and Brian White Formal Techniques for the Modeling and Validation of a Co-operating UAV Team that uses Dubins Set for Path Planning ACC05 Portland,Oregon,USA pp. 4690-4695 2005

[3]. P.B. Sujit and D. Ghose Multiple UAV Search Using Agent Based Negotiation Scheme ACC05 Portland,Oregon,USA pp. 2995-3000 2005

[4]. Eric Gagnon, C. A. Rabbath and Marc Lauzon Heading and Position Receding Horizon Control for Trajectory Generation ACC05 Portland,Oregon,USA pp. 134-139 2005

[5]. Xiaohua Wang, Vivek Yadav, S.N. Balakrishnan, Cooperative UAV Formation Flying with Stochastic Obstacle Avoidance, AIAA Guidance, Navigation and Control Conference and Exhibit 15-18 August 2005, San Francisco, California.

[6]. Richards, A., Schouwenaars, T., How, J., and Feron, E., "Spacecraft Trajectory Planning With Collision and Plume Avoidance Using Mixed-Integer Linear Programming," Journal of Guidance, Control and Dynamics, Vol. 25, No. 4, Aug 2002, pp. 755–764

[7]. Y. Kuwata and J. How, Three Dimensional Receding Horizon Control for UAVs, Massachusetts Institute of Technology, Cambridge, MA AIAA-2004-5144 AIAA Guidance, Navigation, and Control Conference and Exhibit, Providence, Rhode Island, Aug. 16-19, 2004

[8]. Max Meng and Xianyi Yang, A Neural Network Approach to Real-Time Trajectory Generation, Proceedings ofthe 1998 IEEE International Conference on Robotics & Automation, Leuven, Belgium. May 1998

[9]. Laxmi P. Gewali, Simeon Ntafos, 'Path Planning in the Presence of Vertical Obstacles,' IEEE Transaction on Robotics and Automation. Vol 6, No-3, June 1990

[10]. Morari, M. and J.H. Lee "Model Predictive Control: Past, Present and Future," Comp. Chem. Engg., 23, 667-682 (1999).

[11]. S. Grossberg, "Nonlinear neural networks: principles, mechanisms, and architecture," Neural Networks, 1:17-61, 1988.

[12]. Fletcher, R. (1987): Practical Methods of Optimization. John Wily & Sons Ltd.