THE APPLICATION OF NEURAL NETWORKS TO DRONE CONTROL

Dan Greenwood
NETROLOGIC, Inc., 5080 Shoreham Place,
Suite #201, San Diego, CA 92122

ACKNOWLEDGEMENT

The author gratefully acknowledges the contributions of Mr. Michael Myers of TRW, Dr. Christopher Bowman of Ball Systems Engineering and Mr. Fareed Stevenson of NETROLOGIC.

ABSTRACT

The FAA Sponsored a six months research program to investigate the application of neural networks to controlling aircraft. It was found that properly configured neural networks offer powerful new computationally robust methods to generate command vectors corresponding to collision free routes. Methods using neural networks which capture the expertise employed by controllers in resolving conflicts were formed. This paper shows that many of the neural network techniques applied to ATC can also be applied to drone control. Two different networks are presented: a multi-layer feed-forward network using back-propagation and a method using a potential field where a gradient measure is employed to maintain the aircraft separation in real time.

SUMMARY

During our study for the FAA it was found that very restricted conflict detection and resolution functions could be performed using such familiar approaches but vastly more complex learning and modelling paradigms were required to solve non-trivial air traffic control problems using methods which essentially mimic methods employed by actual controllers.

Neural networks provide solutions that are efficiently generated by parallel processors in real time and can “capture” knowledge observed from experts via training to incorporate many intuitive relationships and rules which experts use but cannot articulate. Thus, efficient parallel operation and adaptivity via training both are embedded in the capabilities desirable from neural networks. As such, neural networks are a synergistic and complementary technology with expert systems [Cross] and traditional control techniques.

NETROLOGIC's research to apply neural network methodologies to FAA Air Traffic Control (ATC) problems is an entirely new application of neural networks. Ultimately, the following advantages can be derived from the use of neural networks in ATC:

1) A powerful ATC training aid which contains a robust representation of what a real controller does by use of neural network paradigms and architectures.

2) A real-time decision aid which is trainable by each controller.

3) A simulation of the air traffic environment can be used to train networks to generate collision free paths.

During the study it was found that the network representation problem (selecting network input data sets which represent the air traffic environment was very difficult. A representation which associates one neural network with each aircraft vector and combining it with a gradient representation for other interacting aircraft. A network which defines a potential field and an interaction architecture area was able to sequence twenty aircraft into a successful landing pattern while maintaining separation standards.

This research was the initial effort by the FAA to apply neural networks to air traffic control problems. Despite great progress in fielding the latest computational and data processing tools into the nation's Air Route Traffic Control Centers, TRACONs, and terminal facilities the ever expanding growth in air traffic and diversity of aircraft performance capabilities requires that new methods for enhancing computational capabilities be continually investigated. Neural networks offer a revolutionary information processing capability which in many respects duplicate the performance of biological systems ([Lippmann], [Greenwood,88], and

[DARPA] for an overview of the state of the art of neural networks). It is pointed out in [Cross], [Wesson] and [Greenwood,73] that controlling aircraft even with modern computational tools and procedures is more an art or skill than a science or procedure. Controllers get better at their jobs as they spend time at their consoles detecting real or simulated conflicts and formulating methods to resolve the conflicts. They develop a sense for what is right or wrong with the flow of traffic and particular encounters. It takes many years to develop this sense or level of expertise and as yet their are no clearly articulated set of rules or procedures which captures the essence of good control practice. Neural networks technology offers a tool which can be trained by example to yield desired decision aids / commands for drone control as well as air traffic control.

There is a growing interest by military, aerospace and civil agencies to augment or improve the performance of humans performing difficult control tasks by applying parallel architectures and connectionism to real-time decision making. Interpreting the air traffic control environment and reacting to a temporally presented high dimensional world is an ongoing problem. Automating aspects of air traffic situation assessment and control through the development of connectionist (i.e. neural network) methodologies appears to have significant potential. Neural networks are computers that learn how to generate approximate solutions to problems based upon sample data and built-in learning mechanisms. In other words, neural networks can be trained to identify, on their own, the key features which enable them to distinguish different patterns. A neural network can learn on-line in real-time or can be trained by a user with a sample training set. They do not require expert knowledge representation, logical inferencing schemes, statistical algorithms, or a programmer to develop and code a solution to a user’s problems. However, they do require an architecture with sufficient capacity and a training scheme. Also, neural networks do not provide responses for which no training has occurred nor step-by-step explanations as to how answers are achieved. Thus, neural networks provide a complementary addition to conventional numeric or symbolic (i.e. expert systems, AI) and Von Neumann processing for problems requiring pattern recognition type tasks (see Figure 1)

	APPROACH TO PROBLEM “SOLVE THEN PROGRAM VS TRAIN:				PERFORMANCE “ACCURATE VS COST EFFECTIVE”				
	EFFECTIVE PROBLEM DOMAINS	PROBLEM SOLVING	SOLUTION CONVERGENCE	GROWTH IN PROBLEM COMPLEXITY RECURES	SYSTEM DEVELOPMENT COST	LONG-TERM KNOWLEDGE	EXECUTIVE CONTROL	TESTABILITY	REALIAILITY
MODESTLY PARALLEL NUMERICAL AND PROCESSING	KNOWN FEATURES AND STATISTICS	SOLUTION TECHNIQUES PROGRAMMED SO MORE EFFICIENT	TIMELINESS DEGRADES WITH ACCURACY	MORE SOLUTION TIME AND MEMORY	MORE SOFTWARE INTENSIVE: HIGHER COST	IN LARGE COMPACT DATABASES: SLOWER RETRIEVAL	RULE-DRIVEN TOP=DOWN	LOCAL REPRESEN-TATIONS: CAN EXPLAIN WHY AND HOW	FAULT RECOVERY PLANNED
MASSIVELY PARALLEL ADAPTIVE “NEURAL” PROCESSING	UNKNOWN FEATURES AND STATISTICS	LEARNS SOLUTION VIA SOLUTION EXEMPLARS	TIMELY APPROXIMATE SOLUTIONS	NEW CLUSTERED CLUSTERED NETWORKS	MORE HARDWARE INTENSIVE: LOWER COST	DISTRIBUTED IN CONNECTION WEIGHTS: EFFICIENT RETRIEVAL	SELF-ORGANIZED BOTTOM-UP	DISTRIBUTED REPRESENTATIONS: TESTED CASE BY CASE	INHERENTLY FAULT TOLERANT

Figure 1 Complementarily of Neural Networks and Conventional Processing

Shortly after the beginning of our Phase I research program we decided to use the TRACON ATC simulation program as an ATC simulation environment to be controlled by network control programs developed for this project. There were several advantages and considerations that made TRACON the desirable choice as a simulation.

The first is that the time required to develop a flexible simulation that could be easily demonstrated and accurately represent the Air Traffic Control process would be long. There is a great deal that would have to be modeled about the information normally available to an Air Traffic Controller and the operations of aircraft. TRACON has a reasonable subset of required information already built into the program. In addition, TRACON has a very good graphics interface and is easy to use so that a user can control the simulation by hand. As a result of using TRACON our attention could be focused on developing the network controller software.

Another advantage of using the TRACON program is a built in scoring feature. In training the neural network controller, this feature offers a hands off approach for network adaptation. With it the neural network can control the TRACON game simulation and evaluate its performance based on the scoring feature in TRACON thereby making adjustments using a reinforcement learning procedure as outlined by Barto and Sutton [Barto]. The network controller can adapt the network weights to improve the network performance with respect to the TRACON simulation. Using the TRACON program the human controller can control the TRACON simulation and the aircraft successfully.

Network Input and Output Representations

Successful network input representation depends on meeting a number of specific requirements. First, it is important that the network be capable of generalizing from a small set of training examples. To achieve this with a given set of training examples, the first requirement is that the dimensionality of the input space to the network be small. If a network is given a large input vector, on the order of hundreds of elements, then it would be required to have training examples for all reasonable combinations of inputs and this could be a very large number of training examples. The number of weights in the associated network grows rapidly when the number of inputs are large and in order to achieve adequate generalization a large training set is required. However, having low dimensional input allows the training set size to be small. In this case the probability of generalizing from a small training set is high and the network can run faster since fewer weights will be required for the feedforward calculations.

A second requirement is that the input dimensionality be fixed. For example, if we decided to input the position and velocity of each of the aircraft on the screen to the network with a variable number of aircraft, then we would have a variable number of input for any given situation. This is impractical with respect to the current level of neural network technology since most neural networks require a fixed input space, in other words a fixed input dimensionality. Therefore, it is desirable to have an input representation which can in some way be independent of the number of active aircraft in the system. Two choices were made regarding the input representation to the net. First, it was realized that for multiple aircraft the easiest way to handle the situation was to provide the network with information that was representative of the situation as seen by a single aircraft, instead of having a global view for the situation which basically represents the normal view of an Air Traffic Controller. Such a local view requires that each aircraft has an associated neural network. For each of these local networks associated with each aircraft, the inputs must contain information about the other aircraft on the screen. This local view provides duplicate copies of the network produced for each of the aircraft on the screen, and we are not limited by any dimensionality constraints on the number of aircraft that could be handled at a given time.

The second problem regarding representation is that there are a great number of potential commands that the network could issue, for example, the network could issue a “clear direct to” command when there is a large number of intersections that would be potential candidates for modifying that command. To solve this problem it is best have a combination of a neural network and an expert system which processes the information provided by the neural network. The neural network could learn the fuzzy rules for performing well in a simulation run and the weights of the network would adapt to optimize that performance. The expert system section of the controlling software would handle the situation details. There would be no need for the network to have an output for each of the potential destinations for a given command since they are known to the expert system section. For example for the “cleared direct to ...” command there are only a limited number of reasonable alternatives for the associated destination. The expert system can determine in a fairly straight forward manner the legitimate alternatives for each type of command. Thus, it is the task of the neural network to choose the form of the proper command, if any, when the network is executed, followed by the expert system. For example, if there were a separation conflict at a given altitude, the network output could be “climb to ...” and the available altitude would be determined by the expert system software. By comparison, if a network output were assigned for each available altitude then the number of network outputs, one for each command/modifier combination, would be high and generalization would be difficult to achieve.

With only one copy of the network being local to each aircraft it is still necessary to deal with the issue of how to represent an aircraft’s potential for conflict. The information needed to determine a potential conflict for a given aircraft is the positions and velocities of any remaining aircraft on the screen. Calculations can accurately be made to determine if separations might be violated, by extrapolating from the current state of the system out for a predetermined length of time. The result of this calculation could set a conflict flag as an input to the neural network controller. This piece of information in itself is not sufficient to resolve the conflict. The key to the input representation problem can be found in the ATC Conflict Field Interaction Model developed from TRW’s Electronic Warfare model. The solution derived for this problem was essentially a coarse coding of the information relating to the other aircraft. Since there could be any number of aircraft on the screen at a given time, it was determined through simulations of the TRW system that it would be sufficient to provide the network controller with a gradient vector that represented the accumulated repulsion vector for each of the other aircraft on the screen. This gradient vector could be calculated by the aircraft and the resulting vector, which would represent the sum of the interaction to each of the other aircraft, could be presented to the neural network controller.

The basic idea of a gradient vector is that as one aircraft gets closer to another aircraft a repulsion force vector is developed between the aircraft which is inversely proportional to the distance between the aircraft. An associated potential field is centered at each aircraft which may be circular aircraft. The field geometry should take into account the separation requirements and the anticipated position of the aircraft at some delta time period from the current time. Aircraft fly in a forward direction and therefore its repulsion gradient vector should reflect this fact. The repulsion gradient vector is illustrated in Figure 2 in two dimensions, as a contour plot of the gradient. It shows that an area directly in front of the aircraft has the highest conflict potential and an area behind the aircraft has a low potential since the aircraft is moving away from that point and towards the points in front of the aircraft. Simulation runs at NETROLOGIC using both repulsion and attraction gradients have illustrated that with continuous local control on each aircraft most conflicts can be resolved solely with such gradient information. Attractor gradients are also used to indicate the aircraft closeness and relative position to waypoints, destination airports and flight paths. The use of these gradients is modified, however, in the network controller since the ATC Network Controller cannot provide proportional control to each aircraft. It is necessary for the network controller to provide a limited number of discrete commands at varying intervals of time. Thus, the network controller uses the gradient information to trigger certain command sequences to perform the required avoidance maneuvers.

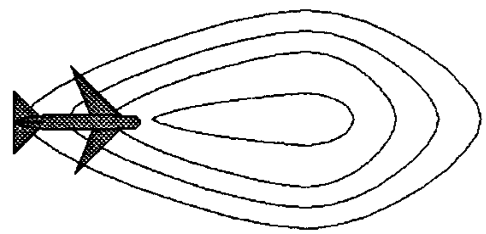

Figure 2 Repulsion Gradient Vector Contours

The basic processing flow for the ATC network controller program is given in Figure 3. When the ATC network controller program is loaded under the DoubleDos operation environment, the ATC network controller loads the TRACON program and begins to send keyboard commands to TRACON to start the simulation. The first operation performed is to erase the radar sweep and the ground outline from the radar display. This operation is performed to simplify the identification of aircraft on the screen to process conflicting pixels in the video graphics interface. The ATC network controller then checks for pending and active aircraft on the radar screen. When a pending aircraft is found, keyboard commands are issued to accept the aircraft. Once accepted, the aircraft position and flight plan information is “read” from the TRACON graphics screen. Only pixel information from the screen is available to the ATC network controller. The program actually uses a template matching character recognition method to “read” the information. For each active aircraft, the current position of the aircraft is determined, and the input information for the neural network is generated.

The inputs to the neural network are as follows:

1. Conflict in the next t minutes.
2. Is other aircraft going to the same destination.
3. Is the next waypoint a tower.
4. Is the next waypoint an intersection.
5. Is the next waypoint a center.
6. X attraction gradient vector.
7. Y attraction gradient vector.
8. Z attraction gradient vector.

9. X repulsion gradient vector.
10. Y repulsion gradient vector.
11. Z repulsion gradient vector.

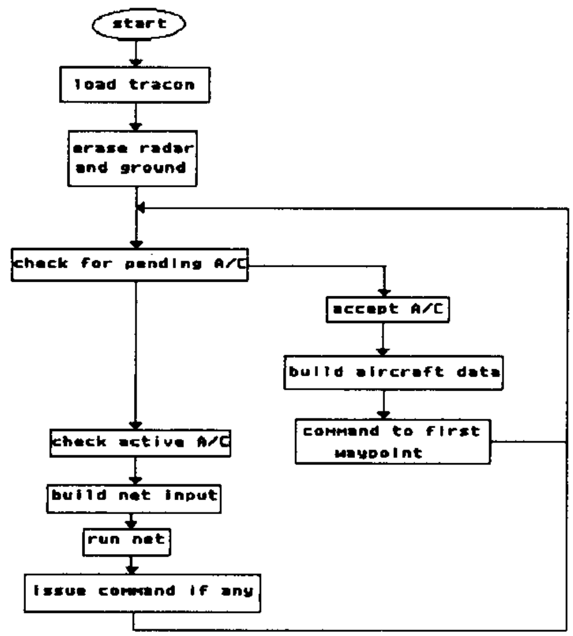

Figure 3 ATC Neural Network Controller TRACON Interface

The outputs of the neural network are defined as follows;

1. Change Altitude.
2. Change speed.
3. Change next waypoint.
4. Issue “cleared direct to ... ”for a tower.
5. Issue “cleared direct to ... ”for a center.
6. Issue “cleared direct to ... ” for an intersection.
7. Issue handoff to tower command.
8. Issue handoff to center command.
9. Set handoff command altitude and speed.
10. Increase speed.
11. Increase altitude.
12. Turn right.
13. Turn left.

Except for the gradient inputs, all the inputs and outputs to the neural network are binary, taking on TRUE/FALSE or YES/NO meanings.

Network Architecture

There are many neural network architectures that could be applied to this problem. The most commonly used network type is the feedforward network. Since this network cannot develop its own internal state some form of feedback is required to maintain any state information. In the ATC environment the state information is as follows:

1. The position and velocity of each aircraft.
2. The flight plan for each aircraft.
3. The commands issued to each of the aircraft.
4. Information regarding the ATC environment such as, the location of airways and intersections, and the location and headings for airports.
5. Weather related information.

For the ATC network controller, all of this information is maintained outside the network and therefore the system can be controlled by a feedforward network. In addition, it should be noted that feedforward networks are easier to train than feedback type networks.

Under the category of feedforward networks, there are several types and several training methods. Kohonen learning can allow a feedforward network to self-organize on the input data. Also, supervised learning methods like backpropagation and MADALINE rule II can be used to train feedforward networks. The MADALINE consists of threshold logic units and the backpropagation network uses “sigmoidal” processing elements. The threshold logic element has several properties that make it desirable for use in the network controller. Threshold logic can be trained in several ways including modified backpropagation, MADALINE rule II and by a reinforcement learning technique called the associative reward/penalty algorithm. Furthermore, it is possible to load the weights to this kind of network “by hand” (e.g. most neural net courses have students build a network for the parity or XOR problem using threshold logic by inspection).

BASELINE MODEL

Providing decision aids for an air traffic controller involves interpretation and learning based on a sampled and quantified environment followed by commands to aircraft. Such a system should ultimately have the ability to:

1) Deal with very high dimension environments (a variable number of aircraft and complex scenes).
2) Be able to react rapidly (reflex) to unambiguous environments.
3) Reflect on planned actions in complex scenes in real time.
4) Execute exploratory learning behaviors designed to enhance knowledge of the environment when it is ambiguous.
5) Resolve conflicts for resources (prioritize message traffic).

These features are incorporated into our baseline model which provides a conceptual framework for embedding the work done in this phase of the study into a complete system in Phase II.

Our basic model satisfying the five requirements listed above is illustrated in Figure 4. The above model processes an air traffic conflict vector, T, generated by a situation assessment preprocessor indicated by (1) in the figure. The model has two processing paths producing both a reflex response and a reflective response. Both responses create a request for resources which may conflict, so that conflicts which may arise must be adjudicated in a conflict resolution (7) function. The reflex response converts known environment interpretation T vectors or temporal sequences of T vectors into responses. The reflective networks which do planning react to an internally generated environmental representation formed by the inner model. The inner model represents both linear and nonlinear aircraft interactions. In a neural network object oriented version of an inner model, the activation at the network nodes represent a measure of net aircraft collision probability or threat, while planning is done by minimizing this probability over a mission, i.e., getting aircraft to their destinations and efficiently (minimizing fuel consumption) within the airspace available while maintaining safe distance separations. Evaluation of the air traffic control environmental feedback or of a proposed effect on the ATC environment will be processed in a response evaluation unit.

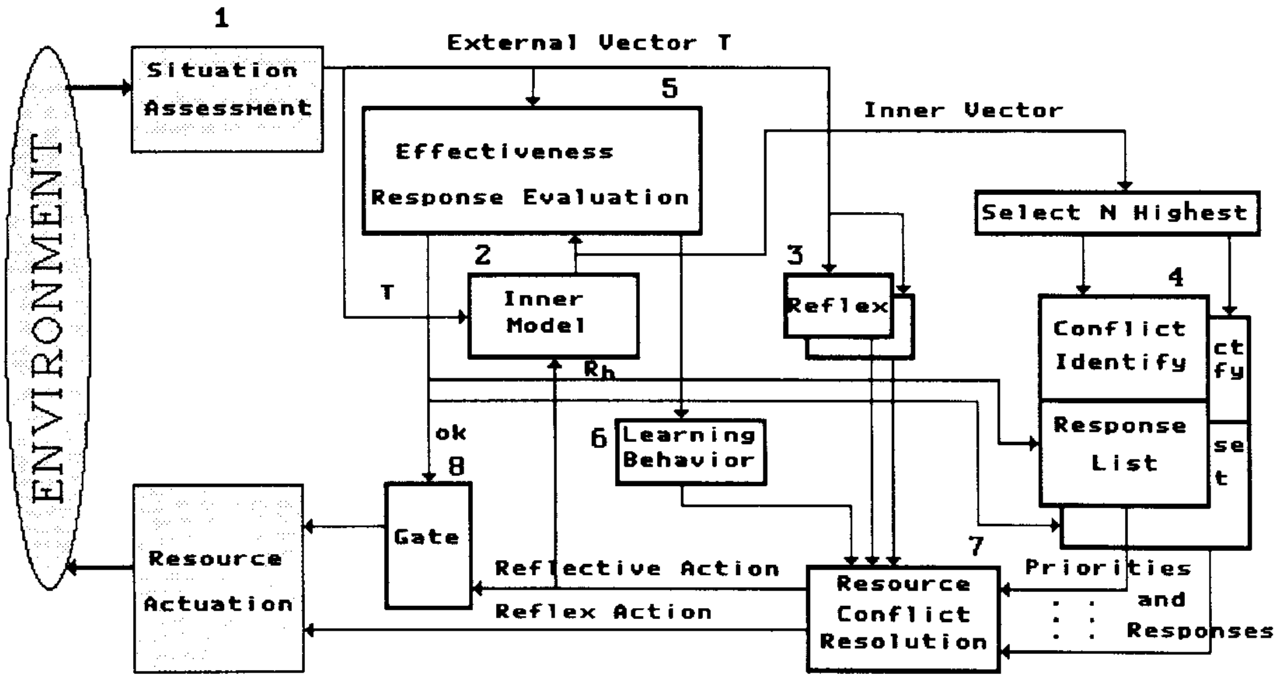

Figure 4 Baseline Cognitive Model

Finally, whenever the ATC environment input T cannot be evaluated, or when all possible response lead to unsatisfactory results in the ATC environment, it may be desirable that a learning behavior be activated in order to improve the situation. The learning behavior is activated by environmental feedback. In a more advanced model, the learning behavior would be tested on an inner model as a planning phase. The use of learning behaviors in any future cognitive ATC system may hold some risk and may not be advisable.

ATC Environment Characterization

The task of mapping an arbitrarily dimensioned, time varying internal representation sufficiently rich to permit the cognitive processor to cope with any situation is critical for success in an autonomous system. In a connectionist system representation of the environment is input via a set of sensors. The number of objects or classes discernable becomes a variable internal representation. For example, the air traffic environment under control may contain a previously unspecified number of objects of potential interest to each pilot, each object belonging to a conflict group, and conflict groups may in turn belong to different classes. In practice the unbounded dimensional world must first be quantized and limited for presentation to a topologically connected information processing system.

When dealing with multiple conflicts, only a fixed number may be dealt with simultaneously by considering their individual properties. As the number of objects in a scene increases, one is forced to prioritize and select the ones to process, for example, those in closest proximity, the fastest, etc. As the numbers of objects in the field of view increase still further, individual object conflict descriptions are augmented by group conflict descriptions. Group properties are cognitively emergent and may include such items as group type, center of mass, cluster volume or shape, distribution (aircraft formation for example), group velocity, number of objects, group intention and so on.

THE AIR TRAFFIC CONTROLLER INNER MODEL

A complete “cognitive” ATC controller system has elements of rapid reflex response, planning and the means for measuring the effects of the controller’s commands. In order to deal with the temporal aspects of aircraft conflicts, an autonomous controller should rely on the use of inner models. These inner models generate test outcomes of possible responses generated in the reflex and reflective networks. These knowledge intensive models demand a methodology with which to gather expert flight controller knowledge by rule or by observation. The knowledge is used to instantiate the system to cope with uncertainty and novelty on a dynamic basis.

ATC Conflict Field Interaction Model

The development of the ATC Conflict Field Interaction Model was originally motivated by the difficulty we anticipated in representing and processing spatial information in a neural network. In a general air traffic environment controllers must keep track of the relative locations and velocities of many planes and react accordingly. Before TRW’s research program started, there were no network models which were able to deal well with spatial information and motion control. Consequently, TRW designed a network which can maintain and update spatial coordinates for all the objects in an arbitrary scenario simultaneously based on flight objectives/intent and perceived collision threats.

The ATC Conflict Field Interaction Model is used to simulate the conflict environment. Traditional simulations use pre-defined trajectories or waypoints to control the motion of objects, however, the ATC Conflict Field Interaction Model uses a heuristic motion control algorithm incorporating field effect interactions and gradient descent to minimize accident probability. Therefore, it can be thought of as an autonomous vehicle control system or a flight path planner as well as a conflict threat simulation.

The system essentially has two systems which operate somewhat independently: the motion controller and the conflict simulator. The motion control method is based on the principle of attraction / repulsion fields between objects arising from mission objectives and perceived conflicts. The network combines each object’s multiple goals to create a trajectory which will allow an aircraft to fly it trajectory free of conflicts.

The motion control algorithm was designed with a parallel neural network implementation in mind. The proposed network implementation would result in a system with constant processing time independent of the number of objects in the scenario. The conflict simulations are state-transition systems which operate independently. Their independent operation makes them a good candidate for parallel implementation, but we did not design our network for this purpose.

REFERENCES

Albert, A., “Regression and the Moore-Penrose Pseudoinverse”, Academic Press, 1972.

Barto, A., “Neuron-like Adaptive Elements That Can Solve Difficult Learning Control Problems”, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-B, No. 5, September/October 1983.

Cross, Captain Stephen E., “Model-Based Reasoning in Expert Systems: An Application to Enroute Air Traffic Control”, AESS Newsletter, November 6, 1984.

DARPA, “Neural Network Study”, Fairfax, VA, AFCEA International Press, 1988.

Greenwood, D., “NASA JSC Neural Networks Survey Results:”, NASA CP-2491, First Annual Workshop on Space Operations, Automation and Robotics, 1988.

Greenwood, D., “En Route Conflict Alert Sensitivity Analysis”, WP-10383, Mitre Corp., McLean, VA, August 1973.

Greville, T., “Studies in Applied Mathematics”, Rev. 2, 15, 1960.

Grossberg, S., “A Prediction Theory For Some Non-Linear Functional-Differential Equations: I. Learning Of Lists”, Journal of Mathematical Analysis and Applications, 22, 643-694, 1968.

Hopfield, J., “Neural networks and physical systems with emergent collective computational abilities”, Proceedings of the National Academy of Sciences USA, 79, 2554-2558, 1982.

Kohonen, T., “Self-Organization and Associative Memory”, Springer-Verlag, Berlin, 1984.

Lippmann, R., “An Introduction to Computing with Neural Nets”, IEEE, ASSP Magazine, April 1987.

Myers, Turner, Kuczewski and Simpson, “ANCP Adaptive Network Cognitive Processor” Final Report for Period March 1987-February 1988, AFWAL-TR-89-1017, Vol I and II.

Rumelhart, D. and McClelland, J., “Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Volumes 1 and 2" , Bradford Books/MIT Press, Cambridge, 1986.

Werbos, P., “Beyond regression: New tools for prediction and analysis in the behavioral sciences”, PhD Dissertation, Harvard University, 1974.

Wesson, R., “TRACON User’s Manual”, Wesson International, Austin, TX, 1988.