This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

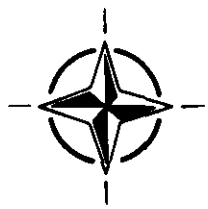
AGARD
ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT
7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD LECTURE SERIES 179

Artificial Neural Network Approaches in Guidance and Control

(Les Réseaux Neuroniques Artificiels, Voie à Explorer dans le Domaine du Guidage et du Pilotage)

This material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel of AGARD and the Consultant and Exchange Programme of AGARD presented on 8th—9th October 1991 in Monterey, United States, 14th—15th October 1991 in Kjeller (near Oslo) and 17th—18th October 1991 in Neubiberg, Germany.



North Atlantic Treaty Organization
Organisation du Traité de l'Atlantique Nord

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The Mission of AGARD

According to its Charter, the mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community;
- Providing scientific and technical advice and assistance to the Military Committee in the field of aerospace research and development (with particular regard to its military application);
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Exchange of scientific and technical information;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.

The content of this publication has been reproduced
directly from material supplied by AGARD or the authors.

Published September 1991

Copyright © AGARD 1991
All Rights Reserved

ISBN 92-835-0635-9

Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

AGARD Lecture Series 179 Advisory Group for Aerospace Research and Development, NATO ARTIFICIAL NEURAL NETWORK APPROACHES IN GUIDANCE AND CONTROL Published September 1991 180 pages Ever increasing operational and technical requirements have led to highly integrated flight, guidance and control, and weapons delivery systems. The effective implementation of these functions makes the fusion and interpretation of sensor data and the multifunctional use of sensor information inevitable. P.T.O	AGARD-LS-179 Guidance and control Flight control Missile guidance Weapon delivery Neural nets Data fusion Parallel processing	AGARD Lecture Series 179 Advisory Group for Aerospace Research and Development, NATO ARTIFICIAL NEURAL NETWORK APPROACHES IN GUIDANCE AND CONTROL Published September 1991 180 pages Ever increasing operational and technical requirements have led to highly integrated flight, guidance and control, and weapons delivery systems. The effective implementation of these functions makes the fusion and interpretation of sensor data and the multifunctional use of sensor information inevitable. P.T.O	AGARD-LS-179 Guidance and control Flight control Missile guidance Weapon delivery Neural nets Data fusion Parallel processing
AGARD Lecture Series 179 Advisory Group for Aerospace Research and Development, NATO ARTIFICIAL NEURAL NETWORK APPROACHES IN GUIDANCE AND CONTROL Published September 1991 180 pages Ever increasing operational and technical requirements have led to highly integrated flight, guidance and control, and weapons delivery systems. The effective implementation of these functions makes the fusion and interpretation of sensor data and the multifunctional use of sensor information inevitable. P.T.O	AGARD-LS-179 Guidance and control Flight control Missile guidance Weapon delivery Neural nets Data fusion Parallel processing	AGARD Lecture Series 179 Advisory Group for Aerospace Research and Development, NATO ARTIFICIAL NEURAL NETWORK APPROACHES IN GUIDANCE AND CONTROL Published September 1991 180 pages Ever increasing operational and technical requirements have led to highly integrated flight, guidance and control, and weapons delivery systems. The effective implementation of these functions makes the fusion and interpretation of sensor data and the multifunctional use of sensor information inevitable. P.T.O	AGARD-LS-179 Guidance and control Flight control Missile guidance Weapon delivery Neural nets Data fusion Parallel processing

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Neural networks, consisting of parallel microcomputing elements, hold great promise for guidance, navigation and control applications because of their ability to learn and acquire knowledge.

The Lecture Series will bring together a group of NATO nation speakers with outstanding experience in this new area of technology. First they will review the fundamentals of neural networks to serve as background so that advances in this new, rapidly evolving technological area can be both understood and appreciated. They will then discuss a number of related applications of direct benefit to the attendees.

This Lecture Series, sponsored by the Guidance and Control Panel of AGARD, has been implemented by the Consultant and Exchange Programme.

ISBN 92-835-0635-9

Neural networks, consisting of parallel microcomputing elements, hold great promise for guidance, navigation and control applications because of their ability to learn and acquire knowledge.

The Lecture Series will bring together a group of NATO nation speakers with outstanding experience in this new area of technology. First they will review the fundamentals of neural networks to serve as background so that advances in this new, rapidly evolving technological area can be both understood and appreciated. They will then discuss a number of related applications of direct benefit to the attendees.

This Lecture Series, sponsored by the Guidance and Control Panel of AGARD, has been implemented by the Consultant and Exchange Programme.

ISBN 92-835-0635-9

Neural networks, consisting of parallel microcomputing elements, hold great promise for guidance, navigation and control applications because of their ability to learn and acquire knowledge.

The Lecture Series will bring together a group of NATO nation speakers with outstanding experience in this new area of technology. First they will review the fundamentals of neural networks to serve as background so that advances in this new, rapidly evolving technological area can be both understood and appreciated. They will then discuss a number of related applications of direct benefit to the attendees.

This Lecture Series, sponsored by the Guidance and Control Panel of AGARD, has been implemented by the Consultant and Exchange Programme.

ISBN 92-835-0635-9

Neural networks, consisting of parallel microcomputing elements, hold great promise for guidance, navigation and control applications because of their ability to learn and acquire knowledge.

The Lecture Series will bring together a group of NATO nation speakers with outstanding experience in this new area of technology. First they will review the fundamentals of neural networks to serve as background so that advances in this new, rapidly evolving technological area can be both understood and appreciated. They will then discuss a number of related applications of direct benefit to the attendees.

This Lecture Series, sponsored by the Guidance and Control Panel of AGARD, has been implemented by the Consultant and Exchange Programme.

ISBN 92-835-0635-9

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

AGARD

NATO OTAN

7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE

FRANCE

Téléphone (1)47.38.57.00 · Télex 610 176
Télécopie (1)47.38.57.99

DIFFUSION DES PUBLICATIONS
AGARD NON CLASSIFIEES

L'AGARD ne détient pas de stocks de ses publications, dans un but de distribution générale à l'adresse ci-dessus. La diffusion initiale des publications de l'AGARD est effectuée auprès des pays membres de cette organisation par l'intermédiaire des Centres Nationaux de Distribution suivants. A l'exception des Etats-Unis, ces centres disposent parfois d'exemplaires additionnels; dans les cas contraire, on peut se procurer ces exemplaires sous forme de microfiches ou de microcopies auprès des Agences de Vente dont la liste suite.

CENTRES DE DIFFUSION NATIONAUX

ALLEMAGNE
Fachinformationszentrum,
Karlsruhe
D-7514 Eggenstein-Leopoldshafen 2

BELGIQUE
Coordonnateur AGARD-VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Directeur du Service des Renseignements Scientifiques
Ministère de la Défense Nationale
Ottawa, Ontario K1A 0K2

DANEMARK
Danish Defence Research Board
Ved Idraetsparken 4
2100 Copenhagen Ø

ESPAGNE
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

ETATS-UNIS
National Aeronautics and Space Administration
Langley Research Center
M/S 180
Hampton, Virginia 23665

FRANCE
O.N.E.R.A. (Direction)
29, Avenue de la Division Leclerc
92320, Châtillon sous Bagneux

GRECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ISLANDE
Director of Aviation
c/o Flugrad
Reykjavik

ITALIE
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

LUXEMBOURG
Voir Belgique

NORVEGE
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PAYS-BAS
Netherlands Delegation to AGARD
National Aerospace Laboratory NLR
Kluyverweg 1
2629 HS Delft

PORTUGAL
Portuguese National Coordinator to AGARD
Gabinete de Estudos e Programas
CLAFA
Base de Alfragide
Alfragide
2700 Amadora

ROYAUME UNI
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

TURQUIE
Milli Savunma Başkanlığı (MSB)
ARGE Daire Başkanlığı (ARGE)
Ankara

LE CENTRE NATIONAL DE DISTRIBUTION DES ETATS-UNIS (NASA) NE DETIENT PAS DE STOCKS DES PUBLICATIONS AGARD ET LES DEMANDES D'EXEMPLAIRES DOIVENT ETRE ADRESSEES DIRECTEMENT AU SERVICE NATIONAL TECHNIQUE DE L'INFORMATION (NTIS) DONT L'ADRESSE SUIT.

AGENCES DE VENTE

National Technical Information Service
(NTIS)
5285 Port Royal Road
Springfield, Virginia 22161
Etats-Unis

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Division
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
Royaume Uni

Les demandes de microfiches ou de photocopies de documents AGARD (y compris les demandes faites auprès du NTIS) doivent comporter la dénomination AGARD, ainsi que le numéro de série de l'AGARD (par exemple AGARD-AG-315). Des informations analogues, telles que le titre et la date de publication sont souhaitables. Veuiller noter qu'il y a lieu de spécifier AGARD-R-nnn et AGARD-AR-nnn lors de la commande de rapports AGARD et des rapports consultatifs AGARD respectivement. Des références bibliographiques complètes ainsi que des résumés des publications AGARD figurent dans les journaux suivants:

Scientifique and Technical Aerospace Reports (STAR)
publié par la NASA Scientific and Technical
Information Division
NASA Headquarters (NTT)
Washington D.C. 20546
Etats-Unis

Government Reports Announcements and Index (GRA&I)
publié par le National Technical Information Service
Springfield
Virginia 22161
Etats-Unis

(accessible également en mode interactif dans la base de données bibliographiques en ligne du NTIS, et sur CD-ROM)

Imprimé par Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

AGARD

NATO OTAN

7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE

FRANCE

Telephone (1)47.38.57.00 · Telex 610 176
Telefax (1)47.38.57.99

DISTRIBUTION OF UNCLASSIFIED AGARD PUBLICATIONS

AGARD does NOT hold stocks of AGARD publications at the above address for general distribution. Initial distribution of AGARD publications is made to AGARD Member Nations through the following National Distribution Centres. Further copies are sometimes available from these Centres (except in the United States), but if not may be purchased in Microfiche or Photocopy form from the Sales Agencies listed below.

NATIONAL DISTRIBUTION CENTRES

BELGIUM
Coordonnateur AGARD — VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Director Scientific Information Services
Dept of National Defence
Ottawa, Ontario K1A 0K2

DENMARK
Danish Defence Research Board
Ved Idraetsparken 4
2100 Copenhagen Ø

FRANCE
O.N.E.R.A. (Direction)
29 Avenue de la Division Leclerc
92320 Châtillon

GERMANY
Fachinformationszentrum
Karlsruhe
D-7514 Eggenstein-Leopoldshafen 2

GREECE
Hellenic Air Force
Air War College
Scientific and Technical Library
Dekelia Air Force Base
Dekelia, Athens TGA 1010

ICELAND
Director of Aviation
c/o Flugrad
Reykjavik

ITALY
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
Aeroporto Pratica di Mare
00040 Pomezia (Roma)

LUXEMBOURG
See Belgium

NETHERLANDS
Netherlands Delegation to AGARD
National Aerospace Laboratory, NLR
Kluyverweg 1
2629 HS Delft

NORWAY
Norwegian Defence Research Establishment
Attn: Biblioteket
P.O. Box 25
N-2007 Kjeller

PORTUGAL
Portuguese National Coordinator to AGARD
Gabinete de Estudos e Programas
CLAFA
Base de Alfragide
Alfragide
2700 Amadora

SPAIN
INTA (AGARD Publications)
Pintor Rosales 34
28008 Madrid

TURKEY
Milli Savunma Başkanlığı (MSB)
ARGE Daire Başkanlığı (ARGE)
Ankara

UNITED KINGDOM
Defence Research Information Centre
Kentigern House
65 Brown Street
Glasgow G2 8EX

UNITED STATES
National Aeronautics and Space Administration (NASA)
Langley Research Center
M/S 180
Hampton, Virginia 23665

THE UNITED STATES NATIONAL DISTRIBUTION CENTRE (NASA) DOES NOT HOLD STOCKS OF AGARD PUBLICATIONS, AND APPLICATIONS FOR COPIES SHOULD BE MADE DIRECT TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS) AT THE ADDRESS BELOW.

SALES AGENCIES

National Technical
Information Service (NTIS)
5285 Port Royal Road
Springfield, Virginia 22161
United States

ESA/Information Retrieval Service
European Space Agency
10, rue Mario Nikis
75015 Paris
France

The British Library
Document Supply Centre
Boston Spa, Wetherby
West Yorkshire LS23 7BQ
United Kingdom

Requests for microfiches or photocopies of AGARD documents (including requests to NTIS) should include the word 'AGARD' and the AGARD serial number (for example AGARD-AG-315). Collateral information such as title and publication date is desirable. Note that AGARD Reports and Advisory Reports should be specified as AGARD-R-nnn and AGARD-AR-nnn, respectively. Full bibliographical references and abstracts of AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
published by NASA Scientific and Technical
Information Division
NASA Headquarters (NTT)
Washington D.C. 20546
United States

Government Reports Announcements and Index (GRA&I)
published by the National Technical Information Service
Springfield
Virginia 22161
United States

(also available online in the NTIS Bibliographic Database or on CD-ROM)

Printed by Specialised Printing Services Limited
40 Chigwell Lane, Loughton, Essex IG10 3TZ

ISBN 92-835-0635-9

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference	3. Further Reference	4. Security Classification of Document
	AGARD-LS-179	ISBN 92-835-0635-9	UNCLASSIFIED

5. Originator Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization
7 rue Ancelle, 92200 Neuilly sur Seine, France

6. Title ARTIFICIAL NEURAL NETWORK APPROACHES IN GUIDANCE AND CONTROL

7. Presented on 8th—9th October 1991 in Monterey, United States, 14th—15th October 1991 in Kjeller (near Oslo) and 17th—18th October 1991 in Neubiberg, Germany.

8. Author(s)/Editor(s)	9. Date
Various	September 1991
10. Author's/Editor's Address	**11. Pages**
Various	180

12. Distribution Statement This document is distributed in accordance with AGARD policies and regulations, which are outlined on the back covers of all AGARD publications.

13. Keywords/Descriptors

Guidance and control
Flight control
Missile guidance
Weapon delivery
Neural nets
Data fusion
Parallel processing

14. Abstract

Ever increasing operational and technical requirements have led to highly integrated flight, guidance and control, and weapons delivery systems. The effective implementation of these functions makes the fusion and interpretation of sensor data and the multifunctional use of sensor information inevitable.

Neural networks, consisting of parallel microcomputing elements, hold great promise for guidance, navigation and control applications because of their ability to learn and acquire knowledge.

The Lecture Series will bring together a group of NATO nation speakers with outstanding experience in this new area of technology. First they will review the fundamentals of neural networks to serve as background so that advances in this new, rapidly evolving technological area can be both understood and appreciated. They will then discuss a number of related applications of direct benefit to the attendees.

This Lecture Series, sponsored by the Guidance and Control Panel of AGARD, has been implemented by the Consultant and Exchange Programme.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

INTRODUCTION

Albert J. Shapiro
GEC-Marconi Electronic Systems Corp.

INTRODUCTION

The objective of this Lecture Series is to present both the fundamentals of neural networks and a number of related Guidance, Navigation and Control (GNC) applications. The lecturers come from several of the participating AGARD countries, specifically Canada, France, Germany, the United Kingdom, and the United States. We will have nine lectures and conclude with a round-table discussion involving all participants.

PERSPECTIVE

The digital computer has impacted GNC from two aspects. It makes possible the implementation of large embedded systems utilizing complex algorithms and control logic. It has also become the primary engineering tool which makes possible the design and analysis of such systems.

Advances in computational speed have enabled the real-time implementation of algorithm-intensive GNC solutions for both aircraft and missiles. Application of Kalman filtering in hybrid-inertial navigation and optimized flight control applications in airframes with complex flexure patterns are examples of practically successful design and hardware/software integration. Kalman filtering allows for less precise sensors to be synergistically integrated through software to provide improved overall system performance.

Airborne missions have become more complex and stressful to the pilot. Scenarios now require threat avoidance, rapid replanning and reconfiguration of navigation modes in the presence of jamming of navigation aids such as GPS, emission management in heavily defended areas, and continuous evaluation of avionics system status in terms of fault detection and isolation and fault tolerant reconfiguration The need for reducing the pilot's workload through relegation of more diagnostic and decision- making functions to the computer has become a necessity.

The application of Artificial Intelligence or Expert Systems to these applications is a significant step in this direction. In conventional problem-solving, deterministic responses are produced for anticipated circumstances but unanticipated situations cannot be properly processed. On the other hand, the Expert System approach has additional information built into its Knowledge Base, approximating the resources of a skilled problem solver. The Inference Engine provides the mechanism to attack the problem with these resources.

However,to quote Dr. Bowen from one of his previous AGARD- sponsored lectures, "An expert system, nonetheless, is quite similar to a real-time control system; for example, both are command and event driven, have feedback loops, require the same instrumentation packages, and access the same kind of data from conventional data bases."

Compare this to the prospect of the machine duplication of functions of the human brain in which, somehow, a natural network of neurons, composed of interconnected living nerve cells, thinks, feels, learns and remembers.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Scientists and engineers are primarily interested in models inspired by brain function and not necessarily the achievement of biological fidelity. The objectives of engineering research in artificial neural networks (ANNs) are to understand how the brain's "computations" are organized and carried out and then to understand the class of neural network models that replicate this "computational power".

The increasing interest in ANNs has been aided by both technological advances as well a deeper understanding of how the brain works. A driving force is the need for a new breed of powerful computers to solve a variety of problems that are proving to be very difficult for conventional digital computers. Cognitive tasks such as pattern recognition under real-world conditions, pattern matching, and combinatorial optimization are some examples. Tasks such as recognizing a familiar face, learning to speak and understand a natural language, and retrieving contextually appropriate information from memory are typically performed naturally by the brain, but are beyond the reach of conventionally programmed computers as well as the rule-based expert systems.

Neurocomputing, that is, nonprogrammmed adaptive information processing systems - artificial neural networks- is a fundamentally new and different information-processing paradigm - the first alternative to algorithmic programming. It holds the potential for significant breakthroughs in the field of GNC - systems which can learn and rapidly accommodate to a wide variety of internal and external stimuli occurring in nonpredetermined combinations. For example, rapid reaction to unforeseen combinations/types of threats and aerodynamic changes, and autonomous vehicles capable of self guidance are but examples of such leaps in capability.

With the foregoing in mind, this Lecture Series has two major themes: a tutorial introduction to ANNs and applications of the overall technology of ANN to the Guidance and Control field.

I hope that these papers will be as informative to you as I am sure they will be to me.

REFERENCES

1. Quinlaven, R. P., "Knowledge-Based Concepts And Artificial Intelligence Applications To Guidance And Control", AGARD Lecture Series No. 155.

2. Bowen, B. A., "Real Time Expert Systems: A Status Report", AGARD Lecture Series No. 155.

3. Vemur, V., "Artificial Neural Networks: Theoretical Concepts", The Computer Society Of The IEEE.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Contents

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Abstract

Ever increasing operational and technical requirements have led to highly integrated flight, guidance and control, and weapons delivery systems. The effective implementation of these functions makes the fusion and interpretation of sensor data and the multifunctional use of sensor information inevitable.

Neural networks, consisting of parallel microcomputing elements, hold great promise for guidance, navigation and control applications because of their ability to learn and acquire knowledge.

The Lecture Series will bring together a group of NATO nation speakers with outstanding experience in this new area of technology. First they will review the fundamentals of neural networks to serve as background so that advances in this new, rapidly evolving technological area can be both understood and appreciated. They will then discuss a number of related applications of direct benefit to the attendees.

This Lecture Series, sponsored by the Guidance and Control Panel of AGARD, has been implemented by the Consultant and Exchange Programme.

Abrégé

Les exigences techniques et opérationnelles toujours plus nombreuses ont amené des systèmes de commandes de vol, de guidage et de pilotage et de lancement d'engins fortement intégrés. La mise en oeuvre effective de ces systèmes passe inévitablement par le fusionnement et le dépouillement des données des capteurs.

Les réseaux neuroniques, qui consistent en des éléments micro-informatiques mise en parallèle, sont très prometteurs pour des applications dans le domaine du guidage, du pilotage et de la navigation, en raison de leur capacité d'apprentissage.

Ce cycle de conférences rassemble un groupe de conférenciers des pays membres de l'OTAN ayant une expérience exceptionnelle dans ce nouveau domaine technologique. Les aspects fondamentaux des réseaux neuroniques seront abordés dans un premier temps pour permettre une estimation des progrès réalisés dans ce domaine en pleine expansion. Un certain nombre d'applications connexes, d'un intérêt particulier pour les participants, seront ensuite discutés.

Ce cycle de conférences est présenté dans le cadre du programme des Consultants et des Echanges, sous l'égide du Panel AGARD du Guidage et du Pilotage.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

List of Authors/Speakers

Lecture Series Director: Dr A.J. Shapiro
Vice President
Electronic and Tactical Systems
GEC Marconi Electronic Systems Corp.
150 Parish Drive
P O Box 932
Wayne, New Jersey 07474-0932
United States

AUTHORS/SPEAKERS

Mr B. Angeniol
MIMETICS
5 Centrale Parc
Avenue Sully Prud'homme
92298 Chatenay Malabry
France

Mr B.A. Bowen
President
Comp Eng Serv Ltd
Suite 600
265 Carling Avenue
Ottawa, Ontario K1S 2E1
Canada

Mr T. Gutschow
HNC Inc.
5501 Oberlin Drive
San Diego, CA 92121-1718
United States

Dipl.-Ing. Uwe Krogmann
Bodenseewerk Geratetechnik GmbH
Intelligent Systems Division
Nussdorfer Str.
7770 Überlingen
Germany

Mr R. Hecht-Nielsen
HNC Inc.
and
Dept. of Electrical & Computer Engineering
University of California, San Diego
La Jolla, CA 92093
United States

Mr P. Simpson
General Dynamics
Electronics Division
PO Box 85310
Mail Zone 7202 K
San Diego, CA 92186-5310
United States

Dr W.A. Wright
British Aerospace
Sowerby Research Centre
FPC 267, P O Box 5
Filton
Bristol BS 127 QW
United Kingdom

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

LS-179

SELECTIVE BIBLIOGRAPHY

This bibliography with abstracts was prepared to support AGARD Lecture Series No. 179 by the Scientific and Technical Information Program of the U.S. National Aeronautics and Space Administration, Washington, D.C., in consultation with the Lecture Series Director, Mr. A. J. Shapiro, GEC-Marconi Electronic Systems Corporation, Wayne, New Jersey.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

UTTL: Optical controller for adaptive phased array antennas using neural network architecture
AUTH: A/DECUSATIS, C.; B/DAS, P. PAA: B/(Rensselaer Polytechnic Institute, Troy, NY) IN: Optoelectronic signal processing for phased-array antennas II; Proceedings of the Meeting, Los Angeles, CA, Jan. 16, 17, 1990 (A91-24926 09-32). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1990, p. 161-172. Research supported by USAF.
ABS: The control of adaptive phased array antennas using the least mean squares (LMS) algorithm is shown to be analogous to the implementation of a two-layer perceptron neural network. The adaptive weights may be calculated using the back propagation algorithm, which is a generalized version of LMS. By using a full perceptron model, additional adaptive weights are introduced at the receiver; this is expected to improve performance over existing systems. An optical processor for the control of adaptive antennas is proposed, based on a two-evel perceptron. It is shown that currently available technology is capable of realizing this receiver; the optical architecture may also be applied to the demands of future wideband interference suppression systems. 90/00/00 91A24942

UTTL: Simulation of heterogeneous neural networks on serial and parallel machines
AUTH: A/LANGE, TRENT E. PAA: A/(California, University, Los Angeles) Parallel Computing (ISSN 0167-8191), vol. 14, Aug. 1990, p. 287-303. Research supported by the W. M. Keck Foundation and ITA Foundation.
ABS: The development tool, DESCARTES is described. This tool provides researchers with the capability to simulate heterogeneous connectionist networks in which the nodes and links may have different processing characteristics and effective cycling rates or which are made up of modular, interacting sub-networks. DESCARTES also makes it possible for researchers to build hybrid networks which combine elements from distributed, localist, and symbolic marker-passing networks. Currently, DESCARTES is implemented on serial machines, where it is able to simulate networks of medium size by utilizing the spreading-activation process to prune unchanging nodes from the update and spreading cycles. Simulation on SIMD (Single Instruction Multiple Data) machines is discussed, focusing on the SIMD simulation cycle, the cycle's update stage, the SIMD cycle's spread-out-to-links stage, and the efficient backpropagation on SIMD machines. Simulation on hypothetical MIMD (Multiple Instruction Multiple Data) machines is also discussed. 90/08/00 91A22124

UTTL: Neural networks and the control of smart systems
AUTH: A/THURSBY, M. H.; B/GROSSMAN, B.; C/YOO, K. PAA: C/(Florida Institute of Technology, Melbourne) IN: U.S.-Japan Workshop on Smart/Intelligent Materials and Systems, Honolulu, HI, Mar. 19-23, 1990, Proceedings (A91-21207 07-23). Lancaster, PA, Technomic Publishing Co., Inc., 1990, p. 242-251. Research supported by the U.S. Army and Florida High Technology and Industry Council.
ABS: Artificial neural networks (ANNs) and their ability to model and control dynamical systems for smart structures, including sensors, actuators, and plants, are considered. Both linear and nonlinear systems have been successfully modeled. Presently, two diverse regimes, smart mechanical systems and smart electromagnetic systems, are being developed. In order to better understand neural controllers as used in the smart electromagnetic structures, the study of ANNs is directed toward understanding the ability of the network to approximate system responses. Networks are being trained to mimic the desired output of the system. The damped sinusoid was chosen as the model and was approximated using a Jordan-like iterative network. The results to date indicate that the ANNs can easily mimic these systems - the question is whether the mechanism that the network applies can be related to the mechanisms for classical analysis. 90/00/00 91A21214

UTTL: Neurocontrol of auto-lock-on target-tracking sight control system
AUTH: A/CHEOK, KA C.; B/SMITH, JAMES C.; C/FERNANDO, JOSEPH P. PAA: C/(Oakland University, Rochester, MI) Control and Computers (ISSN 0315-8934), vol. 17, no. 2, 1989, p. 32-36.
ABS: Neural nets were used to implement the control of an auto-lock-on target-tracking sight/vision control system. The objective of the resultant target-tracking neurocontrol system is to capture and emulate human cognitive action in the eye-hand coordination for tracking a target using a sight system. The paper describes how a tracking neurocontroller was designed and implemented using a microcomputer-based real-time animation simulator. Successful tracking performance of the neurocontrol sight system was achieved in the presence of pseudo-random target maneuvers. 89/00/00 91A19981

UTTL: Electronic neural networks for global optimization
AUTH: A/THAKOOR, A. P.; B/MOOPENN, A. W.; C/EBERHARDT, S. PAA: C/(JPL, Pasadena, CA) CORP: Jet Propulsion Lab., California Inst. of Tech., Pasadena. IN: Intelligent control and adaptive systems; Proceedings of the Meeting, Philadelphia, PA, Nov. 7, 8, 1989 (A91-19635 06-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1990, p. 170-177. Research sponsored by DARPA and SDIO.
ABS: An electronic neural network with feedback architecture, implemented in analog custom VLSI is described. Its application to problems of global optimization for dynamic assignment is discussed. The convergence properties of the neural network hardware are compared with computer simulation results. The neural network's ability to provide optimal or near optimal solutions within only a few neuron time constants, a speed enhancement of several orders of magnitude over conventional search methods, is demonstrated. The effect of noise on the circuit dynamics and the convergence behavior of the neural network hardware is also examined. 90/00/00 91A19642

UTTL: Implementation of expert system/AI technology for reducing ground test in present and future launch systems
AUTH: A/ENGLE, JAMES; B/OWEN, CHARLES; C/COLMENAREZ, LUIS PAA: C/(Rockwell International Corp., Space Systems Div., Downey, CA) AIAA, Aerospace Sciences Meeting, 29th, Reno, NV, Jan. 7-10, 1991. 11 p.
ABS: The application of expert system technology for prelaunch and in-flight health monitoring is considered, and a prelaunch expert system for the Orbiter maneuvering system is outlined. Design requirements and technology concepts for artificial-intelligence/expert-system-based approaches that reduce ground operation costs for a reaction control system on future vehicles are presented. A number of the current AI enabling technologies for reducing ground processing, including expert built-in-test, artificial neural networks, and intelligent machine vision systems are discussed. Attention is concentrated on system integration of AI techniques, engineering-support automation, and intelligent operations paperless systems.
RPT#: AIAA PAPER 91-0655 91/01/00 91A19398

UTTL: Use of Hopfield neural networks in optimal guidance
AUTH: A/STECK, JAMES E.; B/BALAKRISHNAN, S. N. PAA: B/(Missouri-Rolla, University, Rolla) AIAA, Aerospace Sciences Meeting, 29th, Reno, NV, Jan. 7-10, 1991. 6 p.
ABS: A Hopfield neural network architecture for homing missile guidance is considered in this study. A linear quadratic optimal control problem is converted to a Hopfield neural network structure. Several target-intercept scenarios are provided to demonstrate the use of the neural net formulation. Further research directions are recommended.
RPT#: AIAA PAPER 91-0587 91/01/00 91A19372

UTTL: Optical neurochip based on a three-layered feed-forward model
AUTH: A/OHTA, J.; B/KOJIMA, K.; C/NITTA, Y.; D/TAI, S.; E/KYUMA, K. PAA: E/(Mitsubishi Electric Corp., Central Research Laboratory, Amagasaki, Japan) Optics Letters (ISSN 0146-9592), vol. 15, Dec. 1, 1990, p. 1362-1364.
ABS: A GaAs/AlGaAs optical neurochip based on a three-layered feed-forward model is reported. The optical neurochip consists of a light-emitting diode array with 66 elements, a fixed interconnection matrix, and a photodiode array with 110 elements. The interconnection matrix is determined by the backpropagation learning rule with three quantized levels. There are 35, 29, and 26 neurons, respectively, in the input, hidden, and output layers. The excitatory and inhibitory synapses are integrated on one chip. By using the chip and external electronics, the recognition of 10 characters with 5 x 7 bits has been achieved. 90/12/01 91A18667

UTTL: Feedback network with space invariant coupling
AUTH: A/HAEUSLER, GERD; B/LANGE, EBERHARD PAA: B/(Erlangen-Nuernberg, Universitaet, Erlangen, Federal Republic of Germany) Applied Optics (ISSN 0003-6935), vol. 29, Nov. 10, 1990, p. 4798-4805.
ABS: Processing images by a neural network means performing a repeated sequence of operations on the images. The sequence consists of a general linear transformation and a nonlinear mapping of pixel intensities. The general (shift variant) linear transformation is time consuming for large images if done with a serial computer. A shift invariant linear transformation can be implemented much easier by fast Fourier transform or optically, but the shift invariant transform has fewer degrees of freedom because the coupling matrix is Toeplitz. A neural convolution network with shift invariant coupling that nevertheless exhibits autoassociative restoration of distorted images is presented. Besides the simple implementation, the network has one more advantage: associative recall does not depend on object position. 90/11/10 91A17348

UTTL: Neural computation of arithmetic functions
AUTH: A/SIU, KAI-YEUNG; B/BRUCK, JEHOSHUA PAA: A/(Stanford University, CA); B/(IBM Almaden Research Center, San Jose, CA) CORP: Stanford Univ., CA.; IBM Research Lab., San Jose, CA. IEEE, Proceedings (ISSN 0018-9219), vol. 78, Oct. 1990, p. 1669-1675. Research supported by the Joint Services Electronics Program and USAF.
ABS: An area of application of neural networks is considered. A neuron is modeled as a linear threshold gate, and the network architecture considered is the layered feedforward network. It is shown how common arithmetic functions such as multiplication and sorting can be efficiently computed in a shallow neural network. Some known results are improved by showing that the product of two n-bit numbers and sorting of n n-bit numbers can be computed by a polynomial-size neural network using only four and five unit delays, respectively. Moreover, the weights of each threshold element in the neural networks require O(log n)-bit (instead of n-bit) accuracy. These results can be extended to more complicated functions such as multiple products, division, rational functions, and approximation of analytic functions. 90/10/00 91A14887

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

UTTL: Holographic implementation of a fully connected neural network
AUTH: A/HSU, KEN-YUH; B/LI, HSIN-YU; C/PSALTIS, DEMETRI PAA: A/(National Chiao Tung University, Hsinchu, Republic of China); C/(California Institute of Technology, Pasadena)
IEEE, Proceedings (ISSN 0018-9219), vol. 78, Oct. 1990, p. 1637-1645. Research supported by DARPA and USAF.
ABS: A holographic implementation of a fully connected neural network is presented. This model has a simple structure and is relatively easy to implement, and its operating principles and characteristics can be extended to other types of networks, since any architecture can be considered as a fully connected network with some of its connections missing. The basic principles of the fully connected network are reviewed. The optical implementation of the network is presented. Experimental results which demonstrate its ability to recognize stored images are given, and its performance and analysis are discussed based on a proposed model for the system. Special attention is focused on the dynamics of the feedback loop and the tradeoff between distortion tolerance and image-recognition capability of the associative memory.
90/10/00 91A14885

UTTL: Maximum a posteriori decision and evaluation of class probabilities by Boltzmann perceptron classifiers
AUTH: A/YAIR, EYAL; B/GERSHO, ALLEN PAA: A/(IBM Scientific Center, Haifa, Israel); B/(California, University, Santa Barbara)
IEEE, Proceedings (ISSN 0018-9219), vol. 78, Oct. 1990, p. 1620-1628. Research supported by the Weizmann Foundation for Scientific Research, University of California, Bell Communications Research, Inc., et al.
ABS: Neural-network architectures which may offer a valuable alternative to the Bayesian classifier are described. In networks, the a posteriori probabilities are computed with no a priori assumptions about the probability distribution functions that generate the data; the neural classifier uses a general type of input-output mapping which is designed to optimally comply with a given training set. It is shown that the a posteriori class probabilities can be efficiently computed by a deterministic feedforward network which is called the Boltzmann perceptron classifier (BPC). Maximum a posteriori classifiers are also constructed as a special case of the BPC. Structural relationships between the BPC and a conventional multilayer perceptron are given, and it is demonstrated that rather intricate boundaries between classes can be formed even with a relatively modest number of network units. Simulation results show that the BPC is comparable in performance to a Bayesian classifier. 90/10/00 91A14883

UTTL: Nearest neighbor pattern classification perceptrons
AUTH: A/MURPHY, OWEN J. PAA: A/(Vermont, University, Burlington)
IEEE, Proceedings (ISSN 0018-9219), vol. 78, Oct. 1990, p. 1595-1598.
ABS: A three-layer perceptron that uses the nearest-neighbor pattern-classification rule is presented. This neural network is of interest because it is designed specifically for the set of training patterns, and incorporating of the training of the network into the design eliminates the need for the use of training algorithms. The technique therefore provides an alternative to the limitations and unpredictability (such as having too many, too few, or inappropriate training patterns) of the known training techniques. Since the nearest-neighbor classification rule is used, the network is capable of forming arbitrarily complex decision regions. The design and training of the network can be completed in polynomial time, whereas it has been shown that training a neural network is an NP-complete problem. 90/10/00 91A14880

UTTL: Backpropagation through time - What it does and how to do it
AUTH: A/WERBOS, PAUL J. PAA: A/(NSF, Washington, DC)
IEEE, Proceedings (ISSN 0018-9219), vol. 78, Oct. 1990, p. 1550-1560.
ABS: Backpropagation, which is a simple method now being widely used in areas like pattern recognition and fault diagnosis, is reviewed. The basic equations for backpropagation through time, and applications to areas like pattern recognition involving dynamic systems, systems identification, and control, are discussed. Further extensions of this method, to deal with systems other than neural networks, systems involving simultaneous equations, or true recurrent networks, and other practical issues arising with the method are described. Pseudocode is provided to clarify the algorithms. The chain rule for ordered derivatives (the theorem which underlies backpropagation) is briefly discussed. The focus is on designing a simpler version of backpropagation which can be translated into computer code and applied directly by neural-network users. 90/10/00 91A14874

UTTL: 30 years of adaptive neural networks - Perceptron, Madaline, and backpropagation
AUTH: A/WIDROW, BERNARD; B/LEHR, MICHAEL A. PAA: B/(Stanford University, CA) CORP: Stanford Univ., CA.
IEEE, Proceedings (ISSN 0018-9219), vol. 78, Sept. 1990, p. 1415-1442. Research sponsored by SDIO and Lockheed Missiles and Space Co., Inc.
ABS: Fundamental developments in feedforward artificial neural networks from the past thirty years are reviewed. The history, origination, operating characteristics, and basic theory of several supervised neural-network training algorithms (including the perceptron rule, the least-mean-square algorithm, three Madaline rules, and the backpropagation technique) are described. The concept underlying these iterative adaptation algorithms is the minimal disturbance principle, which suggests that during training it is advisable to inject new information into a network in a manner that disturbs stored information to the smallest extent possible. The two principal kinds of online rules that have developed for altering the weights of a network are examined for both single-threshold elements and multielement networks. They are error-correction rules, which alter the weights of a network to correct error in the output response to the present input pattern, and gradient rules, which alter the weights of a network during each pattern presentation by gradient descent with the objective of reducing mean-square error (averaged over all training patterns).
90/09/00 91A14870

UTTL: Expert systems and advanced automation for space missions operations
AUTH: A/DURRANI, SAJJAD H.; B/PERKINS, DOROTHY C.; C/CARLTON, P. DOUGLAS PAA: A/(NASA, Office of Space Operations, Washington, DC); B/(NASA, Goddard Space Flight Center, Greenbelt, MD); C/(Computer Sciences Corp., Laurel, MD) CORP: National Aeronautics and Space Administration, Washington, DC.; National Aeronautics and Space Administration, Goddard Space Flight Center, Greenbelt, MD.; Computer Sciences Corp., Laurel, MD.
IAF, International Astronautical Congress, 41st, Dresden, Federal Republic of Germany, Oct. 6-12, 1990. 8 p.
ABS: Increased complexity of space missions during the 1980s led to the introduction of expert systems and advanced automation techniques in mission operations. This paper describes several technologies in operational use or under development at the National Aeronautics and Space Administration's Goddard Space Flight Center. Several expert systems are described that diagnose faults, analyze spacecraft operations and onboard subsystem performance (in conjunction with neural networks), and perform data quality and data accounting functions. The design of customized user interfaces is discussed, with examples of their application to space missions. Displays, which allow mission operators to see the spacecraft position, orientation, and configuration under a variety of operating conditions, are described. Automated systems for scheduling are discussed, and a testbed that allows tests and demonstrations of the associated architectures, interface protocols, and operations concepts is described. Lessons learned are summarized.
RPT#: IAF PAPER 90-405 90/10/00 91A14013

UTTL: Identification of aerospace acoustic sources using sparse distributed associative memory
AUTH: A/SCOTT, E. A.; B/FULLER, C. R.; C/O'BRIEN, W. F. PAA: C/(Virginia Polytechnic Institute and State University, Blacksburg) CORP: Virginia Polytechnic Inst. and State Univ., Blacksburg.
AIAA, Aeroacoustics Conference, 13th, Tallahassee, FL, Oct. 22-24, 1990. 12 p.
ABS: A pattern recognition system has been developed to classify five different aerospace acoustic sources. In this paper the performance of two new classifiers, an associative memory classifier and a neural network classifier, is compared to the performance of a previously designed system. Sources are classified using features calculated from the time and frequency domain. Each classifier undergoes a training period where it learns to classify sources correctly based on a set of known sources. After training the classifier is tested with unknown sources. Results show that over 96 percent of sources were identified correctly with the new associative memory classifier. The neural network classifier identified over 81 percent of the sources correctly.
RPT#: AIAA PAPER 90-3992 90/10/00 91A12505

UTTL: Modified backpropagation algorithm for fast learning in neural networks
AUTH: A/REYNERI, L. M.; B/FILIPPI, E. PAA: B/(Torino, Politecnico, Turin, Italy)
Electronics Letters (ISSN 0013-5194), vol. 26, Sept. 13, 1990, p. 1564-1566.
ABS: A fast learning rule for artificial neural systems which is based on modifications to a backpropagation algorithm is described. The rule minimizes the error function along the direction of the gradient and backpropagates the error pattern according to a constant error energy approach.
90/09/13 91A12410

UTTL: Application of adjoint operators to neural learning
AUTH: A/BARHEN, J.; B/TOOMARIAN, N.; C/GULATI, S. PAA: A/(JPL; California Institute of Technology, Pasadena); C/(JPL, Pasadena, CA) CORP: Jet Propulsion Lab., California Inst. of Tech., Pasadena.; California Inst. of Tech., Pasadena.
Applied Mathematics Letters (ISSN 0893-9659), vol. 3, no. 3, 1990, p. 13-18. Research supported by DOD and DOE.
ABS: A technique for the efficient analytical computation of such parameters of the neural architecture as synaptic weights and neural gain is presented as a single solution of a set of adjoint equations. The learning model discussed concentrates on the adiabatic approximation only. A problem of interest is represented by a system of N coupled equations, and then adjoint operators are introduced. A neural network is formalized as an adaptive dynamical system whose temporal evolution is governed by a set of coupled nonlinear differential equations. An approach based on the minimization of a constrained neuromorphic energylike function is applied, and the complete learning dynamics are obtained as a result of the calculations. 90/00/00 90A50026

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

UTTL: Integration of parallel image processing with symbolic and neural computations for imagery exploitation
AUTH: A/ROMAN, EVELYN PAA: A/(Optical Systems and Equipment, Lexington, MA) IN: Airborne reconnaissance XIII; Proceedings of the Meeting, San Diego, CA, Aug. 7-9, 1989 (A90-48601 22-06). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1989, p. 72-83.
ABS: Work combining parallel, symbolic, and neural methodologies at different stages of processing for imagery exploitation are discussed, together with a prototype system combining real-time parallel image processing on an 8-stage parallel image-processing engine (PIPE) computer with expert system software. A summary of basic neural concepts is given, and the commonality between neural nets and related mathematics, artificial intelligence, and traditional image processing concepts is shown. This provides numerous choices for the implementation of constraint satisfaction, transformational invariance, inference and representational mechanisms, and software lifecycle engineering methodologies in the different computational layers. 89/00/00 90A48609

UTTL: Neural net classifier for millimeter wave radar
AUTH: A/BROWN, JOE R.; B/ARCHER, SUE; C/BOWER, MARK R. PAA: C/(Martin Marietta Electronic Systems, Orlando, FL) IN: Real-time signal processing XII; Proceedings of the Meeting, San Diego, CA, Aug. 10, 11, 1989 (A90-48408 22-32). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1989, p. 71-76.
ABS: This paper describes the development of a neural net classifier for use in an automatic target recognition (ATR) system using millimeter wave (MMW) radar data. Two distinctive neural net classifiers were developed using mapping models (backpropagation and counterpropagation) and compared to a quadratic (Bayesian-like) classifier. A statistical feature set and a radar data set was used for both training and testing all three classifier systems. This statistical feature set is often used to test MMW ARTs prior to using actual data. Results are presented and indicate that the backpropagation net performed at near 100 percent accuracy for the statistical feature set and slightly outperformed the counterpropagation model in this application. Both networks hold promising results using real radar data. 89/00/00 90A48413

UTTL: Application of neural networks to automatic control
AUTH: A/GOLDENTHAL, WILLIAM; B/FARRELL, JAY PAA: B/(Charles Stark Draper Laboratory, Inc., Cambridge, MA) IN: AIAA Guidance, Navigation and Control Conference, Portland, OR, Aug. 20-22, 1990, Technical Papers. Part 2 (A90-47576 21-08). Washington, DC, American Institute of Aeronautics and Astronautics, 1990, p. 1108-1112.
ABS: The design of a robust control system for vehicles with highly nonlinear, time-varying, or poorly-modeled dynamics poses serious difficulties for all currently advocated design methodologies. These difficulties arise in the design of current aerospace and underwater vehicles and are crucial for proposed autonomous vehicles. In the present paper the use of neural networks in adaptive control loops is proposed, based on the fact that feedforward neural networks with at least one hidden layer have been shown to be dense (under suitable assumptions) on the set of continuous functions. Thus, by the use of a suitable adaptive learning algorithm, the interconnection weights of the network could be selected so that the network approximates the desired nonlinear control law to any specified accuracy. An extension of the backpropagation algorithm is presented which adaptively determines the interconnection parameters necessary for the neural network to function as a closed-loop controller and to force the closed-loop system to match a desired reference response. An example of the application of this algorithm to the control of the cart pole system is included.
RPT#: AIAA PAPER 90-3438 90/00/00 90A47691

UTTL: Advanced architecture for domestic and global aviation systems
AUTH: A/KORGEL, CLAYTON C. PAA: A/(Martin Marietta Information Systems Group, Bethesda, MD) IN: Radio Technical Commission for Aeronautics, Annual Assembly and Technical Symposium, Washington, DC, Dec. 4-6, 1989, Proceedings (A90-46390 21-04). Washington, DC, Radio Technical Commission for Aeronautics, 1989, p. 197-209.
ABS: Candidate elements for the future aviation systems are outlined, and top-down as well as bottom-up system architecture approaches are examined, and it is noted that automation and human factors will dominate the system including airspace and flight management subsystems. Communications systems, surveillance, and navigation and landing are discussed. Since the systems under consideration include possible synergisms, redundancies, and buck-up capabilities, possible options and trade-offs are analyzed. Key technologies for future aviation systems such as the GPS/GLONASS integrated receiving set, real-time expert system/neural networks, antenna avionics, interactive speech and display processing, satellite communication equipment, and microwave monolithic integrated circuits are presented. 89/00/00 90A46398

UTTL: Neural network systems
AUTH: A/GUYON, ISABELLE PAA: A/(AT&T Bell Laboratories, Holmdel, NJ) IN: International Symposium on Numerical Methods in Engineering, 5th, Lausanne, Switzerland, Sept. 11-15, 1989, Proceedings. Volume 1 (A90-44401 20-31). Southampton, England and New York/Berlin, Computational Mechanics Publications/Springer-Verlag, 1989, p. 203-210.
ABS: In the last few years, devices inspired by the architecture of the brain have become much more powerful. A lot of effort has been concentrated on networks using very rough models of neuron cells (formal neurons). The ability of such systems to learn from examples is a particularly attractive feature. Research is still in its infancy, but it is expected that these models will be useful both as models of real brain function and as computational devices for many applications including optimization, pattern recognition, speech analysis, and signal processing. Architectures and the associated learning algorithms that have been proposed are reviewed. The notion of generalization from the training examples are explained, and various examples of problems and applications of practical interest that can be handled by neural networks are presented. 89/00/00 90A44412

UTTL: Neural networks for automatic target recognition
AUTH: A/ROTH, MICHAEL W. PAA: A/(Johns Hopkins University, Laurel, MD) Johns Hopkins APL Technical Digest (ISSN 0270-5214), vol. 11, Jan.-June 1990, p. 117-120. Research supported by the Johns Hopkins University.
ABS: The use of neural networks and neurocomputers is discussed and their applications for automatic target recognition (ATR) are reviewed. A framework is presented illustrating the application of neural network technology to the solution of the ATR problem of recognizing high-value targets in noisy environments and discriminating them from low-value objects and false alarms. Neural network tools which may be applied to ATR needs include collective computation for fast optimalization, neural network learning algorithms, neural network inspired feature selection, and a neural network for higher vision. An example of a binocular stereo displacement map produced using model images and preliminary stereo calculations on the Connection Machine at the Naval Research Laboratory is presented and discussed. It is pointed out that neural learning could facilitate the development of both automatic knowledge acquisition and continuous system refinement, two important ATR advances. 90/06/00 90A44324

UTTL: New directions in missile guidance - Signal processing based on neural networks and fractal modeling
AUTH: A/BOONE, BRADLEY G.; B/CONSTANTIKES, KIM T.; C/FRY, ROBERT L.; D/GILBERT, ALLEN S.; E/KULP, ROBERT L. PAA: E/(Johns Hopkins University, Laurel, MD) Johns Hopkins APL Technical Digest (ISSN 0270-5214), vol. 11, Jan.-June 1990, p. 28-38.
ABS: Projects investigating the utility of signal processing based on neural networks and fractal scene modeling are discussed. New approaches to target recognition and scene-matching development are examined with attention to the performance and characteristics of image-based scene matchers. A discussion on new models and representations for missile guidance includes an investigation of neural network learning models emphasizing the training phase and the various alternatives to target representation. An investigation of the recognition of range-profile ship signatures using a back-propagation neural net with comparisons to baseline statistical classifiers is described. Prospects for future work are discussed including innovative approaches to target acquisition. 90/06/00 90A44318

UTTL: Neural networks for control and system identification
AUTH: A/WERBOS, PAUL J. PAA: A/(NSF, Washington, DC) IN: IEEE Conference on Decision and Control, 28th, Tampa, FL, Dec. 13-15, 1989, Proceedings. Volume 1 (A90-40776 18-63). New York, Institute of Electrical and Electronics Engineers, 1989, p. 260-265.
ABS: A review is presented of the field of neuroengineering as a whole, highlighting the importance of neurocontrol and neuroidentification. Then a description is given of the five major architectures in use today in neurocontrol (in robotics, in particular) and a few areas for future research. Also included are comments on neuroidentification. 89/00/00 90A40788

UTTL: Obscured object recognition for an ATR application
AUTH: A/EICHMANN, G.; B/JANKOWSKI, M.; C/BASU, S.; D/STOJANCIC, M.; E/ROYTMAN, L. PAA: E/(City College, New York) IN: Advances in image compression and automatic target recognition; Proceedings of the Meeting, Orlando, FL, Mar. 30, 31, 1989 (A90-39951 17-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1989, p. 66-73.
ABS: A common and mainly unsolved problem in image processing is occlusion. Occlusion occurs when one or more objects obstruct the sensor's view. In this paper, three methods: a neural network, a superresolving non-parametric predictor, and an Extended-Post Context-free Grammar syntactic pattern recognizer are used to generate the missing data. To illustrate these methods, their application to the reconstruction of obscured Roman characters are presented. 89/00/00 90A39958

UTTL: The elements of adaptive neural expert systems
AUTH: A/HEALY, MICHAEL J. PAA: A/(Boeing Computer Services, Seattle, WA) IN: Applications of artificial intelligence VII; Proceedings of the Meeting, Orlando, FL, Mar. 28-30, 1989. Part 2 (A90-38876 17-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1989, p. 830-837.
ABS: The generalization properties of a class of neural architectures can be modeled mathematically. The model is

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

a parallel predicate calculus based on pattern recognition and self-organization of long-term memory in a neural network. It may provide the basis for adaptive expert systems capable of inductive learning and rapid processing in a highly complex and changing environment. 89/00/00 90A38903

UTTL: Neural networks for self-learning control systems
AUTH: A/NGUYEN, DERRICK H.; B/WIDROW, BERNARD PAA: B/(Stanford University, CA) CORP: Stanford Univ., CA.
IEEE Control Systems Magazine (ISSN 0272-1708), vol. 10, April 1990, p. 18-23. Research supported by SDIO, USAF, Thomson-CSF, and Lockheed Missiles and Space Co., Inc.
ABS: It is shown how a neural network can learn of its own accord to control a nonlinear dynamic system. An emulator, a multilayered neural network, learns to identify the system's dynamic characteristics. The controller, another multilayered neural network, next learns to control the emulator. The self-trained controller is then used to control the actual dynamic system. The learning process continues as the emulator and controller improve and track the physical process. An example is given to illustrate these ideas. The 'truck backer-upper,' a neural network controller that steers a trailer truck while the truck is backing up to a loading dock, is demonstrated. The controller is able to guide the truck to the dock from almost any initial position. The technique explored should be applicable to a wide variety of nonlinear control problems. 90/04/00 90A37571

UTTL: Survey of neural network technology for automatic target recognition
AUTH: A/ROTH, MICHAEL W. PAA: A/(Johns Hopkins University, Laurel, MD) IEEE Transactions on Neural Networks (ISSN 1045-9227), vol. 1, March 1990, p. 28-43.
ABS: A review is presented of ATR (automatic target recognition), and some of the highlights of neural network technology developments that have the potential for making a significant impact on ATR are discussed. In particular, neural network technology developments in the areas of collective computation, learning algorithms, expert systems, and neurocomputer hardware could provide crucial tools for developing improved algorithms and computational hardware for ATR. The discussion covers previous ATR system efforts, ATR issues and needs, early vision and collective computation, learning and adaptation for ATR, feature extraction, higher vision and expert systems, and neurocomputer hardware. 90/03/00 90A34467

UTTL: Identification and control of dynamical systems using neural networks
AUTH: A/NARENDRA, KUMPATI S.; B/PARTHASARATHY, KANNAN PAA: B/(Yale University, New Haven, CT) IEEE Transactions on Neural Networks (ISSN 1045-9227), vol. 1, March 1990, p. 4-27. Research supported by Sandia National Laboratories.
ABS: It is demonstrated that neural networks can be used effectively for the identification and control of nonlinear dynamical systems. The emphasis is on models for both identification and control. Static and dynamic back-propagation methods for the adjustment of parameters are discussed. In the models that are introduced, multilayer and recurrent networks are interconnected in novel configurations, and hence there is a real need to study them in a unified fashion. Simulation results reveal that the identification and adaptive control schemes suggested are practically feasible. Basic concepts and definitions are introduced throughout, and theoretical questions that have to be addressed are also described. 90/03/00 90A34466

UTTL: Comparison of model based vision, statistical based, and neural net based ATRs
AUTH: A/THEIS, TIMOTHY J.; B/AKERMAN, ALEXANDER, III PAA: B/(I-MATH Associates, Inc., Orlando, FL) IN: NAECON 89; Proceedings of the IEEE National Aerospace and Electronics Conference, Dayton, OH, May 22-26, 1989. Volume 4 (A90-30676 12-01). New York, Institute of Electrical and Electronics Engineers, Inc., 1989, p. 1733-1738.
ABS: An effort is made to establish a common ground upon which a comparison of model-based vision (MBV), statistical-based, and neural-net-based (NN) automatic target recognizer (ATR) approaches can be performed. A definition for each type of ATR as compared to a generic ATR is provided. Upon these definitions, the differences, purported risks, and benefits are described. It is found that the comparison between statistical, MBV, and NN approaches to ATR can only be made at a very high system level. The differences primarily deal with how the desired target is represented within the ATR. These representation differences lead to other implementation differences, which affect the performance flexibility and technical achievability of each approach as it is faced with the realities of new target types and engagement conditions. It is noted that as attempts are made to become more specific, there are always attempts to indicate that a particular technique does not belong exclusively to one class of recognizers versus another. Indeed, a hybrid approach of using models to train a statistical-based classifier is valid, but not clearly separable into one class of recognizers. 89/00/00 90A30788

UTTL: Intelligent Mission Adaptive Controller (IMAC)
AUTH: A/GEIGER, KEVIN; B/EDSON, BRUCE; C/MCCORD, JIM PAA: C/(USAF, Avionics Laboratory, Wright-Patterson AFB, OH) IN: NAECON 89; Proceedings of the IEEE National Aerospace and Electronics Conference, Dayton, OH, May 22-26, 1989. Volume 3 (A90-30676 12-01). New York, Institute of Electrical and Electronics Engineers, Inc., 1989, p. 1185-1192.
ABS: The Intelligent Mission-Adaptive Controller (IMAC) research program is investigating distributed-AI (DAI) and adaptive neural system (ANS) technologies for application in active electronic-countermeasure (ECM) resource management. The threat environment for tactical and strategic aircraft requires the ECM system to handle numerous fast-reacting, sometimes agile systems which vary in function from acquisition to weapons guidance. IMAC is an attempt to capture and demonstrate the important concepts of an ECM resource manager. It deals with the tradeoffs between ECM effectiveness, system costs, and near- and far-term survivability. Preliminary results show that a spreadsheet format for acquiring threat/threat-response information is superior to decision-tree and fuzzy-cognitive-map formats. Capturing complex correlations is found to be the key problem for which a good knowledge-representation scheme is essential. 89/00/00 90A30765

UTTL: An application of neural net technology to surveillance information correlation and battle outcome prediction
AUTH: A/MALONEY, P. SUSIE PAA: A/(Lockheed Missiles and Space Co., Inc., Austin, TX) IN: NAECON 89; Proceedings of the IEEE National Aerospace and Electronics Conference, Dayton, OH, May 22-26, 1989. Volume 2 (A90-30676 12-01). New York, Institute of Electrical and Electronics Engineers, Inc., 1989, p. 948-955. Research supported by the Lockheed Missiles and Space Co., Inc.
ABS: The PNN (probabilistic neural network) is a three-layer feed-forward network that uses sums of Gaussian distibutions to estimate the pdf for a training data set. This trained network can then be used to classify new data sets and to provide a probability associated with each classification. The PNN has been applied successfully to two separate ELINT emitter correlation problems (hull-to-emitter and land-based emitter correlation). Each of these applications achieved a high degree of accuracy in identifying the correct emitter among many possible emitters, at an extremely fast rate (about 200,000 times faster than a standard back-propagation neural network). PNN also shows great potential for solving other surveillance-analysis problems: an application to a battle-outcome prediction problem is described. 89/00/00 90A30749

UTTL: The Adaptive Network Cognitive Processor
AUTH: A/EDSON, BRUCE; B/TURNER, CHERYL; C/MYERS, MICHAEL; D/SIMPSON, PAT PAA: A/(USAF, Avionics Laboratory, Wright-Patterson AFB, OH); C/(TRW MEAD AI Center, San Diego, CA); D/(VERAC, Inc., San Diego, CA) IN: AAAIC '88 - Aerospace Applications of Artificial Intelligence; Proceedings of the Fourth Annual Conference, Dayton, OH, Oct. 25-27, 1988. Volume 1 (A90-30226 12-59). Xenia, OH, Dayton SIGART, 1988, p. 133-143.
ABS: The Adaptive Network Cognitive Processor (ANCP) project is an experiment in the use of adaptive network systems to capture the cognitive processes used for deploying electronic countermeasures by a fighter aircraft in an electronic warfare threat environment. A functional architecture was developed and initially implemented using the Mark III neurocomputer. The main capabilities of the ANCP demonstrated were: internal modeling of the threat environment (Field Interaction Net), adaptive flight route planning (Gradient Descent), reflexive threat response (Feed Forward Net) augmented with a reflective or expert threat response (Fuzzy Cognitive Map) in unfamiliar situations (Confidence Filter), on-board recording of unfamiliar situation/expert response for later retraining (Back Error Propagation) as a reflexive response, and initial training with a Learning Apprentice. 88/00/00 90A30231

UTTL: Optoelectronic implementations of neural networks
AUTH: A/PSALTIS, DEMETRI; B/YAMAMURA, ALAN A.; C/LIN, STEVEN; D/GU, XIANG-GUANG; E/HSU, KEN PAA: D/(California Institute of Technology, Pasadena); E/(National Chiao Tung University, Hsinchu, Republic of China) IEEE Communications Magazine (ISSN 0163-6804), vol. 27, Nov. 1989, p. 37-40, 71. Research supported by DARPA, USAF, and U.S. Army.
ABS: The ability of optical systems to provide the massive interconnections between processors required in most neural network models, which constitutes their chief advantage for such applications, is discussed, focusing on holography. Because of the essential nonlinearity of the holographic connections, nonlinear processing elements are needed to perform complex computations. The use of GaAs hybrid optoelectronic processing elements is examined. GaAs is an excellent material for this purpose, since it can be used to fabricate both fast electronic circuits and optical sources and detectors. It is shown how a complete hybrid neural computer can be implemented using available technology developed for conventional computing. An experimentally demonstrated network in which optics plays an even larger role is described.
RPT#: AD-A217133 89/11/00 90A22506

UTTL: Information theory, complexity, and neural networks
AUTH: A/ABU-MOSTAFA, YASER S. PAA: A/(California Institute of Technology, Pasadena) IEEE Communications Magazine (ISSN 0163-6804), vol. 27, Nov. 1989, p. 25-28, 81.
ABS: Some of the main results in the mathematical evaluation of neural networks as information processing systems are discussed. The basic operation of feedback and feed-forward neural networks is described. Their memory

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

capacity and computing power are considered. The concept of learning by example as it applies to neural networks is examined. 89/11/00 90A22504

UTTL: Automatic target recognition on the connection machine
AUTH: A/BUCHANAN, J. ROBERT PAA: A/(Johns Hopkins University, Laurel, MD) Johns Hopkins APL Technical Digest (ISSN 0270-5214), vol. 10, July-Sept. 1989, p. 208-215.
ABS: Automatic target recognition (ATR) is a computationally intensive problem that benefits from the abilities of the Connection Machine (CM), a massively parallel computer used for data-level parallel computing. The large computational resources of the CM can efficiently handle an approach to ATR that uses parallel stereo-matching and neural-network algorithms. Such an approach shows promise as an ATR system of satisfactory performance. 89/09/00 90A11938

UTTL: Applications of neural networks to avionics systems
AUTH: A/SEIDMAN, ABRAHAM N. PAA: A/(Northrop Corp., Aircraft Div., Hawthorne, CA) AIAA Computers in Aerospace Conference, 7th, Monterey, CA, Oct. 3-5, 1989. 11 p.
ABS: The application of neural networks is discussed as a method of solution to a number of outstanding problems in aircraft avionics. The areas of application of artificial neural networks to avionics dealt with are (1) target selection and (2) attack planning/steering. The target selection is approached by the application of a feed-forward, backpropagation network. The attack planning/steering is approached by a new type of parallel processing neural network.
RPT#: AIAA PAPER 89-3093 89/10/00 90A10627

UTTL: A comparison of CMAC neural network and traditional adaptive control systems
AUTH: A/KRAFT, L. GORDON; B/CAMPAGNA, DAVID P. PAA: B/(New Hampshire, University, Durham, NC) IN: 1989 American Control Conference, 8th, Pittsburgh, PA, June 21-23, 1989, Proceedings. Volume 1 (A89-53951 24-63). New York, Institute of Electrical and Electronics Engineers, 1989, p. 884-889.
ABS: A neural-network-based controller similar to the cerebellar model arithmetic computer (CMAC) method of Miller et al. (1987) is compared to a self-tuning regulator and a Lyapunov-based model reference controller. The three control algorithms are tested on exactly the same control problems. Results are obtained when the system being controlled is linear and noise-free when noise is added to the measurements, and when a nonlinear system is controlled. Comparisons made with respect to closed-loop system stability, speed of adaptation, noise rejection, robustness, the number of required calculations, and system tracking performance indicate that the neural-network approach exhibits the potential for solving some of the problems that have plagued more traditional adaptive control systems. 89/00/00 89A53996

UTTL: Adaptive pattern recognition and neural networks
AUTH: A/PAO, YOH-HAN PAA: A/(Case Western Reserve University, Cleveland, OH) Reading, MA, Addison-Wesley Publishing Co., Inc., 1989, 327 p.
ABS: The application of neural-network computers to pattern-recognition tasks is discussed in an introduction for advanced students. Chapters are devoted to the nature of the pattern-recognition task, the Bayesian approach to the estimation of class membership, the fuzzy-set approach, patterns with nonnumeric feature values, learning discriminants and the generalized perceptron, recognition and recall on the basis of partial cues, associative memories, self-organizing nets, the functional-link net, fuzzy logic in the linking of symbolic and subsymbolic processing, and adaptive pattern recognition and its applications. Also included are C-language programs for (1) a generalized delta-rule net for supervised learning and (2) unsupervised learning based on the discovery of clustered structure. 89/00/00 89A51326

UTTL: Microwave diversity imaging and automated target identification based on models of neural networks
AUTH: A/FARHAT, NABIL H. PAA: A/(Pennsylvania, University, Philadelphia) IEEE, Proceedings (ISSN 0018-9219), vol. 77, May 1989, p. 670-681. Research supported by DARPA, USAF, U.S. Army, and NSF.
ABS: It is shown that collective nonlinear signal processing based on models of neural networks combined with the use of suitable target signatures, offers the promise of robust superresolved target identification from partial information. Results are presented of numerical simulations using a neuromorphic processor, where the neural net performs simultaneously the functions of data storage, processing and recognition. The results demonstrate correct identification from as low as 10 percent of a full sinogram representations derived from real data collected in an anechoic chamber environment for three test targets (scale models of B-52, AWAC, and Space Shuttle) and taught to the network. Practical considerations and extensions to real systems are briefly discussed. 89/05/00 89A45106

UTTL: A unified systolic architecture for artificial neural networks
AUTH: A/KUNG, S. Y.; B/HWANG, J. N. PAA: A/(Princeton University, NJ); B/(Southern California, University, Los Angeles, CA) Journal of Parallel and Distributed Computing (ISSN 0743-7315), vol. 6, April 1989, p. 358-387. Research supported by SDIO.
ABS: A programmable ring systolic array is presently developed on the basis of a generic iterative model encompassing artificial neural networks, single-layer feedback networks, multilayer feedforward networks, hierarchical competitive networks, and even some probabilistic models. The architecture thus obtained maximizes VLSI's advantages in terms of intensive and pipelined computing, while circumventing the conventional limitation on communication; it is therefore recommended as a promising structural basis for a universal neurocomputer architecture. 89/04/00 89A41735

UTTL: Back propagation fails to separate where perceptrons succeed
AUTH: A/BRADY, MARTIN L.; B/RAGHAVAN, RAGHU; C/SLAWNY, JOSEPH PAA: B/(Lockheed Corp., Palo Alto, CA); C/(Virginia Polytechnic Institute and State University, Blacksburg) IEEE Transactions on Circuits and Systems (ISSN 0098-4094), vol. 36, May 1989, p. 665-674. Research supported by Lockheed Corp.
ABS: It is widely believed that the back propagation algorithm in neural networks, for tasks such as pattern classification, overcomes the limitations of the perceptron. The authors construct several counterexamples to this belief. They also construct linearly separable examples which have a unique minimum which fails to separate two families of vectors, and a simple example with four two-dimensional vectors in a single-layer network showing local minima with a large basin of attraction. Thus, back propagation is guaranteed to fail in the first example, and likely to fail in the second example. It is shown that even multilayered (hidden-layer) networks can also fail in this way to classify linearly separable problems. Since the authors' examples are all linearly separable, the perceptron would correctly classify them. The results disprove the presumption, made in recent years, that, barring local minima, back propagation will find the best set of weights for a given problem. 89/05/00 89A41634

UTTL: Multitarget tracking with cubic energy optical neural nets
AUTH: A/BARNARD, ETIENNE; B/CASASENT, DAVID P. PAA: B/(Carnegie-Mellon University, Pittsburgh, PA) Applied Optics (ISSN 0003-6935), vol. 28, Feb. 15, 1989, p. 791-798. Research supported by SDIO.
ABS: A neural net processor and its optical realization are described for a multitarget tracking application. A cubic energy function results and a new optical neural processor is required. Initial simulation data are presented. 89/02/15 89A32825

UTTL: Supervised learning of probability distributions by neural networks
AUTH: A/BAUM, ERIC B.; B/WILCZEK, FRANK PAA: A/(California Institute of Technology, Jet Propulsion Laboratory, Pasadena); B/(Harvard University, Cambridge, MA) CORP: Jet Propulsion Lab., California Inst. of Tech., Pasadena.; Harvard Univ., Cambridge, MA. IN: Neural information processing systems; Proceedings of the First IEEE Conference, Denver, CO, Nov. 8-12, 1987 (A89-29002 11-63). New York, American Institute of Physics, 1988, p. 52-61. Research supported by DARPA.
ABS: Supervised learning algorithms for feedforward neural networks are investigated analytically. The back-propagation algorithm described by Werbos (1974), Parker (1985), and Rumelhart et al. (1986) is generalized by redefining the values of the input and output neurons as probabilities. The synaptic weights are then varied to follow gradients in the logarithm of likelihood rather than in the error. This modification is shown to provide a more rigorous theoretical basis for the algorithm and to permit more accurate predictions. A typical application involving a medical-diagnosis expert system is discussed. 88/00/00 89A29008

UTTL: Spaceplanes astronaut's associate control server
AUTH: A/HONG, ROBERT PAA: A/(Grumman Aerospace Corp., Grumman Aircraft Systems Div., Bethpage, NY) IN: IEEE Conference on Decision and Control, 27th, Austin, TX, Dec. 7-9, 1988, Proceedings. Volume 1 (A89-28491 11-63). New York, Institute of Electrical and Electronics Engineers, Inc., 1988, p. 149-154.
ABS: The author addresses the extension of the DARPA/US Air Force Pilot's Associate program to the astronaut's associate application, and particularly the control server aspect. Some representative techniques for implementing this system are discussed. Artificial intelligence (AI) and neural networks are applied synergistically to achieve an optimum system. The author examines such issues as adaptive aiding, performance seeking control, qualitative reasoning, neural networks gradient methods for connectionist networks, and neural machinery for spacecraft control. 88/00/00 89A28506

UTTL: Autonomous reconfiguration of sensor systems using neural nets
AUTH: A/JAKUBOWICZ, OLEG G. PAA: A/(New York, State University, Buffalo) IN: Sensor fusion; Proceedings of the Meeting, Orlando, FL, Apr. 4-6, 1988 (A89-26951 10-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1988, p. 197-203.
ABS: The application of neural networks to autonomous agents (intelligent robots operating in isolated locations) is

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

discussed, and architectures for implementing self-repairing sensor and identification systems aboard autonomous agents are proposed. The example of a four-layer visual system which identifies visual objects is considered in which each processor connection is assigned a weight attribute. It is shown that when one of the units becomes inoperative, neighboring detectors in that layer may be used to reprogram the weights connecting surviving units in order to restore functionality. 88/00/00 89A26975

UTTL: Multisensor integration and fusion - Issues and approaches
AUTH: A/LUO, REN C.; B/KAY, MICHAEL G. PAA: B/(North Carolina State University, Raleigh) IN: Sensor fusion; Proceedings of the Meeting, Orlando, FL, Apr. 4-6, 1988 (A89-26951 10-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1988, p. 42-49.
ABS: Issues concerning the effective integration of multiple sensors into the operation of intelligent systems are presented, and a description of some of the general paradigms and methodologies that address this problem is given. Multisensor integration, and the related notion of multisensor fusion, are defined and distinguished. The potential advantages and problems resulting from the integration of information from multiple sensors are discussed. 88/00/00 89A26957

UTTL: Sensor fusion; Proceedings of the Meeting, Orlando, FL, Apr. 4-6, 1988
AUTH: A/WEAVER, CHARLES B. PAA: A/(Honeywell, Inc., Electro-Optics Div., Lexington, MA) Meeting sponsored by SPIE. Bellingham, WA, Society of Photo-Optical Instrumentation Engineers (SPIE Proceedings, Volume 931), 1988, 218 p. For individual items see A89-26952 to A89-26975.
ABS: Papers are presented on multisensor target detection and classification, a geometric approach to multisensor fusion, and optimal and suboptimal distributed decision fusion. Also considered are information fusion methodology, theoretical approaches to data association and fusion, and adaptive control of multisensor systems. Other topics include target acquisition and tracking in the laser docking sensor, a neural network architecture for evidence combination, an algorithm for sensor fusion, and the application of order statistic filters to detection systems.
RPT#: SPIE-931 88/00/00 89A26951

UTTL: PSRI target recognition in range imagery using neural networks
AUTH: A/TROXEL, S. E.; B/ROGERS, S. K.; C/KABRISKY, M.; D/MILLS, J. P. PAA: D/(USAF, Institute of Technology, Wright-Patterson AFB, OH) IN: Digital and optical shape representation and pattern recognition; Proceedings of the Meeting, Orlando, FL, Apr. 4-6, 1988 (A89-23526 08-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1988, p. 295-301.
ABS: A method for classifying objects invariant to position, rotation, or scale is presented. Objects to be classified were multifunction laser radar data of tanks and trucks at various aspect angles. A segmented Doppler image was used to mask the range image into candidate targets. Each target was then compared to stored templates representing the different classes. A neural network was used to perform the classification with an accuracy near 100 percent. The neural network used in this study was a multilayer perceptron using a back propagation algorithm. 88/00/00 89A23556

UTTL: Neural-network techology and its applications
AUTH: A/ROTH, MICHAEL W. PAA: A/(Johns Hopkins University, Laurel, MD) Johns Hopkins APL Technical Digest (ISSN 0270-5214), vol. 9, July-Sept. 1988, p. 242-253.
ABS: This paper discusses recent developments in neural-network technology in the areas of models, algorithms, and special-purpose computational hardware. Special attention is given to the applications of neural-network technology in such areas as solutions of complex optimization problems, communication modems, pattern recognition, and engineering problems in control systems. 88/09/00 89A18786

UTTL: Artificial neural network approaches to target recognition
AUTH: A/BOWMAN, CHRISTOPHER PAA: A/(Ball Corp., Ball Systems Engineering Div., San Diego, CA) IN: AIAA/IEEE Digital Avionics Systems Conference, 8th, San Jose, CA, Oct. 17-20, 1988, Technical Papers. Part 2 (A89-18051 05-06). Washington, DC, American Institute of Aeronautics and Astronautics, 1988, p. 847-857.
ABS: Artificial Neural Network (ANN) technology is being successfully applied to a variety of pattern recognition problems. The ANN discovers features itself based upon user training. Trained ANN's settle fast to good solutions, thereby providing cost effective self-learned pattern recognition. This paper describes what ANN's are and how they are trained. A taxonomy is given along with ANN dynamics and training equations. ANN system development methodology is summarized. An application of ANN's to stereo image matching ANN and multisensor target recognition avionics is presented.
RPT#: AIAA PAPER 88-4029 88/00/00 89A18179

UTTL: Neural-network implementation of a scan-to-scan correlation algorithm
AUTH: A/MCCURRY, MAX E. PAA: A/(U.S. Army, Advanced Technology Directorate, Huntsville, AL) IN: High speed computing; Proceedings of the Meeting, Los Angeles, CA, Jan. 11, 12, 1988 (A89-14451 03-63). Bellingham, WA, Society of Photo-Optical Instrumentation Engineers, 1988, p. 85-87.
ABS: This paper presents a neural-network approach to the problem of multitarget tracking. The problem is formulated analytically in terms of desired optima and constraints that make it suitable for solution using the neural-network formalism of Hopfield-Tank. The results of computer simulations of a network designed to solve the problem are presented. 88/00/00 89A14460

UTTL: Generalization of back-propagation to recurrent neural networks
AUTH: A/PINEDA, FERNANDO J. PAA: A/(Johns Hopkins University, Laurel, MD) Physical Review Letters (ISSN 0031-9007), vol. 59, Nov. 9, 1987, p. 2229-2232.
ABS: An adaptive neural network with asymmetric connections is proposed that is related to the Hopfield (1984) network with graded neurons. The present back-propagation algorithm uses a recurrent generalization of the delta rule of Rumelhart et al. (1986) to adaptively modify the synaptic weights. The network is architecturally simpler than the master/slave network of Lapedes and Farber (1986), and it vectorizes naturally because the units are homogeneous. 87/11/09 88A18289

UTTL: Engineering cybernetics
AUTH: A/GLORIOSO, R. M. PAA: A/(Massachusetts, University, Amherst, Mass.) Englewood Cliffs, N.J., Prentice-Hall, Inc., 1975. 270 p.
ABS: The present work examines the concepts of adaptation, learning, self-organization, self-repair, game playing by machines, pattern recognition, and artificial intelligence, along with some applications of cybernetics which have emerged so far. The discussion covers fundamental computer organization and behavior, symbols and decisions in machines, information, logic, automata, and search techniques. Specific examples of adaptive, learning, and self-organizing systems as applied to control and communications are provided. The principles of redundant design, fault masking, and repair for creating reliable systems are discussed. Single and multilevel threshold logic synthesis are outlined along with descriptions of the Adaline (adaptive linear element), Madaline (multiple adaptive linear element), and the perceptron. Particular attention is devoted to pattern recognition, where the various aspects of the problem including systems for both optical and acoustical pattern recognition, feature extraction, and pattern classification are defined and analyzed. 75/00/00 76A19444

UTTL: Experiments in image recognition with the aid of expanding networks
AUTH: A/GLADUN, V. P.; B/MAZAEVA, S. P.; C/SAVA, I. G. Problemy Bioniki, no. 6, 1971, p. 63-69. In Russian.
ABS: An image recognition learning algorithm is proposed for a type of neural nets introduced by Gladun (1970) and called the expanding type. According to this definition, such neural nets are progressively built by spare back-up elements during the process of learning. The elements of such nets are identified as active inputs, receptors, associative elements and recognizers, connected by transmitting and forbidding couplings into a single body. Computer experiments are described to illustrate the work of this learning algorithm. 71/00/00 73A15794

UTTL: Memory-based reasoning for advanced launch system operations
AUTH: A/MYLER, HARLEY R.; B/DUBOIS, DEAN A. CORP: University of Central Florida, Orlando. CSS: (Dept. of Computer Engineering.) In its KSC-NASA/UCF Cooperative Agreement Research Projects 17 p (SEE N91-70698 09-61) 91/00/00 91N70701

UTTL: Cascading a systolic array and a feedforward neural network for navigation and obstacle avoidance using potential fields
AUTH: A/PLUMER, EDWARD S. CORP: Stanford Univ., CA. CSS: (Dept. of Electrical Engineering.)
ABS: A technique is developed for vehicle navigation and control in the presence of obstacles. A potential function was devised that peaks at the surface of obstacles and has its minimum at the proper vehicle destination. This function is computed using a systolic array and is guaranteed not to have local minima. A feedfoward neural network is then used to control the steering of the vehicle using local potential field information. In this case, the vehicle is a trailer truck backing up. Previous work has demonstrated the capability of a neural network to control steering of such a trailer truck backing to a loading platform, but without obstacles. Now, the neural network was able to learn to navigate a trailer truck around obstacles while backing toward its destination. The network is trained in an obstacle free space to follow the negative gradient of the field, after which the network is able to control and navigate the truck to its target destination in a space of obstacles which may be stationary or movable.
RPT#: NASA-CR-177575 A-91066 NAS 1.26:177575 91/02/00 91N19771

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

UTTL: Neural networks in nonlinear aircraft control
AUTH: A/LINSE, DENNIS J. CORP: Princeton Univ., NJ. CSS: (Dept. of Mechanical and Aerospace Engineering.) In NASA, Langley Research Center, Joint University Program for Air Transportation Research, 1989-1990 p 151-161 (SEE N91-19024 11-01)
ABS: Recent research indicates that artificial neural networks offer interesting learning or adaptive capabilities. The current research focuses on the potential for application of neural networks in a nonlinear aircraft control law. The current work has been to determine which networks are suitable for such an application and how they will fit into a nonlinear control law. 90/12/00 91N19037

UTTL: Neural networks as a control methodology
AUTH: A/MCCULLOUGH, CLAIRE L. CORP: Alabama Univ., Huntsville. CSS: (Dept. of Electrical and Computer Engineering.) In Alabama Univ., Research Reports: 1990 NASA/ASEE Summer Faculty Fellowship Program 8 p (SEE N91-18967 10-99)
ABS: While conventional computers must be programmed in a logical fashion by a person who thoroughly understands the task to be performed, the motivation behind neural networks is to develop machines which can train themselves to perform tasks, using available information about desired system behavior and learning from experience. There are three goals of this fellowship program: (1) to evaluate various neural net methods and generate computer software to implement those deemed most promising on a personal computer equipped with Matlab; (2) to evaluate methods currently in the professional literature for system control using neural nets to choose those most applicable to control of flexible structures; and (3) to apply the control strategies chosen in (2) to a computer simulation of a test article, the Control Structures Interaction Suitcase Demonstrator, which is a portable system consisting of a small flexible beam driven by a torque motor and mounted on springs tuned to the first flexible mode of the beam. Results of each are discussed. 90/10/00 91N18997

UTTL: Optimal control by neural networks
AUTH: A/BANKS, S. P.; B/HARRISON, R. F. CORP: Sheffield Univ. (England). CSS: (Dept. of Control Engineering.)
ABS: A neural network for the implementation of a nonlinear optimal controller is developed, based on an energy minimization principle. The theory is applicable to any nonlinear problem with a quadratic cost functional, although it would be easy to extend it to non quadratic functionals. A simple example of a scalar, linear, quadratic problem is presented.
RPT#: RR-399 ETN-91-98527 90/06/14 91N15797

UTTL: Massively parallel network architectures for automatic recognition of visual speech signals
AUTH: A/SEJNOWSKI, TERRENCE J.; B/GOLDSTEIN, MOISE CORP: Johns Hopkins Univ., Baltimore, MD.
ABS: This research sought to produce a massively parallel network architecture that could interpret speech signals from video recordings of human talkers. The project's results are summarized: (1) A corpus of video recordings from two human speakers was analyzed with image processing techniques and used as the data for this study; (2) It was demonstrated that a feedforward network could be trained to categorize vowels from these talkers (The performance was comparable to that of the nearest neighbors techniques and to trained humans on the same data); (3) A novel approach was developed to sensory fusion by training a network to transform from facial images to short-time spectral amplitude envelopes. This information can be used to increase the signal to noise ratio and hence the performance of acoustic speech recognition systems in noisy environments; and (4) The use was explored of recurrent networks to perform the same mapping for continuous speech. Results demonstrate the feasibility of adding a visual speech recognition component to enhance existing speech recognition systems. Such a combined system could be used in noisy environments, such as cockpits, where improved communication is needed. This demonstration of presymbolic fusion of visual and acoustic speech signals is consistent with the current understanding of human speech perception.
RPT#: AD-A226968 AFOSR-90-0949TR 90/00/00 91N14805

UTTL: Applications of neural networks to adaptive control
AUTH: A/SCOTT, RUSSELL W., II CORP: Naval Postgraduate School, Monterey, CA.
ABS: The amount of a priori knowledge required to design some modern control systems is becoming prohibitive. Two current methods addressing this problem are robust control, in which the control design is insensitive to errors in system knowledge, and adaptive control, in which the control law is adjusted in response to a continually updated model of the system. This thesis examines the application of parallel distributed processing (neural networks) to the problem of adaptive control. The structure of neural networks is introduced, focusing on the Backpropagation paradigm. A general form of controller consistent with use in neural networks is developed and combined with a discussion of linear least squares parameter estimation techniques to suggest a structure for neural network adaptive controllers. This neural network adaptive control structure is then applied to a number of estimation and control problems using as a model the longitudinal motion of the A-4 aircraft. The purpose of this thesis is to develop and demonstrate a neural network adaptive control structure consistent with adaptive control theory.
RPT#: AD-A225408 89/12/00 91N13938

UTTL: Target detection in Gaussian noise using artificial neural systems
AUTH: A/SOLKA, JEFFREY L.; B/ROGERS, GEORGE CORP: Naval Surface Warfare Center, Dahlgren, VA. CSS: (Strategic Systems Dept.)
ABS: Radar signal processing with multilayered perceptrons was investigated. Networks with no hidden layer and a single hidden layer were tested on field collected millimeter wave target returns that have been corrupted with artificial Gaussian noise at a signal to noise level of 3 dB. Performance as a function of network architecture was characterized.
RPT#: AD-A223983 NSWC/TR-90-171 90/06/00 90N28770

UTTL: Analog hardware for learning neural networks
AUTH: A/EBERHART, SILVIO P. PAA: A/(Jet Propulsion Lab., California Inst. of Tech., Pasadena.) CORP: National Aeronautics and Space Administration. Pasadena Office, CA. ; Jet Propulsion Lab., California Inst. of Tech., Pasadena.
ABS: This is a recurrent or feedforward analog neural network processor having a multi-level neuron array and a synaptic matrix for storing weighted analog values of synaptic connection strengths which is characterized by temporarily changing one connection strength at a time to determine its effect on system output relative to the desired target. That connection strength is then adjusted based on the effect, whereby the processor is taught the correct response to training examples connection by connection.
RPT#: NASA-CASE-NPO-17664-1-CU NAS 1.71:NPO-17664-1-CU US-PATENT-APPL-SN-463720 89/12/28 90N27384

UTTL: OMS FDIR: Initial prototyping
AUTH: A/TAYLOR, ERIC W.; B/HANSON, MATTHEW A. CORP: Ford Aerospace and Communications Corp., Sunnyvale, CA. In NASA, Lyndon B. Johnson Space Center, Third Annual Workshop on Space Operations Automation and Robotics (SOAR 1989) p 545-549 (SEE N90-25503 19-59)
ABS: The Space Station Freedom Program (SSFP) Operations Management System (OMS) will automate major management functions which coordinate the operations of onboard systems, elements and payloads. The objectives of OMS are to improve safety, reliability and productivity while reducing maintenance and operations cost. This will be accomplished by using advanced automation techniques to automate much of the activity currently performed by the flight crew and ground personnel. OMS requirements have been organized into five task groups: (1) Planning, Execution and Replanning; (2) Data Gathering, Preprocessing and Storage; (3) Testing and Training; (4) Resource Management; and (5) Caution and Warning and Fault Management for onboard subsystems. The scope of this prototyping effort falls within the Fault Management requirements group. The prototyping will be performed in two phases. Phase 1 is the development of an onboard communications network fault detection, isolation, and reconfiguration (FDIR) system. Phase 2 will incorporate global FDIR for onboard systems. Research into the applicability of expert systems, object-oriented programming, fuzzy sets, neural networks and other advanced techniques will be conducted. The goals and technical approach for this new SSFP research project are discussed here. 90/03/00 90N25562

UTTL: A comparison of two neural network schemes for navigation
AUTH: A/MUNRO, PAUL CORP: Pittsburgh Univ., PA. CSS: (Dept. of Information Science.) In NASA, Lyndon B. Johnson Space Center, Third Annual Workshop on Space Operations Automation and Robotics (SOAR 1989) p 305-310 (SEE N90-25503 19-59)
ABS: Neural networks have been applied to tasks in several areas of artificial intelligence, including vision, speech, and language. Relatively little work has been done in the area of problem solving. Two approaches to path-finding are presented, both using neural network techniques. Both techniques require a training period. Training under the back propagation (BPL) method was accomplished by presenting representations of current position, goal position pairs as input and appropriate actions as output. The Hebbian/interactive activation (HIA) method uses the Hebbian rule to associate points that are nearby. A path to a goal is found by activating a representation of the goal in the network and processing until the current position is activated above some threshold level. BPL, using back-propagation learning, failed to learn, except in a very trivial fashion, that is equivalent to table lookup techniques. HIA, performed much better, and required storage of fewer weights. In drawing a comparison, it is important to note that back propagation techniques depend critically upon the forms of representation used, and can be sensitive to parameters in the simulations; hence the BPL technique may yet yield strong results. 90/03/00 90N25536

UTTL: A comparison of two neural network schemes for navigation
AUTH: A/MUNRO, PAUL W. CORP: Pittsburgh Univ., PA. CSS: (Dept. of Information Science.) In Texas A&M Univ., NASA/ASEE Summer Faculty Fellowship Program-1989, Volume 2 10 p (SEE N90-24985 18-80)
ABS: Neural networks have been applied to tasks in several areas of artificial intelligence, including vision, speech, and language. Relatively little work has been done in the area of problem solving. Two approaches to path-finding are presented, both using neural network techniques. Both techniques require a training period. Training under the back propagation (BPL) method was

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

accomplished by presenting representations of (current position, goal position) pairs as input and appropriate actions as output. The Hebbian/interactive activation (HIA) method uses the Hebbian rule to associate points that are nearby. A path to a goal is found by activating a representation of the goal in the network and processing until the current position is activated above some threshold level. BPL, using back-propagation learning, failed to learn, except in a very trivial fashion, that is equivalent to table lookup techniques. HIA, performed much better, and required storage of fewer weights. In drawing a comparison, it is important to note that back propagation techniques depend critically upon the forms of representation used, and can be sensitive to parameters in the simulations; hence the BPL technique may yet yield strong results. 89/12/00 90N24991

UTTL: Neuromorphic optical signal processing and image understanding for automated target recognition
AUTH: A/FARHAT, NABIL H. CORP: Pennsylvania Univ., Philadelphia.
ABS: The goal of research is study of computation and learning in neural net models and demonstration of their utility in image understanding and neuromorphic information processing systems for remote sensing and target identification. The approach to achieving this goal has two facets. One is combining innovative architectures and methodologies with suitable algorithms to exploit existing and emerging photonic technology in the implementation of large-scale neurocomputers for use in: the study of complex self-organizing and learning systems, fast solution of optimization problems, feature extraction, (formation of object representation), and pattern recognition. The second facet of the approach is to demonstrate and assess the capabilities of neuromorphic processing in solution of selected inverse-scattering and recognition problems. The problem studied as a test bed for the work is that of automated radar target recognition because of the existing capabilities and expertise in this area.
RPT#: AD-A219827 EO/MO-89-1 89/12/00 90N23884

UTTL: Neural networks in support of manned space
AUTH: A/WERBOS, PAUL J. CORP: National Science Foundation, Washington, DC. In Jet Propulsion Lab., California Inst. of Tech., Proceedings of the 3rd Annual Conference on Aerospace Computational Control, Volume 2 p 916 (SEE N90-23040 16-61)
ABS: Many lobbyists in Washington have argued that artificial intelligence (AI) is an alternative to manned space activity. In actuality, this is the opposite of the truth, especially as regards artificial neural networks (ANNs), that form of AI which has the greatest hope of mimicking human abilities in learning, ability to interface with sensors and actuators, flexibility and balanced judgement. ANNs and their relation to expert systems (the more traditional form of AI), and the limitations of both technologies are briefly reviewed. A few highlights of recent work on ANNs, including an NSF-sponsored workshop on ANNs for control applications are given. Current thinking on ANNs for use in certain key areas (the National Aerospace Plane, teleoperation, the control of large structures, fault diagnostics, and docking) which may be crucial to the long term future of man in space is discussed. 89/12/15 90N23088

UTTL: ALVINN: An Autonomous Land Vehicle In a Neural Nettwork
AUTH: A/POMERLEAU, DEAN A. CORP: Carnegie-Mellon Univ., Pittsburgh, PA.; Pittsburgh Univ., PA. CSS: (Artificial Intelligence and Psychology Project.) Presented at the IEEE Conference on Neural Information Processing Systems: Natural and Synthetic, Denver, CO, Nov. 1988
ABS: ALVINN (Autonomous Land Vehicle In a Neural Network) is an 3 layer back propagation network designed for the task of road following. Currently ALVINN takes images from a camera and a laser range finder as input and produces as output the direction the vehicle should travel in order to follow the road. Training was conducted using simulated road images. Successful tests on the Carnegie Mellon autonomous navigation test vehicle indicate that the network can follow real roads under certain field conditions. The representation developed to perform the task differs greatly when the network is trained under various conditions, suggesting the possibility of a novel adaptive autonomous navigation system capable of tailoring its processing to the conditions at hand.
RPT#: AD-A218975 AIP-77 89/01/00 90N22797

UTTL: A real time neural net estimator of fatigue life
AUTH: A/TROUDET, T.; B/MERRILL, W. PAA: A/(Sverdrup Technology, Inc., Cleveland, OH.) CORP: National Aeronautics and Space Administration. Lewis Research Center, Cleveland, OH. Presented at the International Joint Conference on Neural Networks, San Diego, CA, 17-21 Jun. 1990; cosponsored by IEEE and INNS
ABS: A neural net architecture is proposed to estimate, in real-time, the fatigue life of mechanical components, as part of the Intelligent Control System for Reusable Rocket Engines. Arbitrary component loading values were used as input to train a two hidden-layer feedforward neural net to estimate component fatigue damage. The ability of the net to learn, based on a local strain approach, the mapping between load sequence and fatigue damage has been demonstrated for a uniaxial specimen. Because of its demonstrated performance, the neural computation may be extended to complex cases where the loads are biaxial or triaxial, and the geometry of the component is complex (e.g., turbopump blades). The generality of the approach is such that load/damage mappings can be directly extracted from experimental data without requiring any knowledge of the stress/strain profile of the component. In addition, the parallel network architecture allows real-time life calculations even for high frequency vibrations. Owing to its distributed nature, the neural implementation will be robust and reliable, enabling its use in hostile environments such as rocket engines. This neural net estimator of fatigue life is seen as the enabling technology to achieve component life prognosis, and therefore would be an important part of life extending control for reusable rocket engines.
RPT#: NASA-TM-103117 E-5217 NAS 1.15:103117 90/00/00 90N21564

UTTL: Neural networks for aircraft control
AUTH: A/LINSE, DENNIS CORP: Princeton Univ., NJ. CSS: (Dept. of Mechanical and Aerospace Engineering.) In NASA, Langley Research Center, Joint University Program for Air Transportation Research, 1988-1989 p 167-181 (SEE N90-20921 14-01)
ABS: Current research in Artificial Neural Networks indicates that networks offer some potential advantages in adaptation and fault tolerance. This research is directed at determining the possible applicability of neural networks to aircraft control. The first application will be to aircraft trim. Neural network node characteristics, network topology and operation, neural network learning and example histories using neighboring optimal control with a neural net are discussed. 90/03/00 90N20937

UTTL: Computation and control with neural nets
AUTH: A/CORNELIUSEN, A.; B/TERDAL, P.; C/KNIGHT, T.; D/SPENCER, J. CORP: Stanford Linear Accelerator Center, CA. Presented at the International Conference on Accelerator and Large Experimental Physics Control Systems, Vancouver, British Columbia, 30 Oct. - 3 Nov. 1989
ABS: As energies have increased exponentially with time so have the size and complexity of accelerators and control systems. Neural nets (NN) may offer the kinds of improvements in computation and control that are needed to maintain acceptable functionality. For control their associative characteristics could provide signal conversion or data translation. Because they can do any computation such as least squares, they can close feedback loops autonomously to provide intelligent control at the point of action rather than at a central location that requires transfers, conversions, hand-shaking and other costly repetitions like input protection. Both computation and control can be integrated on a single chip, printed circuit or an optical equivalent that is also inherently faster through full parallel operation. For such reasons one expects lower costs and better results. Such systems could be optimized by integrating sensor and signal processing functions. Distributed nets of such hardware could communicate and provide global monitoring and multiprocessing in various ways e.g., via token, slotted or parallel rings (or Steiner trees) for compatibility with existing systems. Problems and advantages of this approach such as an optimal, real-time Turing machine are discussed. Simple examples are simulated and hardware implemented using discrete elements.
RPT#: DE90-006460 SU-SLAC-PUB-5035 CONF-891094-14 89/10/00 90N18911

UTTL: Neurobeamformer 2: Further exploration of adaptive beamforming via neural networks
AUTH: A/SPEIDEL, S. L. CORP: Naval Ocean Systems Center, San Diego, CA. CSS: (Analysis Branch.)
ABS: This paper discussed neural network technology as a tool for signal processing. Test results show that the adaptive beamformer method, based on neural network technology, performs the desired function of directing a beam so as to enhance a target signal and reject noise and interference. Comparing test output values with a matched-correlation output shows that the plotted crossbar circuit energy minima follow the shape of an inverted match-filter output. The neurobeamformer has certain advantages of implementation and adaptability over other methods. In concept, it is implementable in analog circuitry with no control code required. Thus a compact, simple, low-cost processor component that is not sensitive to array grooming can be produced. A straightforward adaptive beamformer cannot match the interference-cancellation performance of more exotic methods, which include sidelobe cancellers. So, a neuroprocessor, that will include a neurobeamformer as a component, will be built. This neuroprocessor will provide for cancellation of sidelobes, enhance source discrimination and angle-estimation through interaction of beams. Plans for this extended network were influenced by studies of the literature in biological sensory processing, both peripheral and central.
RPT#: AD-A215118 NOSC/TD-1606 89/06/00 90N18226

UTTL: Knowledge-based imaging-sensor fusion system
AUTH: A/WESTROM, GEORGE CORP: Odetics, Inc., Anaheim, CA. In NASA, Langley Research Center, Visual Information Processing for Television and Telerobotics p 215-229 (SEE N90-16204 08-35)
ABS: An imaging system which applies knowledge-based technology to supervise and control both sensor hardware and computation in the imaging system is described. It includes the development of an imaging system breadboard which brings together into one system work that we and others have pursued for LaRC for several years. The goal is to combine Digital Signal Processing (DSP) with

This Document Provided by the ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

Knowledge-Based Processing and also include Neural Net processing. The system is considered a smart camera. Imagine that there is a microgravity experiment on-board Space Station Freedom with a high frame rate, high resolution camera. All the data cannot possibly be acquired from a laboratory on Earth. In fact, only a small fraction of the data will be received. Again, imagine being responsible for some experiments on Mars with the Mars Rover: the data rate is a few kilobits per second for data from several sensors and instruments. Would it not be preferable to have a smart system which would have some human knowledge and yet follow some instructions and attempt to make the best use of the limited bandwidth for transmission. The system concept, current status of the breadboard system and some recent experiments at the Mars-like Amboy Lava Fields in California are discussed. 89/11/00 90N16220

UTTL: Real-time support for high performance aircraft operation
AUTH: A/VIDAL, JACQUES J. CORP: California Univ., Los Angeles. CSS: (Dept. of Computer Science.)
ABS: The feasibility of real-time processing schemes using artificial neural networks (ANNs) is investigated. A rationale for digital neural nets is presented and a general processor architecture for control applications is illustrated. Research results on ANN structures for real-time applications are given. Research results on ANN algorithms for real-time control are also shown.
RPT#: NASA-CR-185475 NAS 1.26:185475 89/01/00 90N10075

UTTL: Integration of perception and reasoning in fast neural modules
AUTH: A/FRITZ, DAVID G. CORP: George Washington Univ., Washington, DC.; Cognitive Information Systems Co., Silver Spring, MD. CSS: (Inst. for Artificial Intelligence.) In NASA. Goddard Space Flight Center, The 1989 Goddard Conference on Space Applications of Artificial Intelligence p 349-356 (SEE N89-26578 20-63)
ABS: Artificial neural systems promise to integrate symbolic and sub-symbolic processing to achieve real time control of physical systems. Two potential alternatives exist. In one, neural nets can be used to front-end expert systems. The expert systems, in turn, are developed with varying degrees of parallelism, including their implementation in neural nets. In the other, rule-based reasoning and sensor data can be integrated within a single hybrid neural system. The hybrid system reacts as a unit to provide decisions (problem solutions) based on the simultaneous evaluation of data and rules. Discussed here is a model hybrid system based on the fuzzy cognitive map (FCM). The operation of the model is illustrated with the control of a hypothetical satellite that intelligently alters its attitude in space in response to an intersecting micrometeorite shower. 89/04/00 89N26603

UTTL: Empirical analysis and refinement of expert system knowledge bases
AUTH: A/WEISS, SHOLOM M.; B/KULIKOWSKI, CASIMIR A. CORP: Rutgers - The State Univ., New Brunswick, NJ. CSS: (Center for Expert Systems Research.)
ABS: Classification methods from statistical pattern recognition, neural nets, and machine learning were applied to four real-world data sets. Each of these data sets has been previously analyzed and reported in the statistical, medical, or machine learning literature. The data sets are characterized by statistical uncertainty; there is no completely accurate solution to these problems. Training and testing or resampling techniques are used to estimate the true error rates of classification methods. Detailed attention is given to the analysis of performance of the neural nets using back propagation. For these problems, which have relatively few hypotheses and features, the machine learning procedures for rule induction or tree induction clearly performed best.
RPT#: AD-A206226 89/02/28 89N24858

UTTL: Neuromorphic learning of continuous-valued mappings in the presence of noise: Application to real-time adaptive control
AUTH: A/TROUDET, TERRY; B/MERRILL, WALTER C. PAA: A/(Sverdrup Technology, Inc., Cleveland, OH.) CORP: National Aeronautics and Space Administration. Lewis Research Center, Cleveland, OH. Presented at the International Conference on Neural Networks, Washington, DC, 18-22 Jun. 1989; sponsored by the IEEE
ABS: The ability of feed-forward neural net architectures to learn continuous-valued mappings in the presence of noise is demonstrated in relation to parameter identification and real-time adaptive control applications. Factors and parameters influencing the learning performance of such nets in the presence of noise are identified. Their effects are discussed through a computer simulation of the Back-Error-Propagation algorithm by taking the example of the cart-pole system controlled by a nonlinear control law. Adequate sampling of the state space is found to be essential for canceling the effect of the statistical fluctuations and allowing learning to take place.
RPT#: NASA-TM-101999 E-4706 NAS 1.15:101999 89/00/00 89N24856

UTTL: Modified backward error propagation for tactical target recognition
AUTH: A/PIAZZA, CHARLES C. CORP: Air Force Inst. of Tech., Wright-Patterson AFB, OH. CSS: (School of Engineering.)
ABS: This thesis explores a new approach to the classification of tactical targets using a new biologically-based neural network. The targets of interest were generated from Doppler imagery and forward looking infrared imagery, and consisted of tanks, trucks, armored personnel carriers, jeeps and petroleum, oil, and lubricant tankers. Each target was described by feature vectors, such as normalized moment invariants. The features were generated from the imagery using a segmenting process. These feature vectors were used as the input to a neural network classifier for tactical target recognition. The neural network consisted of a multilayer perceptron architecture, employing a backward error propagation learning algorithm. The minimization technique used was an approximation to Newton's method. This second order algorithm is a generalized version of well known first order techniques, i.e., gradient of steepest descent and momentum methods. Classification using both first and second order techniques was performed, with comparisons drawn.
RPT#: AD-A202666 AFIT/GE/ENG/88D-36 88/12/00 89N21644

UTTL: Automatic voice recognition using traditional and artificial neural network approaches
AUTH: A/BOTROS, NAZEIH M. CORP: University of Southern Illinois, Carbondale. CSS: (Dept. of Electrical Engineering.) In NASA, Lyndon B. Johnson Space Center, National Aeronautics and Space Administration (NASA)/American Society for Engineering Education (ASEE) Summer Faculty Fellowship Program 1988, Volume 1 13 p (SEE N89-20058 12-99)
ABS: The main objective of this research is to develop an algorithm for isolated-word recognition. This research is focused on digital signal analysis rather than linguistic analysis of speech. Features extraction is carried out by applying a Linear Predictive Coding (LPC) algorithm with order of 10. Continuous-word and speaker independent recognition will be considered in future study after accomplishing this isolated word research. To examine the similarity between the reference and the training sets, two approaches are explored. The first is implementing traditional pattern recognition techniques where a dynamic time warping algorithm is applied to align the two sets and calculate the probability of matching by measuring the Euclidean distance between the two sets. The second is implementing a backpropagation artificial neural net model with three layers as the pattern classifier. The adaptation rule implemented in this network is the generalized least mean square (LMS) rule. The first approach has been accomplished. A vocabulary of 50 words was selected and tested. The accuracy of the algorithm was found to be around 85 percent. The second approach is in progress at the present time. 89/02/00 89N20064

UTTL: Simulation tests of the optimization method of Hopfield and Tank using neural networks
AUTH: A/PAIELLI, RUSSELL A. CORP: National Aeronautics and Space Administration. Ames Research Center, Moffett Field, CA.
ABS: The method proposed by Hopfield and Tank for using the Hopfield neural network with continuous valued neurons to solve the traveling salesman problem is tested by simulation. Several researchers have apparently been unable to successfully repeat the numerical simulation documented by Hopfield and Tank. However, as suggested to the author by Adams, it appears that the reason for those difficulties is that a key parameter value is reported erroneously (by four orders of magnitude) in the original paper. When a reasonable value is used for that parameter, the network performs generally as claimed. Additionally, a new method of using feedback to control the input bias currents to the amplifiers is proposed and successfully tested. This eliminates the need to set the input currents by trial and error.
RPT#: NASA-TM-101047 A-88275 NAS 1.15:101047 88/11/00 89N14004

UTTL: Genetic algorithms for adaptive real-time control in space systems
AUTH: A/VANDERZIJP, J.; B/CHOUDRY, A. CORP: Alabama Univ., Huntsville. CSS: (Center for Applied Optics.) In NASA, Marshall Space Flight Center, Third Conference on Artificial Intelligence for Space Applications, Part 2 p 47-51 (SEE N88-24188 17-61)
ABS: Genetic Algorithms that are used for learning as one way to control the combinational explosion associated with the generation of new rules are discussed. The Genetic Algorithm approach tends to work best when it can be applied to a domain independent knowledge representation. Applications to real time control in space systems are discussed. 88/06/00 88N24195

UTTL: Third Conference on Artificial Intelligence for Space Applications, part 2
AUTH: A/DENTON, JUDITH S.; B/FREEMAN, MICHAEL S.; C/VEREEN, MARY CORP: National Aeronautics and Space Administration. Marshall Space Flight Center, Huntsville, AL. Conference held in Huntsville, Ala., 2-3 Nov. 1987; sponsored by NASA, Marshall Space Flight Center, Huntsville, Ala. and Alabama Univ., Huntsville ANN: Topics relative to the application of artificial intelligence to space operations are discussed. New technologies for space station automation, design data capture, computer vision, neural nets, automatic programming, and real time applications are discussed. For individual titles, see N88-24189 through N88-24197.
RPT#: NASA-CP-2492-PT-2 M-576-PT-2 NAS 1.55:2492-PT-2 88/06/00 88N24188

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

UTTL: Memory efficient evaluations of nonlinear stochastic equations and C3 applications
AUTH: A/CONNELL, JOHN C., JR. CORP: Naval Postgraduate School, Monterey, CA.
ABS: The Statistical Mechanical Neural Computer (SMNC) developed in this thesis utilizes a Statistical Mechanical Nonlinear Algorithm (SMNA) to determine the long-time probability distribution of highly nonlinear stochastic systems. The use of the SMNA and a novel mesoscopic scaling technique help provide the SMNC with the capabilities of neural computers without the drawbacks of huge connection matrices and their attendant computational requirements. In this thesis, the SMNC is initially used to verify the ability of the SMNA to duplicate relatively simple, single variable path integral solutions to nonlinear Fokker-Planck equations. After the fundamental algorithms are validated, the SMNC's ability to simulate a two-variable, multicellular problem by modeling a portion of the neocortex consisting of 100,000 neural units is discussed. There are many important applications of the SMNC and its unique SMNA to C3 systems including radar, sonar and electronic signals processing, missile guidance systems and an integrated battle management system. Such C3 systems will benefit from the SMNC's potential to efficiently filter large amounts of data, recognize patterns and anticipate, with some degree of uncertainty, the future state of highly nonlinear stochastic systems.
RPT#: AD-A189872 87/12/00 88N22569

UTTL: Position, scale, and rotation invariant target recognition using range imagery
AUTH: A/TROXEL, STEVEN E. CORP: Air Force Inst. of Tech., Wright-Patterson AFB, OH. CSS: (School of Engineering.)
ABS: This thesis explores a new approach to the recognition of tactical targets using a multifunction laser radar sensor. Targets of interest were tanks, jeeps, and trucks. Doppler images were segmented and overlaided onto a relative range image. The resultant shapes were then transformed into a position, scale, and rotation invariant (PSRI) feature space. The classification processes used the correlation peak of the template PSRI space and the target PSRI space as features. Two classification methods were implemented: a classical distance measurement approach and a new biologically-based neural network multilayer perception architecture. Both methods demonstrated classification rates near 100 percent with a true rotation invariance demonstrated up to 20 degrees. Neural networks were shown to have a distinct advantage in a robust environment and when a figure of merit criteria was applied. A space domain correlation was developed using local normalization and multistage processing to locate and classify targets in high clutter and with partially occluded targets.
RPT#: AD-A188828 AFIT/GEO/ENG/87D-3 87/12/00 88N19772

UTTL: Automated radar target recognition based on models of neural nets
AUTH: A/MIYAHARA, SHUNJI CORP: Pennsylvania Univ., Philadelphia.
ABS: Two methods of target recognition are proposed: (1) the use of sinogram representations as learning set in associative memory, based on models of neural nets as classifier; and (2) use of polarization representation for use in neural net based associative memory as a classifier. Using microwave scattering data of scaled model targets, the concepts for the target recognition were demonstrated by computer simulation of a 1024 (32 by 32) element neural net associative memory based on the outer product model. The simulations show that partial input, consisting of less than 10 percent of the total information, can identify the targets. Two-dimensional optical implementations of a neural net of 8 by 8 binary neurons were studied. Fault tolerance and robustness were examined, using a four-dimensional clipped outer product ternary Tijkl mask to establish the weighted interconnections of the net and electronic feedback based on closed loop TV systems. The performance was found to be in agreement with that of computer simulation, even though aberration of lenses and the defects of the system were present. These results confirm the practical suitability of the opto-electronic approach to the neural net implementation and pave the way for the implementation of larger networks. 87/00/00 88N18804

UTTL: Teaching artificial neural systems to drive: Manual training techniques for autonomous systems
AUTH: A/SHEPANSKI, J. F.; B/MACY, S. A. CORP: TRW, Inc., Redondo Beach, CA. In NASA. Lyndon B. Johnson Space Center, Houston, Texas, First Annual Workshop on Space Operations Automation and Robotics (SOAR 87) p 231-238 (SEE N88-17206 09-59)
ABS: A methodology was developed for manually training autonomous control systems based on artificial neural systems (ANS). In applications where the rule set governing an expert's decisions is difficult to formulate, ANS can be used to extract rules by associating the information an expert receives with the actions taken. Properly constructed networks imitate rules of behavior that permits them to function autonomously when they are trained on the spanning set of possible situations. This training can be provided manually, either under the direct supervision of a system trainer, or indirectly using a background mode where the networks assimilates training data as the expert performs its day-to-day tasks. To demonstrate these methods, an ANS network was trained to drive a vehicle through simulated freeway traffic.
87/10/00 88N17238

UTTL: NASA JSC neural network survey results
AUTH: A/GREENWOOD, DAN CORP: Netrologic, Inc., San Diego, CA. In NASA. Lyndon B. Johnson Space Center, Houston, Texas, First Annual Workshop on Space Operations Automation and Robotics (SOAR 87) p 97-110 (SEE N88-17206 09-59)
ABS: A survey of Artificial Neural Systems in support of NASA's (Johnson Space Center) Automatic Perception for Mission Planning and Flight Control Research Program was conducted. Several of the world's leading researchers contributed papers containing their most recent results on artificial neural systems. These papers were broken into categories and descriptive accounts of the results make up a large part of this report. Also included is material on sources of information on artificial neural systems such as books, technical reports, software tools, etc.
87/10/00 88N17220

UTTL: Models of the vestibular system and postural control
AUTH: A/YOUNG, L. R.; B/WEISS, A. PAA: B/(Mass. Eye and Ear Infirmary) CORP: Massachusetts Inst. of Tech., Cambridge. In NASA. Ames Res. Center Technol. and the Neurologically Handicapped p 151-168 (SEE N75-19975 11-54)
ABS: Applications of control theory and systems analysis to the problem of orientation and posture control are discussed, with the possible long range goals of contributing to the development of hardware for rehabilitation of the handicapped. 74/00/00 75N19992

UTTL: The brain as a model for LSI
AUTH: A/ALBUS, J. S. CORP: National Aeronautics and Space Administration. Goddard Space Flight Center, Greenbelt, MD. IN ITS SIGNIFICANT ACCOMPLISHMENTS IN SCI. AND TECHNOL. AT GODDARD SPACE FLIGHT CENTER 1970 P 292-294 /SEE N71-25256 13-34/ 70/00/00 71N25326

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

LS-179

INTRODUCTION TO NEURAL COMPUTING AND CATEGORIES OF NEURAL NETWORK APPLICATIONS TO GUIDANCE, NAVIGATION AND CONTROL

by

Uwe K. Krogmann

Bodenseewerk Gerätetechnik GmbH

Intelligent Systems Division

Nussdorfer Str. - D-7770 Überlingen

FRG

1. Introduction

"Future computer generation imitates man". "Many small cells are stronger than one large cell!". Such headlines are to be found in the media in connection with a new kind of information processing, the so-called "Artificial Neural Networks (ANN)". As the term suggests, these networks are an attempt to imitate the biological paradigm, our brain, in structure and function.

In the course of evolution our central nervous system (brain and spinal cord) has developed into a gigantic information- processing network to which the sensory paths from sense organs lead and from which the motor paths lead to the muscles. All stimuli are supplied to the central nervous system where they are processed into perceptions, sensations etc. and trigger off our actions.

In our organism many organ systems work together. Only the central nervous system communicates as superior system with all others by collecting their information and coordinating their functions.

Basically similar problems will be found in future technical equipment and systems. Based on the structure of the biological brain, the creation of artificial neural networks (abbreviated ANN) is aimed to technically realize capabilities and characteristics such as self-organisation, learning and associative memory. This is achieved by the particular structure of neural networks where a large number of simple processor elements (PE) are interconnected with uni-directional signal channels to single- or multi-layer networks. All processing elements are working in parallel as compared to one central, extremely efficient computer for sequential arithmetic and/or symbolic information processing.

For the solution of a problem with a conventional computer (e.g. personal computer (PC)) an algorithm, a procedure or a set of rules has to be developed and coded in software, i.e. a sequence of instructions. These instructions are then carried out sequentially by the computer.

By contrast, ANNs are not programmed but trained and learn like their biological paradigm, the brain. This is done by changing the intensity of the connections between the processor elements and by generating or eliminating structural connections. Thus the "knowledge" of an ANN lies in the topology and in the intensity of its connections, i.e. the strength of the connection weights between the PEs.

With their capabilities of self-organisation, learning (adaptation) and association, ANNs can be used wherever it is difficult to describe a problem algorithmically, the development of the operational software is very cost-intensive or wherever unprecise, incomplete or even contradictory input data must be considered. Owing to the parallel information processing ANN are fault-tolerant and thus very reliable.

Ever-increasing requirements placed on more demanding and complex systems on the one hand and financial resources getting increasingly scarce on the other force us to filter out key technologies showing the potential for a high cost-benefit ratio to meet the increased requirements. In this respect Artificial Neural Networks represent a new technology in the field of signal and information processing for Guidance and Control systems. This article is intended to give a short introduction into ANN and their application in guidance and control.

2. General Structure of Guidance and Control Problems

G.a.C. problems extend over several hierarchically structured levels and the communication functions between these levels as shown in Fig. 1. The represented interconnection of the different function levels (scenario, mission, trajectory, air vehicle state) can be conceived of as a hierarchically structured control system. The objects on which G.a.C. functions are performed on the mentioned levels represent the control plants. Information processing by which actuation is generated on all levels from sensor information represents the controller which is of primary concern here (Fig. 2).

The controlling feedback chain typical of all G.a.C. levels requires functions such as recognizing and assessing the situation; defining action goals; generating optimum or favorable solutions; decision-making; planning and finally performing as well as monitoring of actions. Hence, behavior levels of mental capabilities can be assigned to the function levels (Fig. 1).

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

CASCADED G.a.C. LEVELS

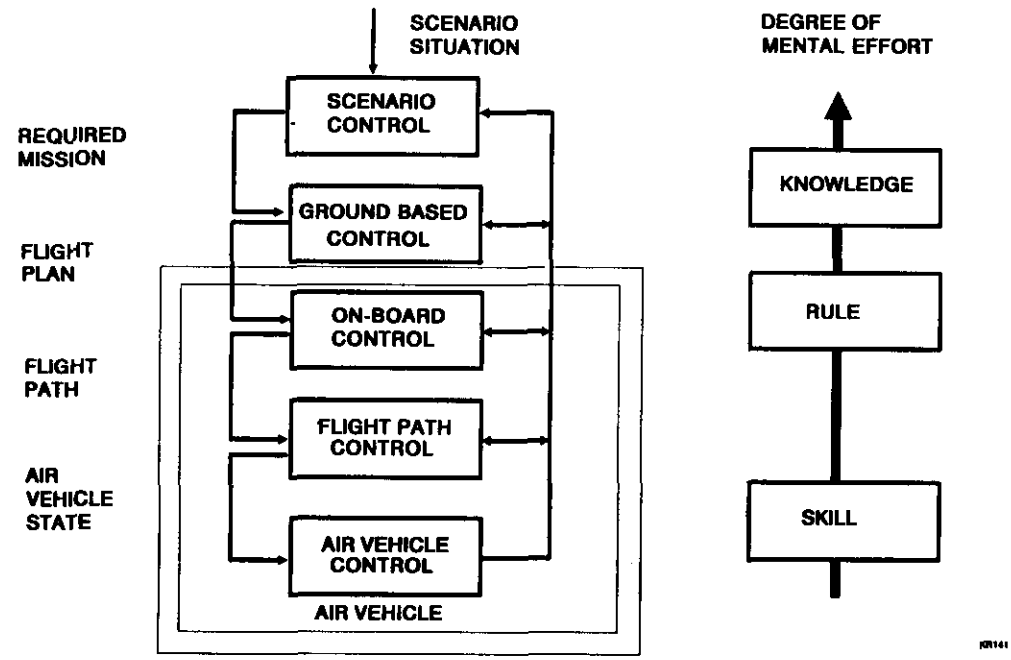
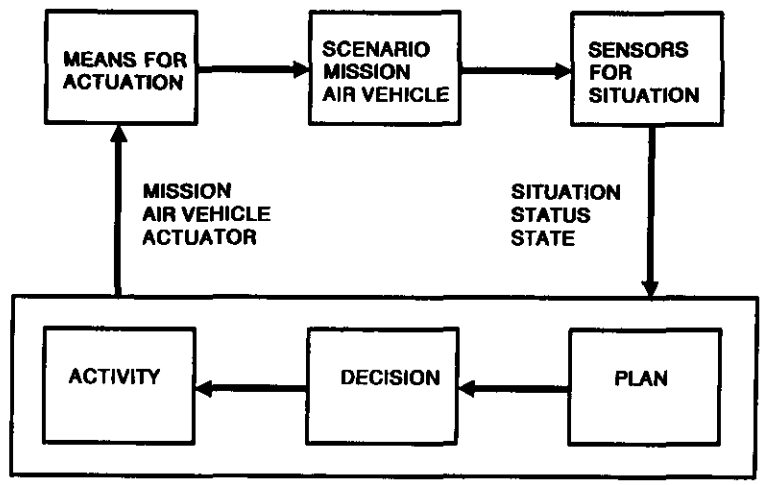


FIGURE 1

COMMON BASIC STRUCTURE OF G.a.C. LEVELS



CONTROLLER/INFORMATION PROCESSING

FIGURE 2

For reasons of human limitations in more demanding dynamic scenarios and in the operation of complex, highly integrated systems, there is the necessity for extended automation of these functions on higher levels such as trajectory control as well as mission management and control. Furthermore, the implementation of intelligent functions on lower levels such as the fusion and interpretation of sensor data, multifunctional use of sensor information and smart/brilliant sensors become inevitable.

The technical implementation of the intelligent G.a.C. feedback chain functions leads to a signal processing structure which contains conventional arithmetic, symbolic and sub-symbolic elements (Fig. 3). Whereas the symbolic element can be implemented utilizing expert-system software techniques, the sub-symbolic element represents the application of ANNs. In building ANNs the brain is utilized as biological paradigm. In the following its function and structure are to be briefly explained as far as this is important for understanding ANNs.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

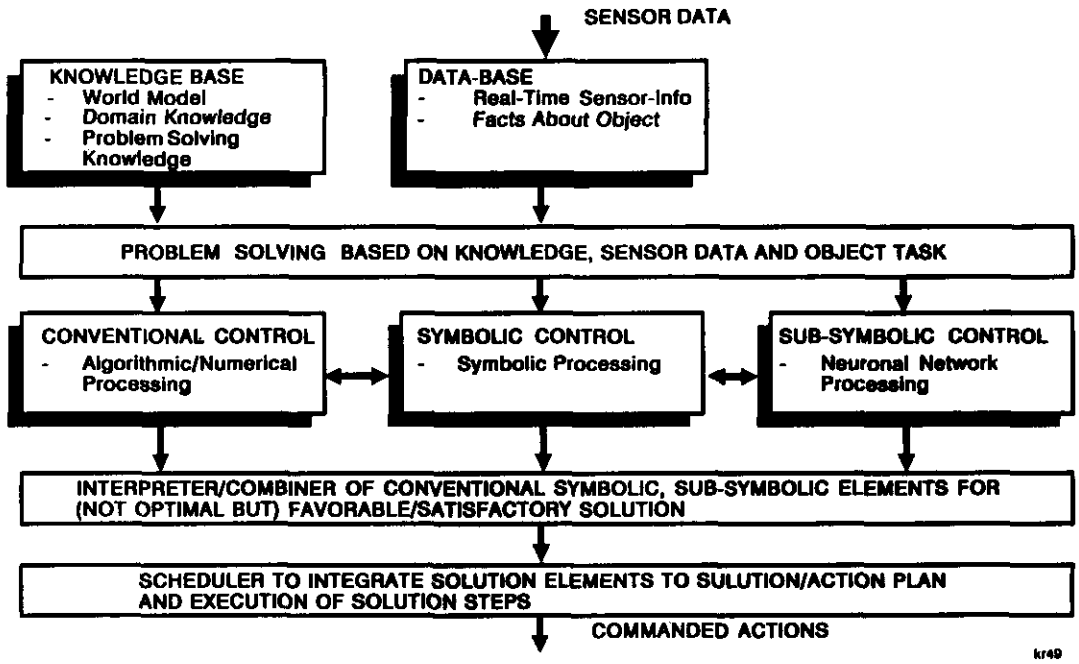


FIGURE 3

3. The biological brain as paradigm

Function and structure

Two different functions of the brain are to be looked at. First, there is the rational thinking with a function in conscious steps performed in a particular serial sequence. The digital computers we use today with a sequential processing of instructions listed in programs (computers in so-called von-Neumann architecture) were developed in the 1940s based on the investigation of sequentially conscious thinking.

On the other hand, there are the much more complex structures of unconscious thinking or unconscious intelligence. Here, a lot of environment data are processed within the context of our sensory perception and characteristics extracted. The sensorimotor control of our motions as well as three-dimensional thinking are largely unconscious. The structures of unconscious thinking provide the basis for the enormous capacity of our memory. All of these functions performed unconsciously are running parallel in networks in which so-called neurons interact due to a close interconnection and by means of electrochemical processes.

Our brain is organized as highly integrated system in functional units, which are interconnected via variable connections, with each functional unit having about one thousand to one hundred thousand nerve cells. These each have ten to ten thousand equally variable, so-called synaptic connections to other neurons. In total, our central nervous system roughly contains the astronomical number of one hundred to one thousand billion nerve cells. It is clear that this enormous information-processing system cannot be completely structured and programmed prenatally even if genetic information is taken into account. The brain has the capability to organize itself, learn and establish associations.

To imitate biological information processing models for different levels of organisation and of abstraction have to be considered. First, there is the level of the individual neuron where it is a matter of representing the static and dynamic electrical characteristics as well as the adaptive behavior of the neuron. On the network level the interconnection of identical neurons to form networks is examined to describe specific sensor- and motoricity-related functions such as filtering, projection operations, controller functions in nonlinear, biological systems. Networks on the mental function level are the most complicated ones and comprise functions such as perception, solution of problems, strategic proceeding etc. These are the networks on the highest level of biological information processing.

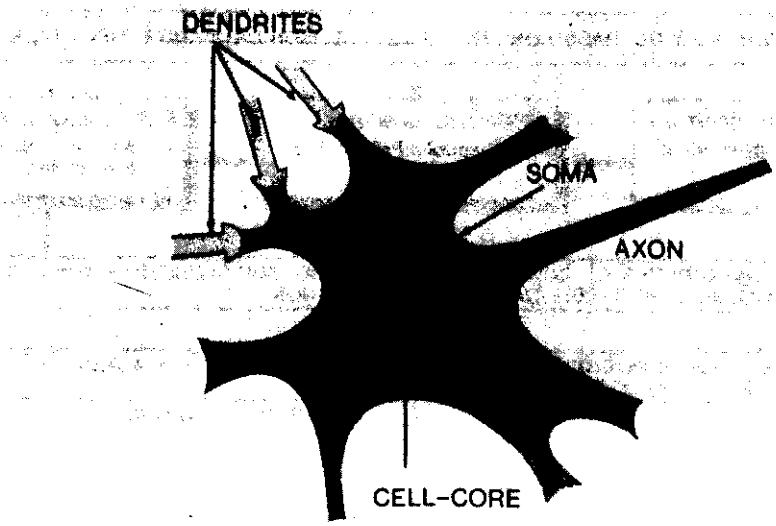
The Neuron

The nerve cell (the neuron) comprises the cell body (soma) which surrounds the cell nucleus (Fig. 4). The cell body has a long processus, the axon (or neurite) which ends in numerous ramifications which are attached to other cells via so-called synaptic end heads thus forming the synaptic connections between neurons. The synaptic connection is to points where the cell body is expanded to so-called dendrites.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

In the stationary electro-chemical state the cell has a resting potential of about - 80 mV. If a nerve cell is stimulated by another cell via the synaptic connection, a short-time pole reversal of about 30 millivolts with a duration of 1 millisecond results (Fig. 5). This so-called action or activation potential moves with up to 100 meters per second across the axon to neighboring cells. The stimulus must exceed a specific threshold so that this action potential is generated. The action potential immediately drops after its rise and the cell returns to the resting state.

The degree of cell stimulation, i.e. the density of information is determined by the frequency of the action potentials. The greater the stimulus, the higher the sequence of impulses on the axon.



SIMPLIFIED BIOLOGICAL NEURON

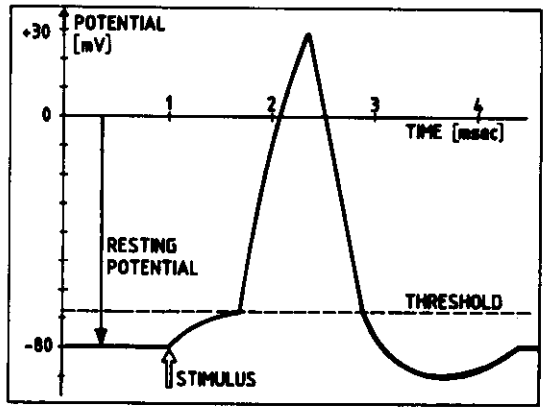
FIGURE 4

What is very important for the learning and adaptive capabilities of biological neural networks is the so-called plasticity characteristic of the synapses. This characteristic gives the neurons a memory such that their reaction to an impulse received depends on their past history, i.e., for instance, how many impulses have already been transmitted by it before and in which sequence. In this process, the past history is taken into account over minutes, hours, yes even over much longer periods of time (long-term memory).

Apart from the stimulating neurons there are also inhibitive neurons. These produce transmitters which increase the negative charge in the interior and thus the resting potential of the receiving cell. These inhibitive neurons can blank out action potentials of stimulating neurons, which are transmitting simultaneously, in the joint receiving neuron. Hence, all potentials received via synaptic connections are added on the receiving neuron; those from stimulating neurons with a plus sign and those from inhibitive neurons with a minus sign. The sum of all inputs triggers the neuron activation via a nonlinear activation function.

EXCITATION PATTERN IN RESPONSE OF A STIMULUS (ACTIVATION POTENTIAL)

FIGURE 5



What is remarkable in this connection:

Today's digital computers have cycle times (time for processing a partial information) of 4 to 5 nanoseconds. The comparable cycle time of a neuron (time for processing a stimulus up to readiness for receiving a new stimulus) is 4 to 5 milliseconds. Thus the digital computer is a million times faster than the neuron. Despite this enormous difference in the time for processing a piece of information and for reacting to a stimulus neural networks are in many applications superior to digital computers with sequential processing of often extensive programs regarding the execution time due to their parallel information processing.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

4. The Artificial Neuron

On the basis of the biological neuron a simplified model of the artificial neuron is shown in Fig. 6. According to this figure, the neuron acts like an integrator with feedback which integrates the weighted presynaptic input signals and maps the postsynaptic activation x_j onto the output signal by means of one of the nonlinear functions shown as examples. A threshold value b_j is also taken into account. The differential equation for the activation x_j of the j-th neuron taking into consideration the learned connecting weights w_{ji} describes the short-term behavior of the neuron (short-term memory, STM) as a function of the input signals s_i; i = 1,2...n. In this equation the inhibitory inputs are taken into account by the term h_j (see fig. 6). Because activation is a nonnegative entity the condition $x_j \geq = 0$ must be imposed.

The ability to learn and memorize something owing to the plasticitiy property of the biologic neuron is in the case of the artificial neuron obtained by adaptation of the connecting weights. As a consequence, the short-term behavior of an element is made dependent on its case-history (long-term memory, LTM).

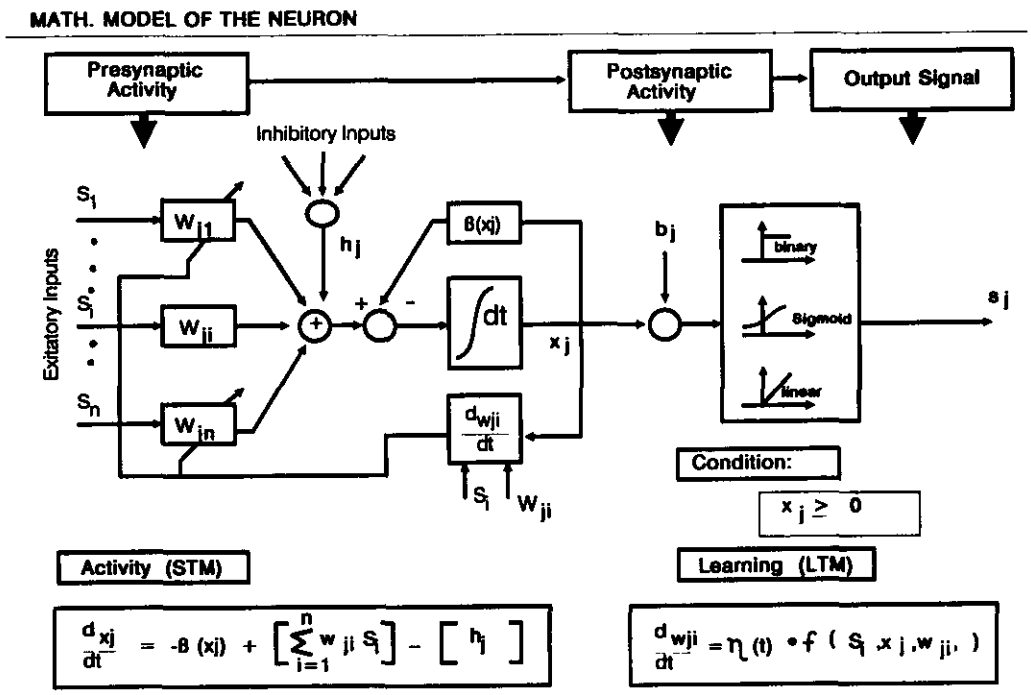


FIGURE 6

The differential equation for the adaptation of the connecting weights as shown in Fig. 6 describes the dynamics of learning as a function of the instantaneous values of the connecting weights (input weights), the activation and the input quantity. For controlling the learning speed the function $\eta(t)$ is also introduced. The connecting weight leading from the input i to the j-th neuron is called w_{ji}.

Depending on the particular form of the x_j- and w_{ji}-equations there are different neural and network models (paradigms). A partitioning into excitatory and inhibitory input signals is, however, not necessarily required if the former are considered to be positive and the latter negative input signals and the activation x_j can also assume negative values.

A simplification of the neural model according to Fig. 6 considers the stationary activation status and is shown in Fig. 7. It was introduced by McCulloch and Pitts (1943). The resulting output signal is

$$s_j = f\left(\sum_i s_i w_{ji} - b_j\right) \qquad (1)$$

Instead of substracting the threshold b_j from the sum of the weighted input signals, it can be interpreted as an additional weight w_{jo} with a constant input "1" such that the activation equation becomes

$$x_j = \sum_i s_i w_{ji} + w_{jo} \qquad (2)$$

Based on this equation the artificial neuron can be represented as a basic processor element (PE) as shown in Fig. 8. It is remarkable that the summation of the weighted input signals is mathematically identical with the scalar product of the input and weight vector. Geometrically it is thus a measure for the correlation between the input vector $\underline{s}$ and the instantaneous weight vector $\underline{w}_j$ of the j-th PE, as shown in the following equation:

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

$$\sum_{i=1}^{n} s_i w_{ji} = \langle \underline{s}, \underline{w}_j \rangle = || \underline{s} || \cdot || \underline{w}_j || \cdot \cos (\underline{s}, \underline{w}_j) \qquad (3)$$

Therefore the PE can be imagined to perform a vector pattern matching operation. Equation (3) can be looked at as the fundamental equation of all adaptive networks.

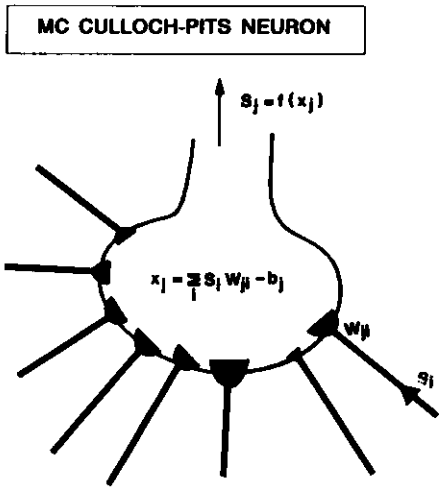


FIGURE 7

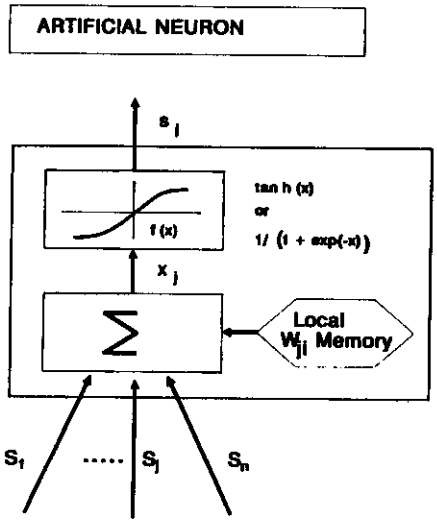


FIGURE 8

The PE according to Fig. 8 has two properties representing important preconditions for the arrangement in parallel network structures: Only local inputs and local w_{ji} memory (i.e. no other information from a network is required); only one output signal is produced which is propagated to other PEs or represents an output element of a network.

Possibly there are a large number of potential nonlinear output functions (threshold functions) The signum and sigmoid function, hyperbolic tangent, binary step function, saturated ramp function are employed by the majority of PE types.

From a neurophysiological point of view the artificial neuron as described here is too simple. For the designer of advanced adaptive systems this is, however, no restriction. Utilizing important characteristic features of the biological paradigm like interconnection of a large number of simple PEs with parallel processing and adaptible connection weights as well as nonlinear output functions yields the possibility for the design of processing units with unprecedented capabilities.

5. Adapative Processing Elements

The Adaptive Linear Combiner (ALC) is known from adaptive signal processing. In Fig. 9 is $\underline{x}_k = [x_{ok} x_{1k} ... x_{lk}]^T$ the input vector and $\underline{w}_k = [w_{ok} w_{1k} ... w_{lk}]^T$ the weight vector of the ALC at time t_k. The output quantity as the sum of the weighted input quantities is

$$y_k = \sum_{i=o}^{L} w_{ik} \cdot x_{ik} = \langle \underline{w}_k, \underline{x}_k \rangle = \underline{w}_k^T \underline{x}_k \qquad (4)$$

The same relationship applies for a Finite Impulse Response (FIR) filter if $\underline{x}_k = [x_k x_{k-1} ... x_{k-l+1}]^T$ is set there, x_{k-i} with i = 1,2...(l-1) being the filter input quantities delayed by one clock cycle each (delay operator z^{-1}).

The weights are adapted by means of the LMS (least mean squares) algorithm which minimizes the square of the deviation of the output quantity y_k from the desired output quantity d_k which is considered to be known. For this deviation and its square the following equations apply:

$$e_k = d_k - \underline{x}_k^T \underline{w}_k \qquad (5)$$

$$e_k^2 = d_k^2 - 2 d_k \underline{x}_k^T \underline{w}_k + \underline{w}_k^T \underline{x}_k \underline{x}_k^T \underline{w}_k \qquad (6)$$

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

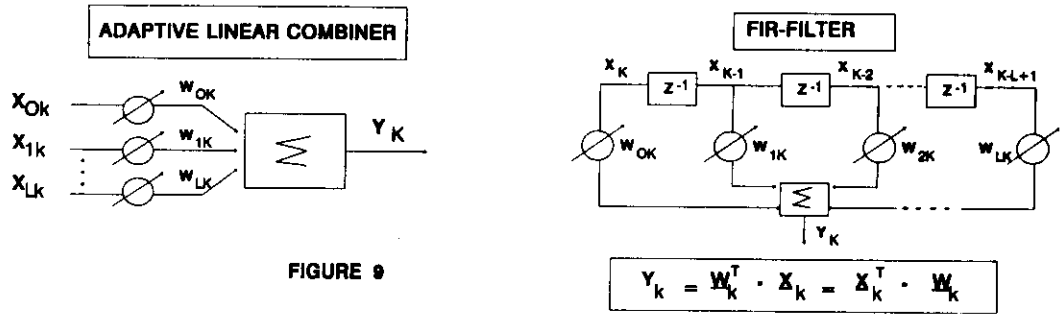


FIGURE 9

At each iteration in the adaptive process the gradient estimate becomes

$$\hat{\nabla}_k = -2\, e_k\, x_k = -2\,(d_k - y_k)\, x_k \qquad (7)$$

With this simple gradient estimate the LMS algorithm is of steepest descent type by updating the weight vector according to

$$w_{k+1} = w_k - \mu \cdot \hat{\nabla}_k = w_k + 2\mu\,(d_k - y_k) \cdot x_k \qquad (8)$$

The steepest descent step size parameter μ regulates speed and stability of adaptation. Adaptive signal processing based on the ALC with LMS adaptation has been successfully applied in systems identification, adaptive noise cancelling, adaptive prediction and others.

The Adaptive Linear Neuron (ADALINE) as the simplest nonlinear processing element is closely related to the ALC. As shown in Fig. 10 it utilizes the signum output function. Since the output signal from the summation is used for the error determination needed for adaptation of the input weights, the LMS algorithm can be used here, as well.

The structure of the perceptron which is also shown in Fig. 10 is identical with that of the ADALINE. The only difference is that the PERCEPTRON convergence algorithm uses the output signal s_k for error recognition for the weight adaptation. In both cases η is introduced to control the adaptation/learning rate.

Because of the nonlinear output function the ADALINE and PERCEPTRON become capable of input signal classification. They are capable to recognize whether a particular input pattern belongs to a corresponding class or not. The classifying ability of the PERCEPTRON is illustrated in Fig. 11. For this purpose a simple element with 2 inputs and 1 output is investigated. Assuming w_i = const. after completion of learning. The classification equation in this case describes a straight line in the input signal plane (S_1 - S_2 plane). This line separates the two classes. If the PERCEPTRON has more than two inputs the straight separating line changes into a plan (3 inputs) or to a hyperplane (> 3 inputs). The PERCEPTRON adaptation algorithm converges when classes can be separated linearly. In practice, this is frequently not the case or not known a priori. Then, the arrangement of simple processor elements (e.g. PERCEPTRON) in multi-layer networks is required.

Thus, the transition from the individual adaptive element to the arrangement of such elements to form artificial neural networks becomes necessary.

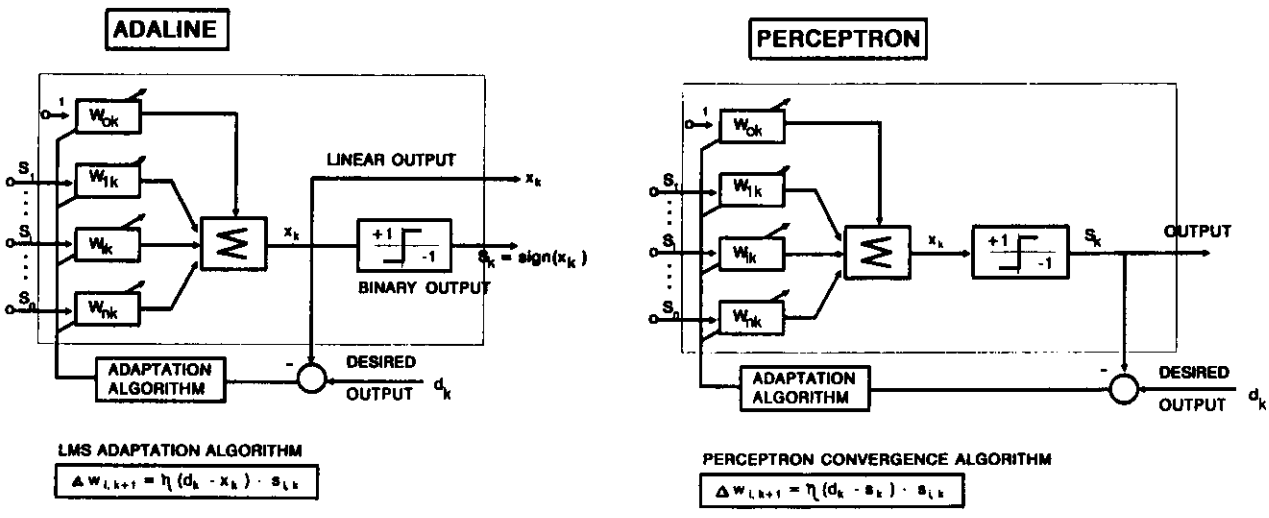
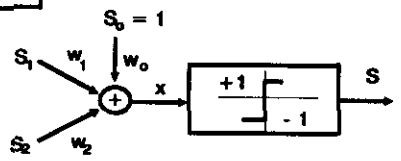


FIGURE 10

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

PERCEPTRON CLASSIFICATION PROCESS

TWO INPUT PERCEPTRON



ACTIVITY

$$x = w_o + w_1 S_1 + w_2 S_2$$

OUTPUT

$$S(x) = \begin{cases} 1 & \text{IF } x \geq 0 \\ -1 & \text{IF } x < 0 \end{cases}$$

DEFINITION OF CLASSES

INPUT PATTERN ϵ (CLASS A) IF $S(x) = 1$

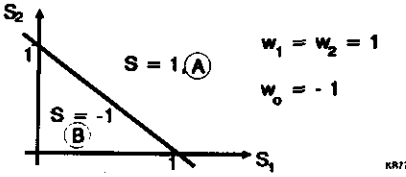
INPUT PATTERN ϵ (CLASS B) IF $S(x) = -1$

SEPARATION OF CLASSES

$$x = w_o + w_1 S_1 + w_2 S_2 = 0$$

$$S_2 = -\frac{w_1}{w_2} S_1 - \frac{w_o}{w_2}$$

CLASSIFICATION AREAS



KR77/90

FIGURE 11

6. Artificial Neural Nets (ANN)

Definition

According to a definition used in literature, the term artificial neural net (ANN) means massively parallel connected arrangements of simple elements adaptive in general (but not necessarily) with hierarchically organized structure from which it is expected that they interact with the objects of the real world in a similar way as the networks in biological systems do. ANN are accordingly structured from groups of simple processor elements that are arranged in layers (Fig. 12). Each layer comprises a certain quantity of PEs that communicate via connections with different adaptable weights. There are intra- and inter-layer connections.

For the structure of ANN, three basic elements are important: organized topology of interconnected cells (PEs), method of encoding (learning) information, method of recalling information. They are dealt with briefly in the following.

Artificial Neural Network

NETWORK

PROCESSING ELEMENT

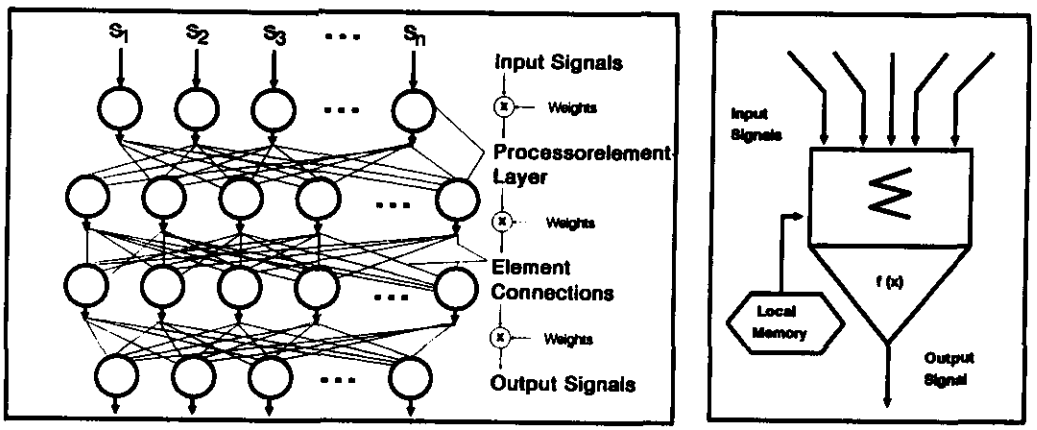


FIGURE 12

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

ANN, a nonlinear system

The ANN dynamics can be described by a set of nonlinear differential equations for an autonomous dissipative dynamic system.

$$\dot{x}_j = \frac{dx_j}{dt} = f_j(x_1, x_2, \ldots x_n) \qquad (9)$$

with x_j as real variable, i.e. neuron activation. Since the f_j do not explicitly depend on time, the system is said to be autonomous.

With dissipative systems, the flow in the phase-space characterized by the field of velocity vectors is

$$\dot{\vec{x}}(t) = [\dot{x}_1\ \dot{x}_2 \ldots \dot{x}_n]^T = [f_1\ f_2 \ldots f_n]^T \qquad (10)$$

contracting, i.e. $dv/dt < 0$

The volume contraction can be computed directly from the differential equations (9) without knowing the solution as follows

$$\frac{dv}{dt} = \int_V d^n x\ \mathrm{DIV}(\dot{\vec{x}}) \qquad (11)$$

with

$$\mathrm{DIV}(\dot{\vec{x}}) = \frac{\partial \dot{x}_1}{\partial x_1} + \ldots \frac{\partial \dot{x}_n}{\partial x_n} = \frac{\partial f_1}{\partial x_1} + \ldots \frac{\partial f_n}{\partial x_n} \qquad (12)$$

On account of $dv/dt < 0$ the volume element is mapped onto a subspace of the phase space asymptotically with the volume zero. This subspace is a so-called attractor.

There are two kinds of attractors: periodical attractors as asymptotically stable limit cycles and asymtotically stable fix points as attractors which are primarily of interest to stable ANN. These asymptotic solutions (fix points for $t \to \infty$) do not depend on the initial conditions. Moreover, the type of behaviour of a general non-linear system, whether stable, unstable, oscillatory or chaotic, depends critically on the input applied to it. For ANN, those nonlinear structures are therefore suitable that achieve asymptotic stable fix points (attractors) for a large range of input patterns i.e. large input signal space (input vector space). In these fix points, the knowledge contained in the input patterns can be stored, or the input signals can be classified. This is accomplished by modification of f_j during the learning (training) process of the ANN.Upon completion of learning the w_{ij} are fixed. In the recall mode the system acts as a short term memory (STM) dynamic system; i.e. a content addressable memory (CAM). Based on this brief representation of non-linear system behaviour, the network operation can be summarized as delineated in Fig. 13.

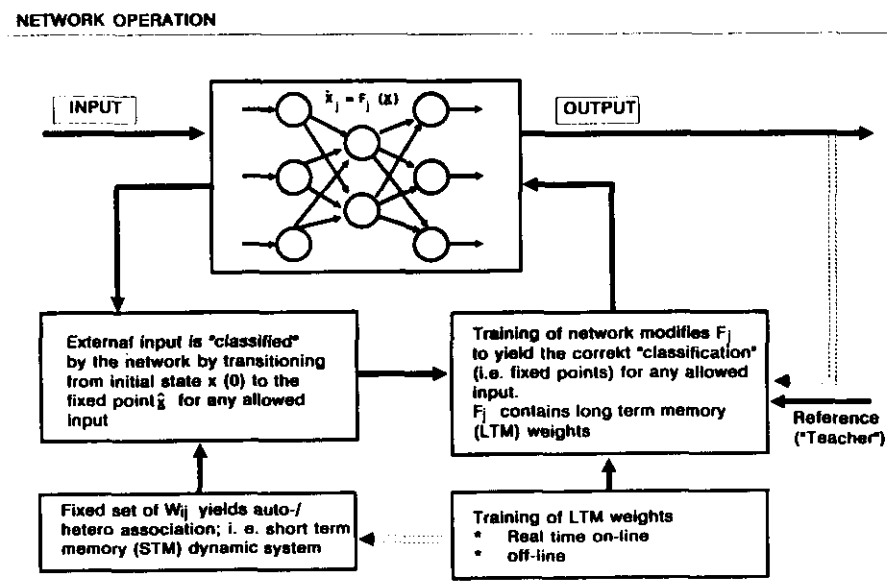


FIGURE 13

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Applying the direct method of Lyapunov, different theorems can be utilized to prove stability for ANN with feedback connections. A general design principle for absolutely stable information processing and memory storage by nonlinear feedback networks is the Cohen-Grossberg theorem with the j-th PE activation dynamics as shown in Fig. 14. For this class of nonlinear feedback systems two important cases for the PE's activation dynamics can be discerned: the so-called additive and shunting short-term memory equations as shown in fig. 15. These equations respresent the basis for the design and analysis of a number of specialized networks as applied for particular problems. An example for a stable network structure belonging to the Cohen-Grossberg class is one with self-exciting recurrent connections and neighbour-inhibiting ones, the so-called competitive systems as shown in fig. 16.

STABILITY THEOREMS

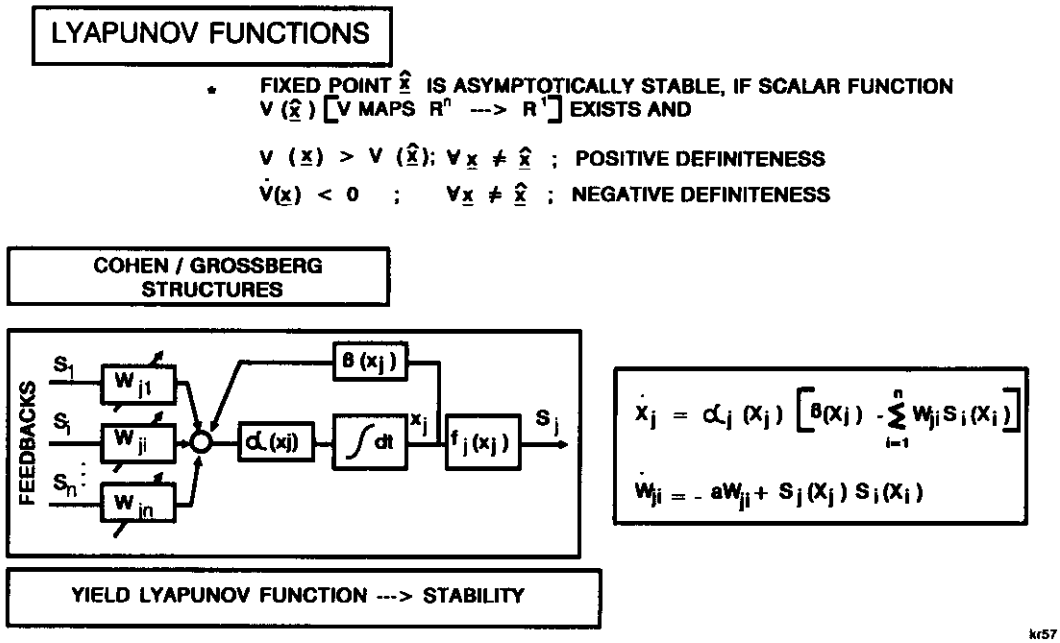


FIGURE 14

COHEN-GROSSBERG THEOREM
ABSOLUTELY STABLE NONLINEAR FEEDBACK SYSTEMS STM EQUATIONS

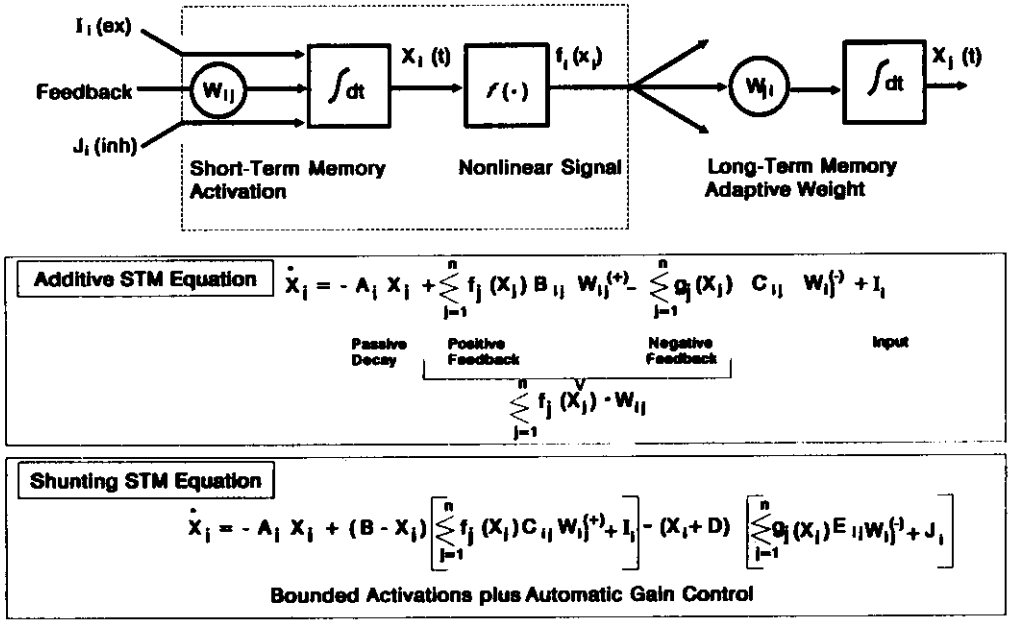


FIGURE 15

Furthermore, ANN are realized in structures which show only feedforward connections. These are inherently stable if they comprise stable single elements which is achievable by a corresponding output function.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

COMPETETIVE SYSTEMS

- NETWORK STRUCTURES WITH SELFEXCITING RECURRENT CONNECTIONS AND NEIGHBOUR-INHIBITING CONNECTIONS

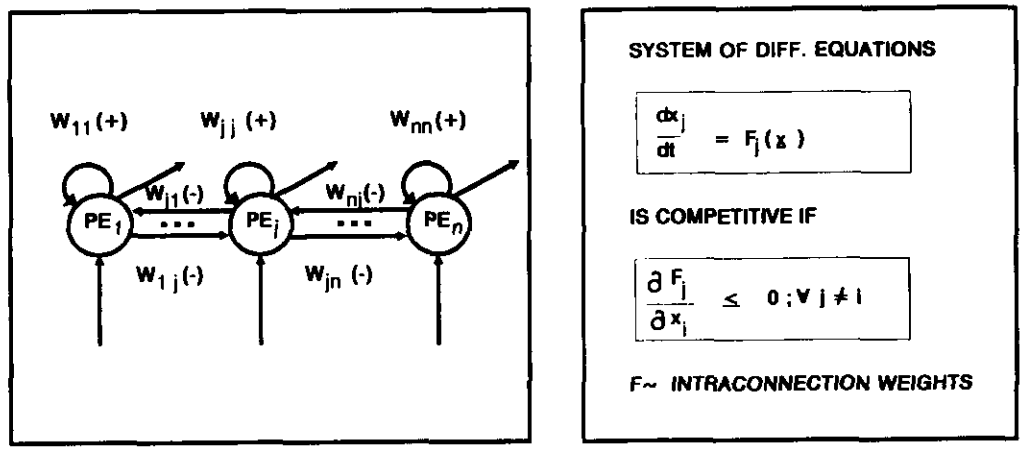


FIGURE 16

In addition to global stability, convergence of a network plays an important role. The stability problem occurs in the recall phase of ANN with feedback connections. The convergence concerns the minimization of the error between the desired and the computed ANN output signal. For this reason, the convergence is of importance in supervised learning and must be specially verified for each corresponding ANN model and the appropriate learning strategy.

It shall be mentioned that there are ANN applications which demand periodical attractors. The corresponding ANN are trained for stable limit cycle oscillations.

Learning, Self-Organisation (Encoding)

Contrary to the conventional proceeding in which the solution of a problem must be available in form of an algorithm, the way to solve a definite task is self-organized and learned in the case an ANN is used. The ANNs with hard-wired encoding are an exception to this. In their case, the knowledge of the problem and its solution is practically implemented by prenatal determination of the topology as well as the strength of the connections by the designing expert.

In the case of self-organization, the neural net forms an internal cognitive model of the task and thus replaces the mathematical description. This is done by generating the suitable meshing and weights. The problem here is the determination of the modification strategy which leads directly to the problem of learning.

As mentioned already, in case of a massively parallel net (Fig. 12) the knowledge lies with the way of linking the elements (PE) as well as with the strength of the linkings (interconnection weights). If learning is understood as a modification of the knowledge, the network interconnections can be changed in three ways: generation of new interconnections, loss of existing interconnections and change of existing interconnection weights. From adaptive signal processing, parameter and structure-adaptive filter structures are known. Regarding ANNs the procedure is that so far only the weight factors of given ANN structures are modified. By the interconnection weight zero, an interconnection can be interrupted (acts like a structural change) or conversely its efficiency can be increased by increasing the weight factor.

As shown in Fig. 17, supervised and unsupervised learning (encoding) can be discerned. The supervised learning by error backcoupling has already been explained in chapter 5 when dealing with simple adaptive processing elements. For multi-layer nets, the method of error backcoupling fails. Here, the so-called back propagation algorithm must be used for the supervised learning since for the PEs on hidden layers the desired output cannot be classified. In the example as treated in chapter 9 a back propagation ANN is considered in more detail. The reinforced learning as mentioned in Fig. 17 is looked at again in chapter 8 when dealing with the neuro control problem.

If no predefined training data are available or if their generation is too time and cost consuming, self-organizing nets must be utilized that learn unsupervisedly. Based on local information and internal ANN control, the net self-organizes the presented data and discovers its emergent collective properties. Unsupervised Hebbian learning (Donald Hebb, 1949) is important to many ANN designs. From Fig. 18 it becomes evident that the Hebbian learning rule computes a correlation between the presynaptic signal (s_i) and the postsynaptic activation (x_j) where a positive correlation ($x_j s_i > 0$) is causing a weight increase. Also a passive decay term ($- \alpha w_{ij}$) is often added in the $\dot{w}_{ij}$ equation. In each case, only local information is required as compared to error backcoupling or error back propagation, i.e. the presynaptic signal on the input path, the postsynaptic activation of the PE and possibly the actual interconnection weight value. In many cases, the output signal $s_j(x_j)$ is used instead of the postsynaptic activation. For the class of Cohen-Grossberg structures as mentioned before, the so-called passive decay and gated decay unsupervised learning equation for the long-term memory weight adaptation can be utilized (Fig. 19).

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

LONG-TERM-MEMORY LEARNING TAXONOMY

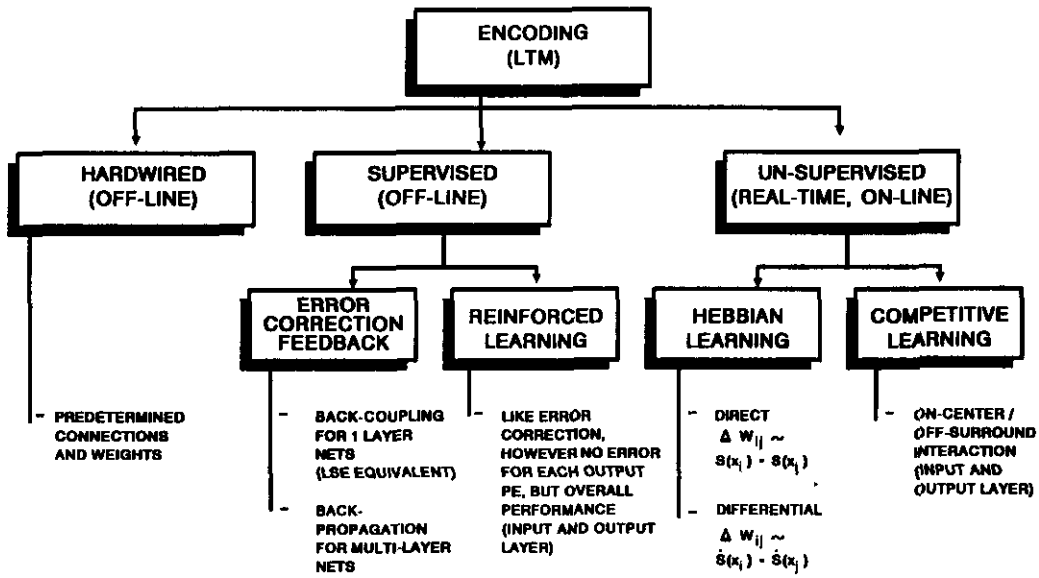


FIGURE 17

UNSUPERVISED HEBBIAN LEARNING (SELF-ORGANIZATION)

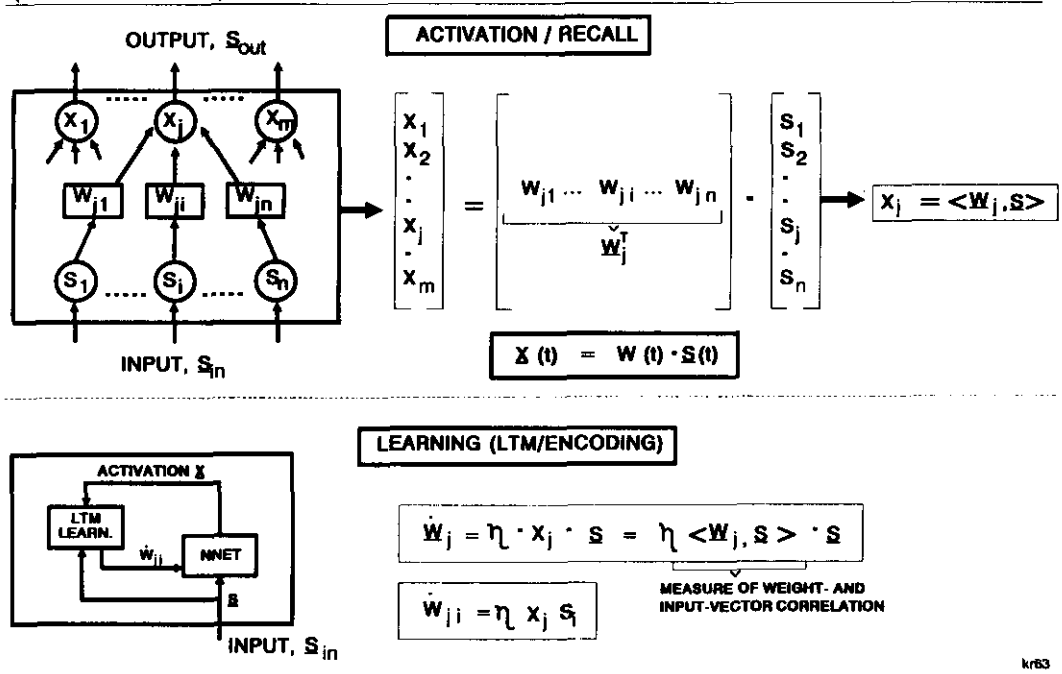
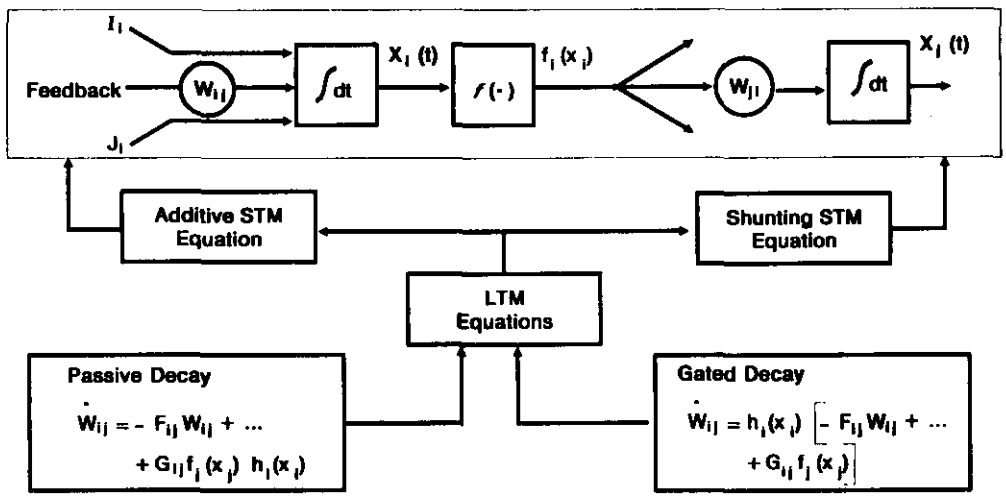


FIGURE 18

Different learning paradigms are making use of the so-called competitive learning. In its simplest version, competitive learning works in combination with recall as shown in Fig. 20 (off-line, unsupervised). The weight vector $\underline{w}_j$ that matches best with the input vector will yield the highest activation of the associated PE. This is called the winning PE and only its input weight vector ($\underline{w}_j$) - and none of the others - is adjusted in proportion to its euclidean distance $\underline{d}_j$ from the input vector ($\underline{u}$). In an extension the output layer PEs can compete with each other intra-layer by sending positive feedback signals to itself (recurrent self-excitation) and negative signals to all its neighbours (lateral neighbour inhibition). This type of connection was already shown in Fig. 16 when mentioning competitive systems.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM



A general design frame for absolutely stable information processing and memory storage (CAM) by nonlinear competitive-cooperative feedback networks

FIGURE 19

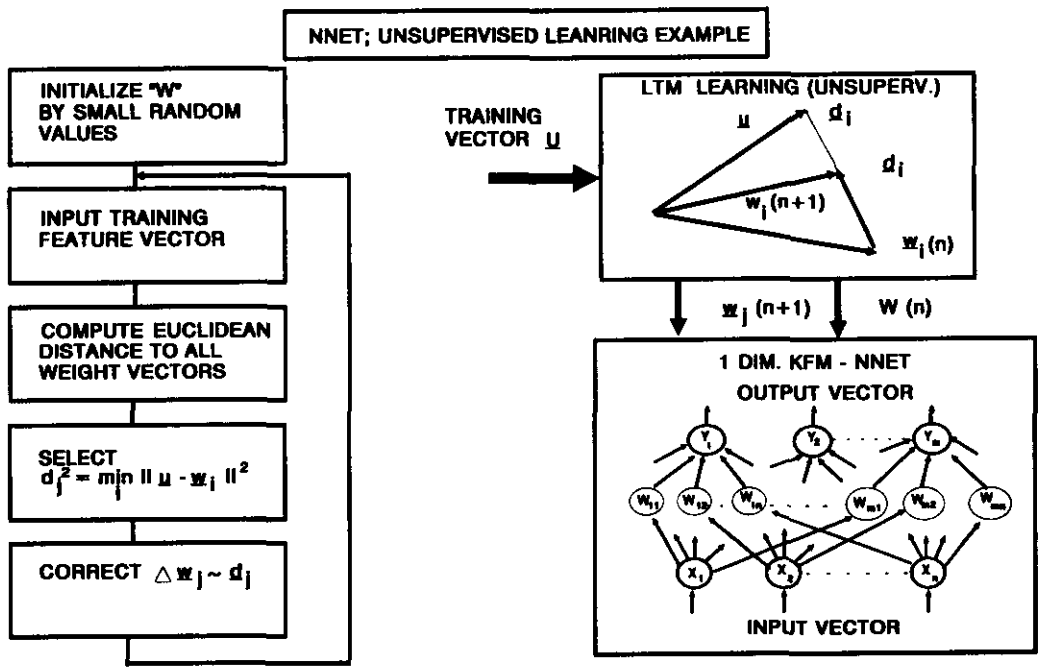


FIGURE 20

Important Self-organizing ANN models

a) Self-organizing Feature Map (SOFM)

The so-called self-organizing feature map introduced by Kohonen consists of a one- or two-dimensional arrangement of the PEs. The structure is completely meshed and processes real-valued input signals. The PEs simply form the sum of their weighted inputs. The modification of the interconnection weights is made according to the previously mentioned method of competitive learning (Fig. 20). However, in addition to the weight vector of the winner PE ($\underline{w}_j$), the weight vectors of a predetermined neighbourhood of cells in the area N_j are also modified in this case. This area is reduced with increasing training time.

In Fig. 21 a portion of a one-dimensional SOFM network with a two-dimensional input-(feature) space is shown (left). When an input vector is presented the winning network node (PE) is identified corresponding to the minimum euclidean distance between input and weight vector of that node (see also Fig. 20). The weight vector of the winner (closest to input vector) and those of its neighbours, regardless of their values, are updated to learn the current input by moving closer to its position (Fig. 21, right).

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

If the training vectors form individual boundaries in the feature space, the weight vectors will have adapted themselves after a sufficient quantity of learning steps in such a way that they represent corresponding classes. I.e. topologically close processing elements correspond to physically adjacent groups of input vectors (classes) that is, SOFM's can compute the probability density function (PDF) of input vectors and represent it implicitely by their own density.

Learning of SOFM is unsupervised but learning phase and application phase are separated from each other. Thus, problems can arise during the application phase, in the case of a slow change of the input data or the occurrence of data not learned by the net.

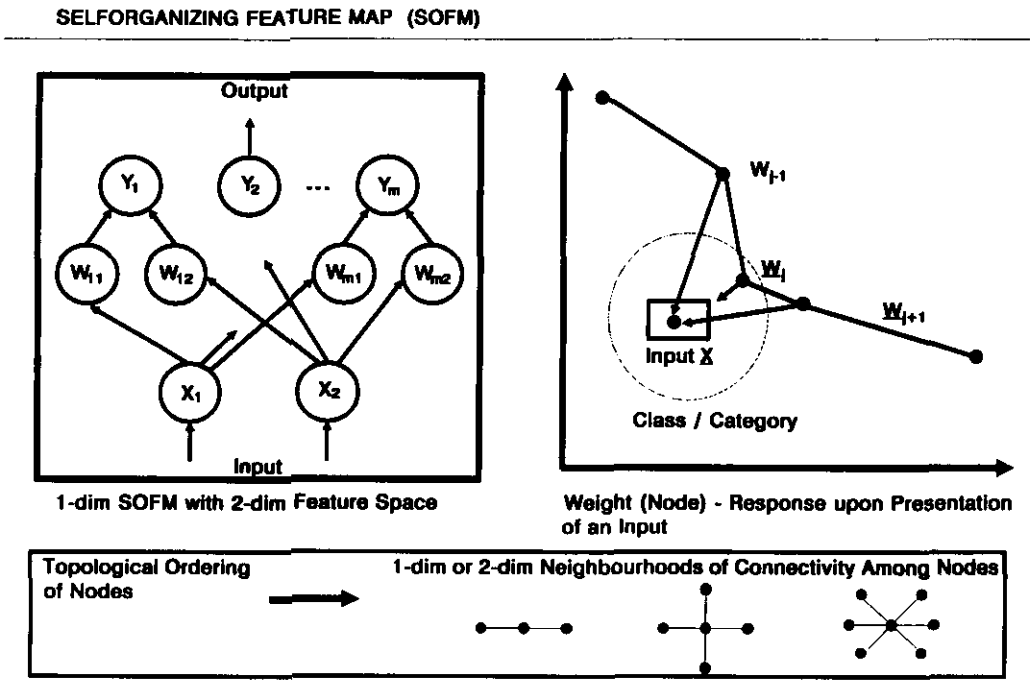


FIGURE 21

b) Adaptive Resonance Theory (ART)

To mimic cognitive functions autonomous self-organizing systems require means that are capable of learning, memory and recognition in an unpredictable world with no teacher available. Corresponding computational units must continue to learn in a stable fashion where this new learning must not force unselective forgetting of past acquired knowledge.

Grossberg and Carpenter (1987) designed the so-called ART network (Adaptive Resonance Theory) in order to solve the dilemma between stability of the learned knowledge and the plasticity I.e. the capability of continued learning. ART networks are stable enough to preserve significant post-acquired knowledge but nevertheless remain adaptable enough to incorporate new information whenever it might appear. The basic idea which lead to the ART was the discovery that a 3-layer net (Fig. 22) with competitive learning can perform any mapping from input (feature) space R^m to output (category, class) space R^n. The ART-net can be imagined as a two-layer structure resulting from folding back the three-level network on itself as shown in Fig. 22. Thus the simple ART module includes a bottom-up competitive learning stage in combination with a top-down outstar system, both representing adaptive filters with associated LTM weights.

The main function of the ART is that the top-down attentive feedback encodes learned expectations (learned bottom-up) in response to arbitrary temporal sequences of spatial input patterns in real time. A large enough mismatch at level F_1 quickly resets the F_2 code before new learning can occur by triggering the orienting signal (Fig. 22). The F_2 code is reset if the degree of match is smaller than a predetermined vigilance parameter. In this basic configuration of the ART, stored patterns can be permanently updated on the one hand and on the other hand, additional pattern classes in the net can be generated if the input pattern has no similarity to existing pattern classes.

The different methods of learning and self-organisation possess particular important characteristics (Fig. 23) which have to be considered when selecting the appropriate network for a given application. In Fig. 23 ARTMAP is a new architecture using multiple ARTs in a network hierarchy with supervised associative learning. Also the Vector Associative Map (VAM)Network is a new design for fast unsupervised realtime error-based learning. It might play an important role in sensory motor control type problems. In its key features it is complementary to the ART net.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

ART 1 NETWORK

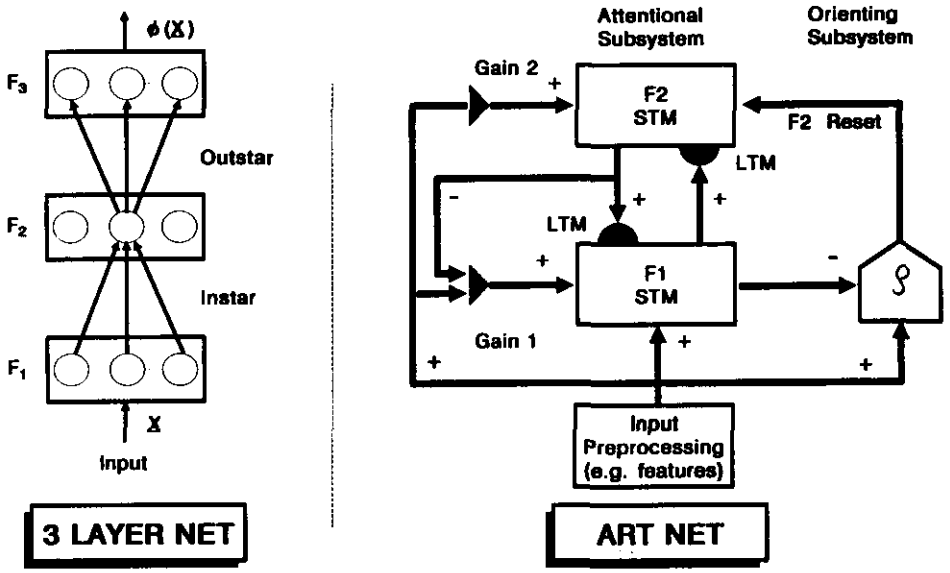


FIGURE 22

FUNDAMENTAL LEARNING FEATURES
DETERMINE CLASSES OF APPLICATION

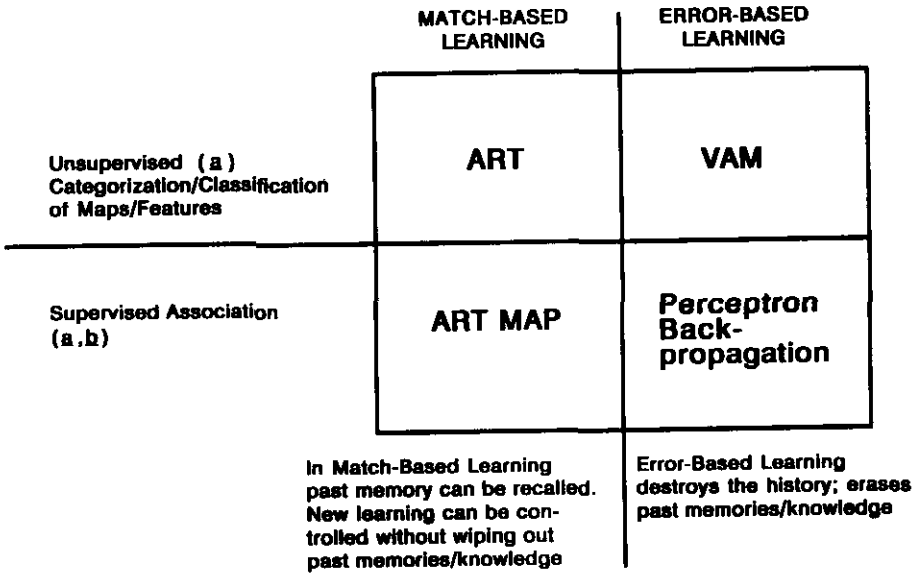


FIGURE 23

ANN Recall Operation

In the preceding chapter, the gathering and storage of information and knowledge in an ANN was treated with some detail. The recall concerns the retrieval of information stored in ANN; i.e. the STM function. The recall procedure is in general given by the solution of an activation equation in connection with a particular output function as shown in general form in Fig. 6. Similarly as for learning, there are some basic recall paradigms, from which for Cohen-Grossberg structures the additive and the shunting STM equations were given in Fig. 15.

Basic functions of trained ANN

As already mentioned ANNs are actually content addressable memories (CAM) which either recall stored information or encode new input information self-contained or supervised by a teacher. Applying ANNs the following basic functions can be performed (see also Fig. 13):

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Auto-association:	With a noisy or incomplete input pattern (vector), the undisturbed complete input pattern is generated as output.
Hetero-association:	Input pattern and associated output pattern are different.
Classification:	Each input pattern is assigned to one of several classes which are defined by the output pattern.

In all three cases, the storage of associative mappings is concerned. There are many applications where system elements must be described by such stimulus-response type mappings including such as linear, nonlinear, logical or binary ones.

ANN Summary

While Fig. 13 shows the operational processes, the main features are resummarized in the following: Artificial neural nets

- are computers that learn how to solve problems
- problem solving is based on sample data and learning mechanisms
- they do not require expert knowledge representation, logical inferencing schemes, statistical algorithms or specialist/analyst to develop and code a solution
- they are trained to identify self-containedly the key features and associations enabling them to distinguish different patterns
- can learn on-line real-time or can be trained off-line by a sample data set
- do require an appropriate architecture with sufficient capacity and paradigmatic learning/training scheme
- they consist of three major elements: organized topology of interconnected processing elements, method of encoding information, method of recalling information.

Their strengths and weaknesses are summarized as follows:

Strength:

- unique solutions based on user data examples
- no need to know algorithms
- less/no software needed, more hardware-processing power required
- provides solutions to problems such as: pattern matching and recognition, data compression, near-optimal solutions to optimization problems, non-linear system modelling and control, function approximation etc.
- inherent parallel processing structure yields faster solutions to a number of computation-intensive problems
- internal generation of complex decision areas by means of non-linear combination of input vector components
- robust performance in view of noisy and disturbed input signals
- inherently fault-tolerant

ANN weaknesses are that they are not applicable to all processing problems and do require training and test data examples - with a few exceptions.

A comparison of ANN with conventional digital computers is summarized in table 1. This leads directly to some remarks regarding the utilization of ANNs.

COMPARISON CONV. DIGITAL vs. NEURAL PROCESSING

TABLE 1

FEATURE	DIGITAL COMPUTER	NEURAL PROCESSING
Processing order	- Programs with serially performed instructions	- Parallel programs with comparatively few steps
Knowledge storage	Static copy of knowledge is stored in addressed memory location - New information destroys old information	- Information stored in the interconnection of neurons - Knowledge adapted by changing interconnection strengt
Processing control	Central processing unit monitors all activities and has access to global information, creating processing bottleneck and critical point of failure	- No control nor monitoring of a neuron's activity - Neuron's output only a function of its locally available information from interconnected neighbours
FAULT TOLERANCE	- Removal of any processing component leads to a defect - Corruption of memory is irretrival, leads to a failure	- Distributed knowledge/information representation across many neurons and their interconnection - If portion of neurons removed, information retained through redundant distributed encoding DISTRIBUTED ENCODING
	⟶ FAULT-INTOLERANT	⟶ FAULT-TOLERANT

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

General remarks for ANN application

Concerning the potential application it can be said that in many problems with only small or almost no knowledge existing on the object concerned, or where the parameters and states of this object can neither be described mathematically nor by rules and facts in a somehow reliable manner, the development of sequential algorithms for conventional processors is extremely difficult. The necessary expenditure of cost and time for the algorithm and software development, verification and validation is correspondingly high.

Contrary to the sequential conventional information processing, the processing of information utilizing neural nets offers in general considerable advantages for all applications which are characterized by limited knowledge on the object. In contrast with the programmed sequential computing, ANN can be applied successfully for the solution of problems with inexact and incomplete or even contradictory input data.

The ability of neural nets to learn by examples (training patterns) or even unsupervised is of particular importance. It is not necessary to program a task-specific function or information. If representative example data are available in sufficient number and by training of the net with these data, due to its generalization property the net can tolerate input data which are superimposed by noise and disturbances, for the recognition of the input patterns.

By the use of non-linear processing elements in the network, multi-level nets can form complex decision areas in the feature space. This corresponds mathematically to a non-linear mapping of the input vector space onto that of the output vector. This allows also the *modelling of non-linear systems.*

G.a.C. Applications of Artificial Intelligence and Neural Computing

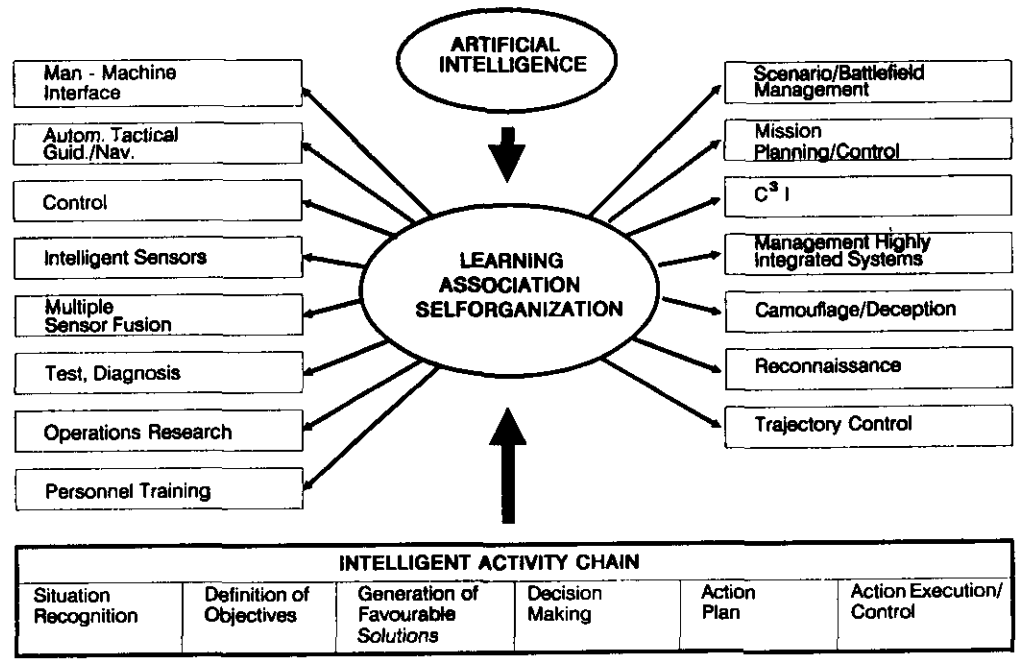


FIGURE 24

7. Categories for ANN application

It has already been mentioned (chapter 2) that *G.a.C. problems extend over several hierarchically structured levels.* In order to perform G.a.C. functions on these levels, the implementation of a controlling feedback chain typical of all G.a.C. levels is required. Any technical apparatus which implements the feedback chain in a real-time autonomous system requires the solution of perception problems as associated with the sensors and cognition problems (e.g. recognition, hypothesis testing) as far as the remaining functions are concerned. Very often in such systems, exploratory, goal-oriented actions will be performed resulting in a perception-cognition-action-recognition cycle.

It has been shown (Fig. 3) that for the implementation of such quasi mental functions elements of artificial intelligence are required. In addition to more conventional expert system techniques ANN will gain an increasing importance within this scope. Therefore, as shown in Fig. 24, the application potential for ANN covers many areas, extending from relatively simple applications in intelligent sensory and actuator systems to highly complex mission and scenario management problems.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

Areas which represent potential categories for successful ANN application and which are recurring in many G.a.C. systems are the following:

- pattern recognition, signal classification
- associative memories
- self-organization, learning
- knowledge acquisition, adaptive expert systems
- adaptive signal processing
- control, stabilization, guidance
- decision finding
- optimization procedures
- integration and fusion of multiple sensor data
- robotics, sensory-motor control

It falls beyond the scope of this paper to treat these categories here in more detail. On account of the importance for G.a.C., some further considerations concerning neuro-control should, however, be made.

8. Neuro Control

The application of ANN for control, stabilization and guidance of objects can be considered as a further step in the evolution of control techniques to face up to the challenges within the scope of more complex systems which require more adaptation and self-organization capabilities. Thereby, the main problem is concerned with the real-time control of objects which are nonlinear and noisy and where the dynamics of which is time-varying, only incomplete or even unknown at all.

As common to all ANN, a characteristic feature of the neuro controllers is that they are not programmed but trained either supervised off-line or unsupervised on-line.

As a generalized example the structure of a fault-tolerant, adaptive/learning neuro control system especially suitable for applications on the lower levels of the G.a.C. systems hierarchy (missiles, manned/unmanned air vehicles, robotics, mobile robots etc.) is shown in Fig. 25. As can be seen by this example, neuro-control systems can include subsystems for pattern recognition in sensor data, failure detection and identification, dynamic modelling etc. which are realized as ANN, however, are only of secondary importance for the actual neuro-control problem.

Learning mechanisms based on error backcoupling as shown in Fig. 10 are less suitable for many neuro-control applications since they require a reference signal for supervised learning for the outputs of each single ANN element (PE). These reference signals are often not available from the natural environment.

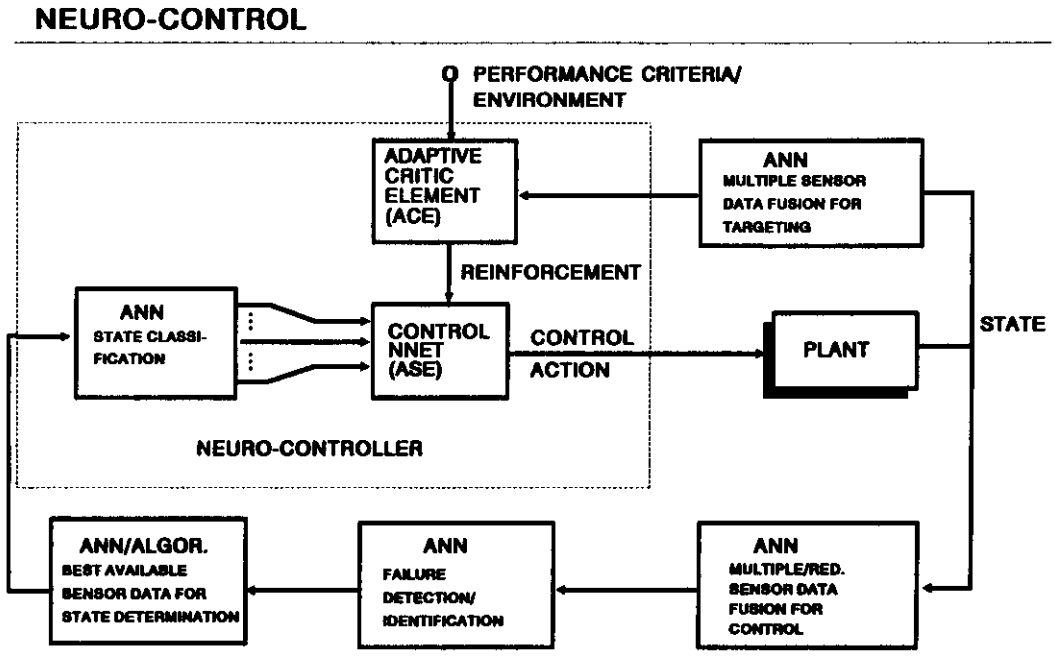


FIGURE 25

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The unsupervised Hebbian learning is often also not applicable to neuro-control. As already mentioned and shown in Fig. 18 it takes information for the adaptation of the connection weights from the network itself (PE activation x_j, PE input s_j). For many ANN applications this is a very favourable characteristic since no communication with the outside world is required for learning. However, this can be a serious disadvantage for neuro-control applications if a particular performance criterion must be met which is referenced to the environment.

The reinforcement learning paradigm (Fig. 26) takes this circumstance into account. Thereby, the reinforcement (r_k) is a measure for the change of a behaviour or performance criteria and thus considers the success or failure of a control action. The eligibility (e_j) of a synaptical pathway is a function of the product resulting from the signal on this pathway and the output of the corresponding PE looked at for a particular delayed period of time. Thus, the eligibility is a measure of up to which extent the input signal on a synaptic connection has also led to a large output signal. The eligibility should decay (for example exponentially, Fig. 26) unless another high value of the eligibility results from the simultaneous occurance of an input signal and the resulting PE activation. The reinforcement learning is formally similar to the Hebbian learning if the PE activation (x_j) and the PE input (s_j) are replaced by reinforcement (r) and eligibility (e).

There are a number of neuro-control paradigms applicable to the design of the actual neuro-controller. The interested reader must refer to the available literature.

As a frontend problem of neuro-control relevant data and facts from similar and/or dissimilar sensor information are to be obtained. Therefore in two examples ANN designs for multiple redundant sensor data failure detection and identification as well as for target identification are briefly presented in the following chapter.

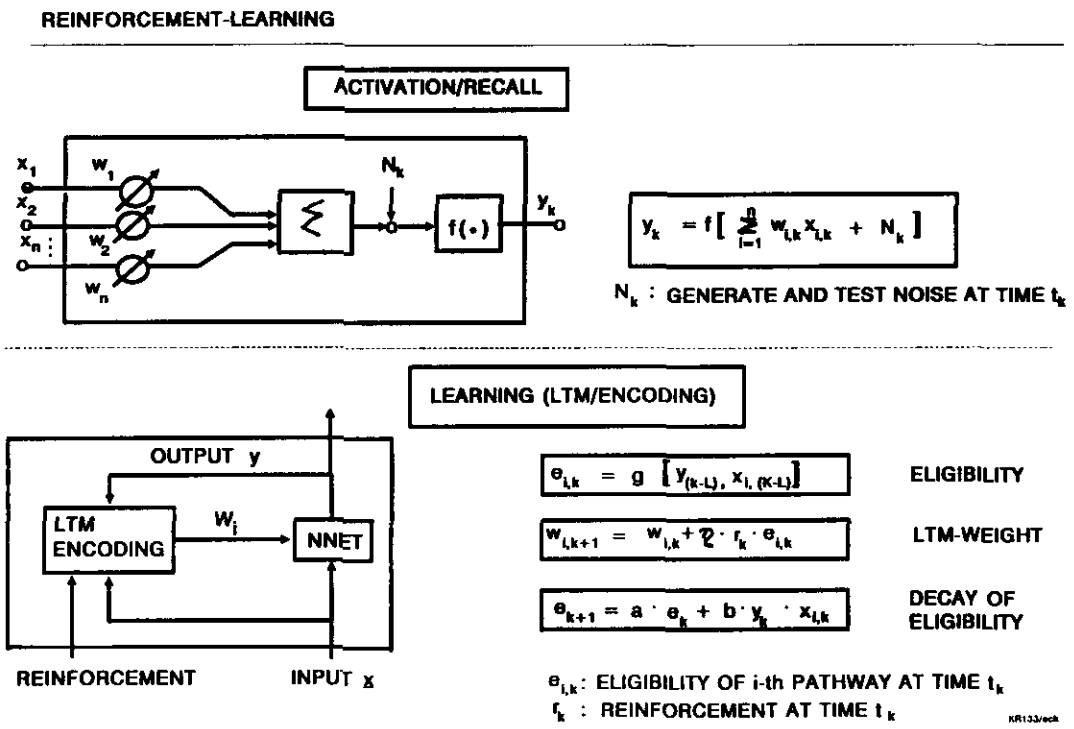


FIGURE 26

9. Example

As a first example the fault tolerant measurement of the proprio-specific motion state of an air vehicle shall be looked at here. For this purpose, a number of redundant sensors for the angular rate (e.g. gyros) as well as for the linear acceleration (accelerometers) are utilized. In order to meet the reliability and fault tolerance requirements with a minimum number of sensors, the arrangement of the sensors is skewed such that each sensor monitors several axes of the air vehicle. The problem now is to detect faults and performance degradation and to localize the possibly defective sensor among the redundantly available ones.

The block diagram of the signal processing elements required for this purpose is shown in Fig. 27. The measurement vector $\underline{m}$ comprises the sensor outputs and is a function of the real physical motion state. Moreover, the amaneable measurement contains contributions due to step, ramp- or stochastic type failures, represented by the failure vector $\underline{\varepsilon}$.

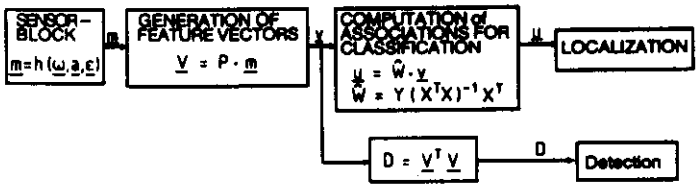
In a first ANN element, so-called validation or feature vectors $\underline{v}^T = [v_1, v_2 \ldots v_n]$ are determined by a projection of the measurement vector $\underline{m}$. In the case of a specified fixed sensor geometry a hard-wired ANN (Fig. 27, left network part) can be used where the connection weights represent the projection mapping P.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The second ANN element performs the fault detection and localization which corresponds to a classification. The output signals calculated by the network and accumulated in the classification vector $\underline{u}$ gives information which sensor is defective i.e. which class the present input feature vector is belonging to. Correspondingly, only one of the output (u_i in Fig. 27) has a high activation (~ 1) while the output of the others is small. In the following two network models are considered for the classification task ANN-module.

FDIR NETWORK

- **BLOCKDIAGRAM**



- **FAILURE DETECTION AND LOCALIZATION**

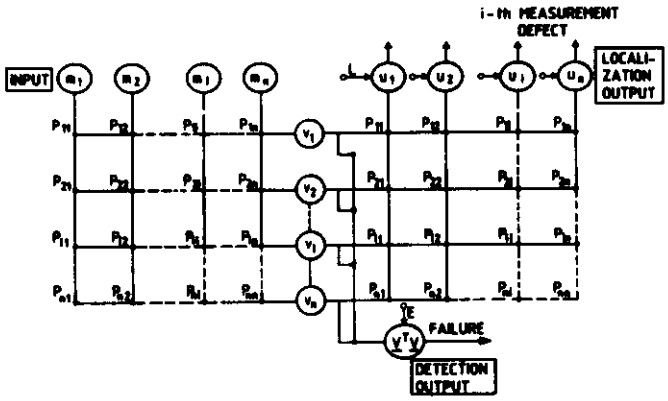


FIGURE 27

Optimal Linear Associative Memory (OLAM)

If the determination of the feature vector ($\underline{v}$) is the result of a mathematically exact modelling of the relation between characteristics and classes, an OLAM can be used. The process of encoding (learning) the input information is then reduced to the a-priori calculation of the optimal weight matrix $\hat{W}$ which mappes the input vector $\underline{v}$ onto the classification vector $\underline{u}$. The optimal weight matrix $\hat{W}$ yielding the least mean square correlation between input ($\underline{v}_k$) and output vector ($\underline{u}_k$) pairs (k = 1,2...m) is computed from the pseudo-inverse of the matrix X as shown in fig. 27. Here, $X = (\underline{v}_1, \underline{v}_2 \ldots \underline{v}_m)$; $Y = (\underline{u}_1, \underline{u}_2 \ldots \underline{u}_m)$. Greville's recursive algorithm can be utilized for example to compute the pseudo-inverse of X. The recalling is simply the multiplication of $\underline{v}$ by the optimal weight matrix W.

With OLAM, the total ANN for the fault detection and localization is hardwired as shown in Fig. 27. It is a three layer network. The number of input PE corresponds to the number of sensor signals, the number of PE in the hidden layer to the dimension of the feature vectors and the number of PE in the output layer to the number of defects or failures to be localized.

Back-propagation network

It has already been mentioned that for the case of linearly unseparable classes multilayered nets must be used for classification. The network model most widely used for this kind of application is the back propagation network. Its function and the associated equations are briefly reviewed here (Fig. 28).

Input and output variables are scaled. The PE of the input layer merely memorize the present input signal. Each input layer PE is connected with all PEs in the hidden layer. The latter multiply the input signals as well as the bias with the associated weight factors

$$x_j = \sum_i s_i w_{ji} + w_o \qquad (13)$$

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

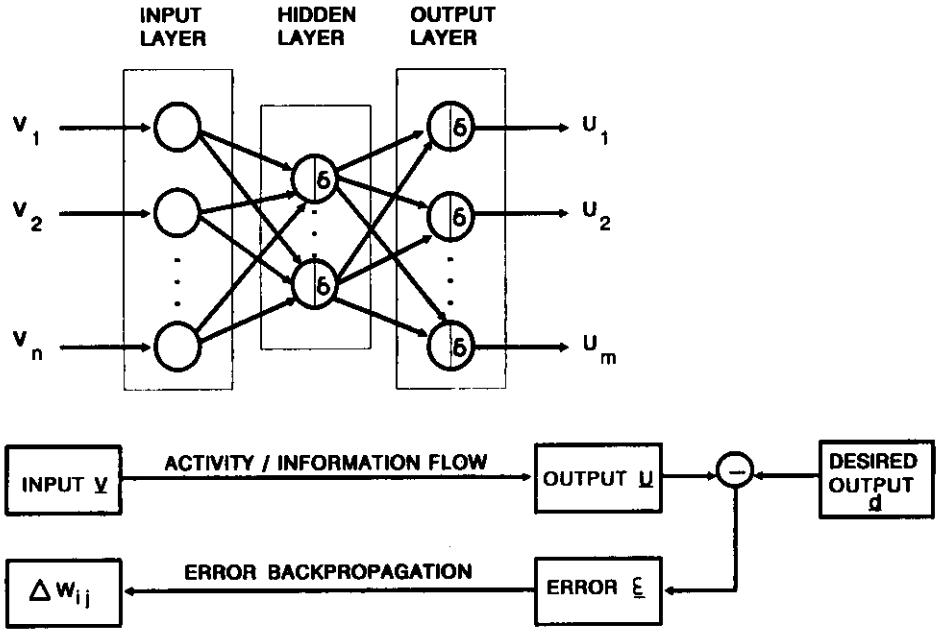


FIGURE 28

The output signal of the j-th hidden PE is the sigmoid function of its activation x_j

$$s_j = \sigma(x_j) \tag{14}$$

with

$$\sigma(x) = 1 / (1+e^{-x}) \tag{15}$$

The same functions are performed by each PE of the output layer while these are also completely interconnected with the PEs of the hidden layer. There are no intra-layer connections. Therefore the back propagation ANN is a feed forward structure in which each element of a follow-up layer receives inputs from all elements of the preceeding layer.

The learning is performed by adapting the connection weights in such a way that the sum of the squares of the error between network output variables ($\underline{u}$) and the desired output variables ($\underline{d}$) of a set of training data is minimized.

Let us assume that there are M input/output vector pairs $\underline{v}^{(m)}$; $\underline{u}^{(m)}$ for the training. Initially, the weights are set to small random values. After the processing of the m-th training data pair, the weights are adapted as follows:

$$w^{(m)} = w^{(m-1)} + \Delta w^{(m)} \tag{16}$$

where $\Delta w^{(m)}$ for the weights between hidden and output layer becomes

$$\Delta w_{kj}^{(m)} = \eta\, \sigma'(x_k)\, (d_k^{(m)} - u_k^{(m)})\, s_j \tag{17}$$

and for the weights between input and hidden layer

$$\Delta w_{ji}^{(m)} = \eta\, \sigma'(x_j) \left[\sum_{k=1}^{N} \sigma'(x_k)\, (d_k^{(m)} - u_k^{(m)})\, \cdot w_{kj}^{(m-1)} \right] v_i^{(m)} \tag{18}$$

In this, η is again a measure for the learning rate, $\sigma'(x)$ is the derivative of the output function and v_i is the i-th input signal.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

As shown by equation (18) the error between actual and desired output is backpropagated from output to hidden-layer PEs. Furthermore, there is a weight transport from output to hidden-layer.

The total network for fault detection and localization (see also Fig. 27) is shown in Fig. 29. As can be seen, the backpropagation ANN is preceeding by the network for the generation of feature vectors already introduced in Fig. 27.

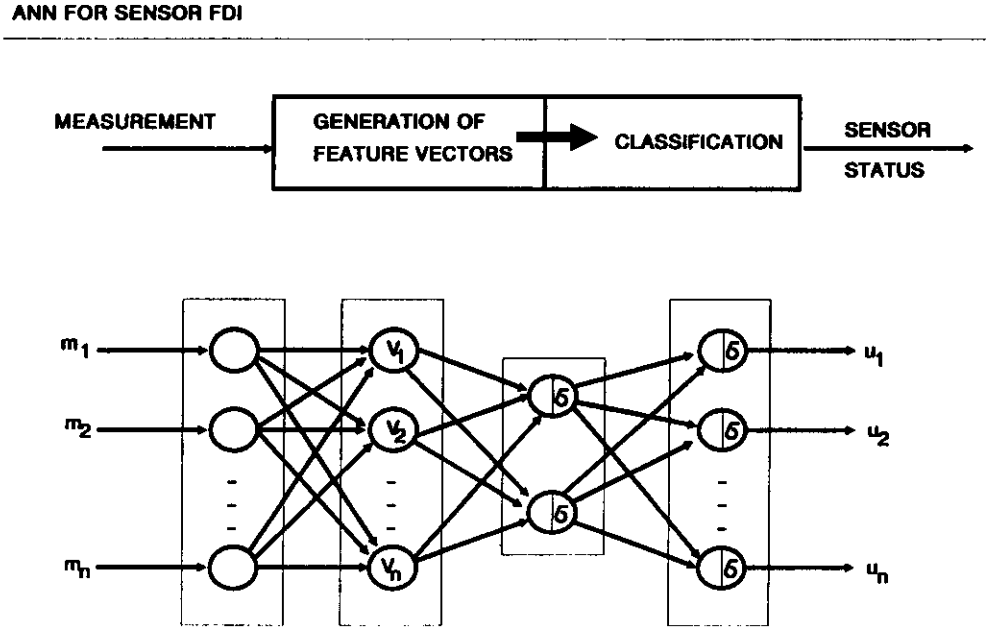


FIGURE 29

The backpropagation part has been trained with training data $\underline{v}^{(m)}$, $\underline{u}^{(m)}$ in approx. 2,000 supervised learning steps. Tests with test data sets showed very good results also with very noisy feature vectors.

The optimization of the number of PEs in the hidden layer generally is a problem of the backpropagation net. The PEs of the input and output layer are determined by the dimensions of the feature and classification vectors.

Neural target classification

For the classification of different targets in infra-red (IR) images, a classifier has been designed on the basis of a backpropagation ANN and compared with the results obtained with a polynomial classifier. The superiority of the neural classifier becomes evident from Fig. 30 where the detection rate is plotted against the false alarm rate for both classifiers. It shall be mentioned here finally that the design and the training of the neural classifier requires far less expense as compared to the development of the conventional one.

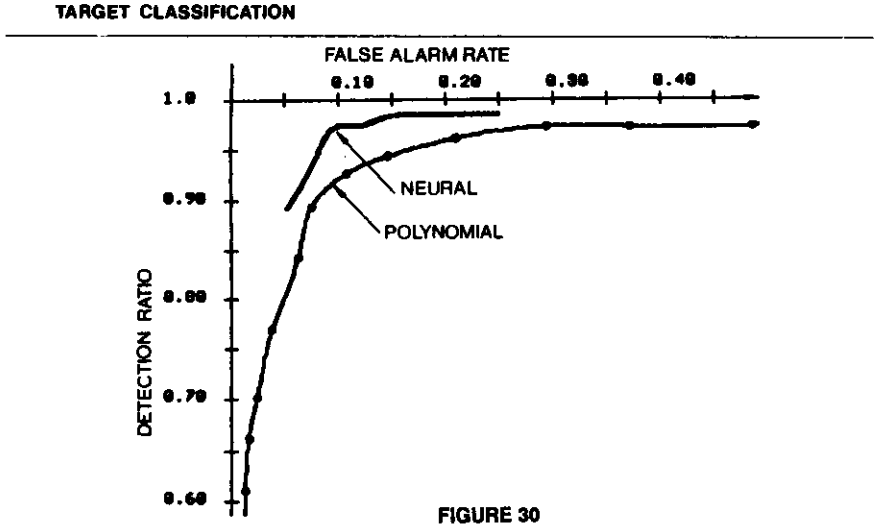


FIGURE 30

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

10. Implementation of neural nets

The following includes a brief summary concerning the possibilities available today for the realization or implementation of ANN:

- Software realizations for existing computers (super computers, massive parallel computers, conventional computers) which are in principle not designed for ANN implementations. Thereby, a mapping of the ANN is made by virtualization on systems and structures in which no or not all connections and processing elements of ANN are indeed physically existing, i.e. they appear as memory areas and/or program structures.

- Electronical implementations which are specifically designed for the layout of the ANN signal processing (bus-related processors, co/attached processors, special integrated circuits). Also analog devices are promising for high-speed ANN implementations.

- Electro-optical or purely optical realizations. These will probably gain great importance in future.

11. Final Remarks

Concerning the artificial neural nets, there is at present a big euphoria. If we look at it soberly, however, it cannot be neglected that there is a whole variety of unsettled questions requiring intensive research. In consideration of the obtained knowledge state and if we are aware of the still unsettled questions, ANN can be used profitably for particular tasks already today.

The biological nerve system is the living example for the fact that strongly meshed systems of an extremely high order can adopt stable states. Moreover, without supervised control, these biological systems are able to act purposively and task oriented. By an extensive comprehension of the biological paradigm, the brain, we must try and strive to recognize the regularities which might be of decisive use to us for the stabilization and self-organization of highly integrated complex dynamic systems.

REFERENCES:

1. Anderson, J.A. and Rosenfeld, E. (1988)
Neurocomputing, Foundations of Research, The MIT Press, Cambridge, Massachusetts

2. Carpenter, G.A. and Grossberg, S. (1990)
ART 3: Hierarchical Search Using Chemical Transmitters in Self-Organizing Pattern Recognition Architectures, Neural Networks 3, 129 - 152

3. Cohen, M. and Grossberg, S. (1983).
Absolute stability of global pattern formation and parallel storage by competitive neural networks, IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-13, pp. 815-825

4. Eckmiller, R. (1990)
Concerning the Emerging Role of Geometry in Neuroinformatics, in *Parallel Processing in Neural Systems and Computers*, R. Eckmiller, G. Hartmann and G. Hauske (Editors), Elsevier Science Publishers B.V. (North Holland)

5. Fukushima, K. (1988)
Neocognitron: A hierarchical neural network capable of visual pattern recognition, Neural Networks, Vol. 1, pp. 119-130

6. Hebb, D.O. (1949)
The Organization of Behavior, Chapter 4, Wiley, New York

7. Hecht-Nielsen, R. (1990)
Neurocomputing, Addison-Wesley: Reading, MA.

8. Kohonen, T (1984)
Self-Organization and Associative Memory, Springer Verlag, Berlin

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

9. Mc.Culloch, W.S. and Pitts W. (1943)

A Logical Calculus of the Ideas Immanent in Nervous Activity, Bulletin of Mathematical Biophysics 5: 115-133

10. Rumelhart, D.E., Hinton G.E. and Williams R.J. (1986)

Learning Internal Representations by Error Propagation, Parallel Distributed Processing: Explorations in the Microstructures of Cognition, Vol. 1, Rumelhart, D.E. and Mc.Clelland, J.L. (Eds.) MIT Press, Cambridge, Massachusetts

11. Simpson, P. (1990a)

Artificial Neural Systems: Foundations, Paradigms, Applications and Implementations, Pergamon Press: Elmsford, NY.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

Neural Network Paradigms

Patrick K. Simpson
General Dynamics Electronics Division
P.O. Box 85310, MZ 7202-K
San Diego, CA 92138

SUMMARY

Building intelligent systems that can model human behavior has captured the attention of the world for years. So, it is not surprising that a technology inspired by the mind and brain such as neural networks has generated great interest. This chapter will provide an evolutionary introduction to neural networks by beginning with the key elements and terminology of neural networks and then developing the topologies, learning laws and recall dynamics from this infrastructure. The perspective taken in this paper is largely that of an engineer, emphasizing the application potential of neural networks and drawing comparisons with other techniques that have similar motivations. Mathematics will be relied upon in many of the discussions to make points as precise as possible.

1. OVERVIEW OF PAPER

This paper begins with a review of what neural networks are and why they are so appealing. A typical neural network is immediately introduced to illustrate several of the key features. Then, the fundamental elements of a neural network such as input and output patterns, the processing element, connections, and threshold operations are described, followed by descriptions of neural network topologies, learning algorithms, and recall dynamics. Next, a taxonomy of neural networks is presented that uses two of their key characteristics: learning and recall. Finally, a comparison of neural networks and similar non-neural information processing methods is presented.

2. WHAT ARE NEURAL NETWORKS AND WHAT ARE THEY GOOD FOR?

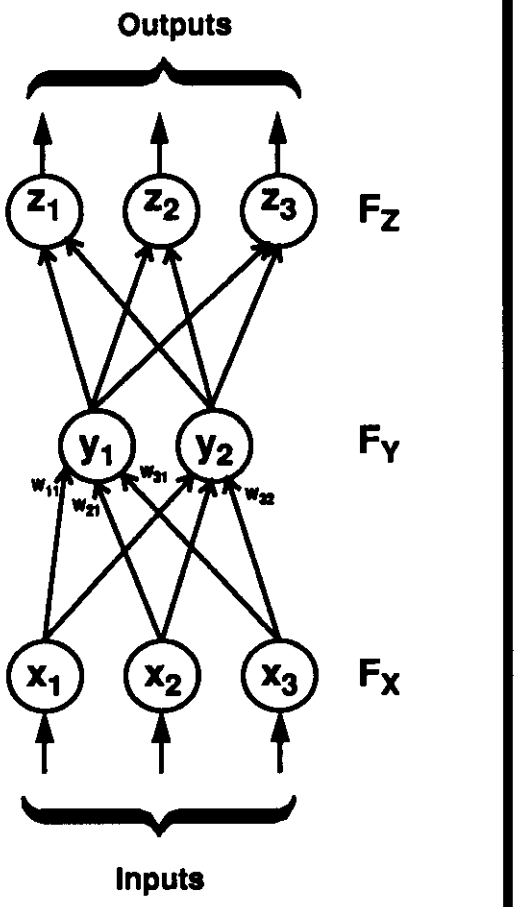
Neural networks are information processing systems. In general, neural networks can be thought of as "black box" devices that accept inputs and produce outputs. Some of the operations that neural networks perform include:

- classification - an input pattern is passed to the network and the network produces a representative class as output.
- pattern matching - an input pattern is passed to the network and the network produces the corresponding output pattern.
- pattern completion - an incomplete pattern is passed to the network and the network produces an output pattern that has the missing portions of the input pattern filled in.
- noise removal - a noise-corrupted input pattern is presented to the network and the network removes some (or all) of the noise and produces a cleaner version of the input pattern as output.
- optimization - an input pattern representing the initial values for a specific optimization problem are presented to the network and the network produces a set of variables that represent a solution to the problem.
- control - an input pattern represents the current state of a controller and the desired response for the controller and the output is the proper command sequence that will create the desired response.

Neural networks consist of layers of processing elements and weighted connections. Each layer in a neural network consists of a collection of processing elements (PEs). Each PE

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Figure 1: A Typical Neural Network



collects the values from all of its input connections, performs a predefined mathematical operation (typically a dot-product followed by a threshold), and produces a single output value.

Figure 1 illustrates a typical neural network with three layers denoted F_X, F_Y, and F_Z. The bottom layer, F_X, accepts inputs into PEs x_1, x_2, x_3. A collection of weighted connections (sometimes called "weights" or "connections") connect the F_X PEs to the F_Y PEs. The F_Y PEs, y_1 and y_2, are the hidden layer. Similarly, the F_Y PEs are connected to the F_Z PEs which form the output layer. The weight names serve as both a label and a value. As an example, in Figure 1 the connection from the F_X PE x_1 to the F_Y PE y_2 is the connection weight w_{12} (the connection from x_1 to y_2). By adjusting the connection weights, information is stored in the network. The value of the connection weights are often determined by a neural network learning procedure (although sometimes they are predefined and hardwired into the network).By performing the update operations for each of the PEs the neural network recalls information.

There are two important features illustrated by the neural network shown in Figure 1 that apply to all neural networks:

- *Local Operations*. Each PE acts independently of all others. A PE's output relies only on its constantly available inputs from the abutting connections. The information provided by the adjoining connections is all a PE needs to process. Information from other PEs where an explicit connection does not exist is not necessary.
- *Distributed Representation*. The large number of connections provides a large amount of redundancy and facilitates a distributed representation. A large number of connections must be eliminated for a significant amount of information to be destroyed.

The first feature allows neural networks to operate efficiently in parallel. The last feature provides neural networks with inherent fault-tolerance and generalization qualities that are very difficult to attain from typical computing systems. In addition to these features, neural networks can learn arbitrary nonlinear mappings given the proper topology, nonlinear processing elements from nonlinear threshold operations, and appropriate learning rules. The ability to learn nonlinear mappings simply by presenting instances of input and output patterns is a powerful attribute shared by few systems.

There are three primary situations where neural networks are useful:

- Situations where only a few decisions are required from a massive amount of data (e.g. speech and image processing).

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

- Situations where nonlinear mappings must be automatically acquired (e.g. loan evaluations and robotic control).
- *Situations where a near-optimal solution to* a combinatorial optimization problem is required very quickly (e.g. airline scheduling and telecommunication message routing).

To summarize, the foundations of neural networks consist of an understanding of the nomenclature and a firm comprehension of the rudimentary mathematical concepts used to describe and analyze neural network processing. In a broad sense, neural networks consist of three principle elements:

- *Topology.* A neural network's organization into interconnected layers.
- *Learning.* The adjustment of weights to store information.
- *Recall.* Retrieving information stored in the weights.

Sections 4, 5, and 6 describes each of these elements, respectively. Prior to these discussions, Section 3 will address the fundamental components used to create a neural network: connections, processing elements, and threshold functions.

3. DISSECTING NEURAL NETWORKS

A convenient neural network analogy is the directed graph, where the edges and nodes correspond to weights and PEs, respectively. In addition to connections and processing elements, threshold functions and input/output patterns are also basic elements in the design, implementation and use of neural networks. After a description of the terminology used to describe neural networks, each of these elements will be examined in turn.

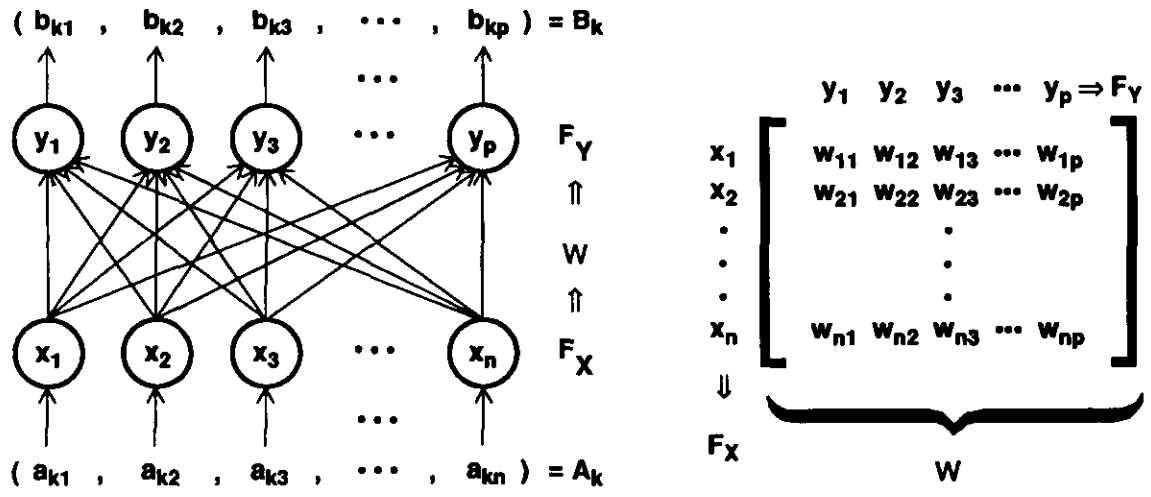
3.1. Terminology

Unfortunately, neural network terminology remains varied, with a standard yet to be adopted (although there is an effort to create one, cf. Eberhart, 1990). To illustrate some of the terminology introduced here, please refer to Figure 2.

Input and output vectors (patterns) are denoted by subscripted capital letters from the beginning of the alphabet. The input m patterns are denoted as $A_k = (a_{k1}, a_{k2}, ..., a_{kn})$; k = 1, 2, ..., m, and the output patterns as $B_k = (b_{k1}, b_{k2}, ..., b_{kp})$; k = 1, 2, ..., m.

The PEs in a layer will be denoted by the same subscripted variable. The collection of PEs in a layer form a vector and these vectors will be denoted by capital letters from the end

Figure 2: Two-layer Feedforward Neural Network & Weight Matrix



This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

of the alphabet. In most cases three layers of PEs will suffice. The input layer of PEs is denoted as $F_X = (x_1, x_2, ..., x_n)$, where each x_i receives input from the corresponding input pattern component a_{ki}. The next layer of PEs will be the F_Y PEs, then the F_Z PEs (if either layer is necessary). The dimensionality of these layers depends on its use. Using the network in Figure 2 as an example, the second layer of the network is the output layer, hence the number of F_Y PEs must match the dimensionality of output patterns. In this instance, the output layer is denoted as $F_Y = (y_1, y_2, ..., y_p)$, where each y_j is correlated with the j'th element of B_k.

Connection weights are stored in weight matrices. Weight matrices will be denoted by capital letters toward the middle of the alphabet, such as U, V, and W. Referring to the example in Figure 2, this two layer neural network requires one weight matrix to fully connect the layer of n F_X PEs to the layer of p F_Y PEs. The matrix shown in Figure 2 describes the full set of connection weights between F_X and F_Y, where the weight w_{ij} is the connection weight from the i'th F_X PE, x_i, to the j'th F_Y PE, y_j.

3.2. Input and Output Patterns

Neural networks can not operate unless they have data. Some neural networks require only single patterns and others require pattern pairs. Note that the dimensionality of the input pattern is not necessarily the same as the output pattern. When a network only works with single patterns, it is an autoassociative network. When a network works with pattern pairs it is heteroassociative.

One of the key issues when applying neural networks is determining what the patterns should represent. For example, in speech recognition there are many different types of features that can be employed (Lippmann, 1989), including: linear predictive coding coefficients, Fourier spectra, histograms of threshold crossings, cross-correlation values. The proper selection and representation of these features can greatly affect the performance of the network.

In some instances the representation of the features as a pattern vector is constrained by the type of processing the neural network can perform. Some networks can only process binary data, such as the Hopfield network (Hopfield, 1982; Amari, 1972), Binary Adaptive Resonance Theory (Carpenter & Grossberg, 1987a), and the Brain- State-in-a-Box (Anderson, et al., 1977). Others can process real-valued data such as backpropagation (Werbos, 1974; Parker, 1982; Rumelhart, Hinton, & Williams, 1986) and Learning Vector Quantization (Kohonen, 1984). Creating the best possible set of features and properly representing those features is the first step toward success in any neural network application (Anderson, 1990).

3.3. Connections

A neural network is equivalent to a directed graph (digraph). A digraph has edges (connections) between nodes (PEs) that allow information to flow in only one direction (the direction denoted by the arrow). Information flows through the digraph along the edges and is collected at the nodes. Within the digraph representation, connections determine the direction of information flow. As an example, in Figure 2 the information flows from the F_X layer through the connections, W, to the F_Y layer. Neural networks extend the digraph representation to include a weight with each edge (connection) that modulates the amount of output signal passed from one node (PE) down the connection to the adjacent node. For simplicity, the dual role of connections will be employed. A connection both defines the information flow through the network and it modulates the amount of information passing between to PEs.

The connection weights are adjusted during a learning process that captures information. Connection weights that are positive valued are excitatory connections. Those that with negative values are inhibitory connections. A con-

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

nection weight that has a zero value is the same as not having a connection present. By only allowing a subset of all the possible connections to have non-zero values, sparse connectivity between PEs can be simulated.

It is often desirable for a PE to have an internal bias value (threshold value). Panel (a) of Figure 3 shows the PE y_j with three connections from F_X $\{w_{1j}, w_{2j}, w_{3j}\}$, and a bias value, θ_j. It is convenient to consider this bias value as an extra connection, w_{0j}, emanating from the F_X PE x_0, with the added constraint that x_0 is always equal to 1 as shown in panel (b). This mathematically equivalent representation simplifies many discussions. Throughout the paper this method of representing the bias (threshold) values will be employed.

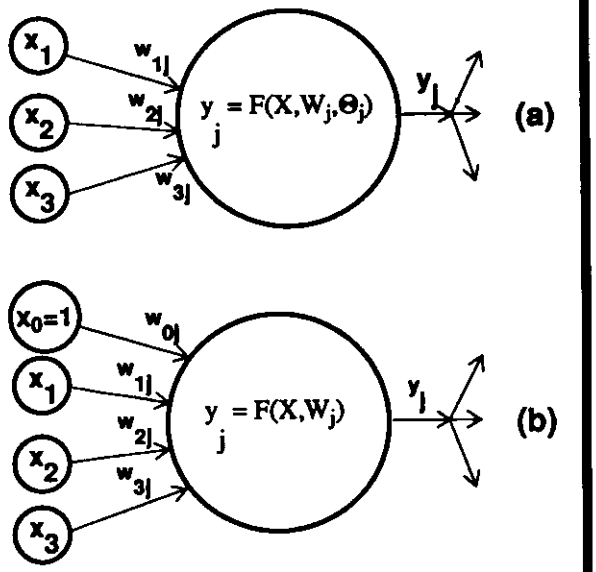
3.4. Processing Elements

The processing element (PE) is the portion of the neural network where all the computing is performed. Figure 3 illustrates the most common type of PE. A PE can have one input connection, as is the case when the PE is an input layer PE and it receives only one value from the corresponding component of the input pattern, or it can have several weighted connections, as is the case of the F_Y PEs shown in Figure 2 where there is a connection from every F_X PE to each F_Y PE. Each PE collects the information that has been sent down its abutting connections and produces a single output value. There are two important qualities that a PE must possess:

- *Local Operations*. Described earlier in §1.
- *Single Output Value*. Each PE produces a single output value that is propagated through the connections from the emitting PE to other receiving PEs or it will be output from the network.

These two qualities allow neural networks to operate in parallel. The value of the PE and its label use the same symbol. As an example, the output PE label y_j in Figure 3 represents both the PEs placement in the network and its value.

Figure 3: The Processing Element



There are several mechanisms for computing the output of a processing element. The output value of the PE shown in Figure 3(b), y_j, is a function of the outputs of the preceding layer, $F_X = X = (x_1, x_2, ..., x_n)$ and the weights from F_X to y_j, $W_j = (w_{1j}, w_{2j}, ..., w_{nj})$. Mathematically, the output of y_j is a function of its inputs and its weights,

$$y_j = F(X, W_j). \qquad (1)$$

3.4.1. Linear Combination

The most common computation performed by a PE is a linear combination (dot-product) of the input values, X, with the abutting connection weights, W_j, followed by a threshold operation (cf. Simpson, 1990a; Hecht-Nielsen, 1990; Maren, Harston & Pap, 1990). Using the PE in Figure 3(b) as an example, the output y_j is computed using the equation

$$y_j = f\left(\sum_{i=0}^{n} x_i w_{ij}\right) = f(X \bullet W_j) \qquad (2)$$

where $W_j = (w_{1j}, w_{2j}, ..., w_{nj})$ and f is one of the threshold functions described in §3.4. of this

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

chapter. The dot product update has a very appealing quality that is intrinsic to its computation. Using the relationship $A_k \cdot W_j = \cos(A_k, W_j)/\|A_k\| \|W_j\|$, it is seen that the larger the dot product (assuming fixed length A_k and W_j) the more similar the two vectors are. Hence, the dot product can be viewed as a similarity measure.

3.4.2. Mean-Variance Connections

In some instances PEs will have two connections interconnecting PEs instead of just one as shown in Figure 4. One use of these dual connections is to allow one set of the abutting connections represent the mean of a class and the other the variance of the class (Lee & Kil, 1989; Robinson, Niranjan, & Fallside, 1988). In this case, the output value of the PE depends on the inputs and both sets of connections, i.e. $y_j = F(X, V_j, W_j)$, where the mean connections are represented by $W_j = (w_{1j}, w_{2j}, ..., w_{nj})$ and the variance connections $V_j = (v_{1j}, v_{2j}, ..., v_{nj})$ for the PE y_j. Using this scheme, the output of y_j is calculating the difference between the input, X, and the mean, W_j, divided by the variance, V_j, squaring the resulting quantity, and passing this value through a Gaussian threshold function to produce the final output value as follows

$$y_j = g\left(\sum_{i=1}^{n}\left(\frac{w_{ij} - x_i}{v_{ij}}\right)^2\right) \qquad (3)$$

where the Gaussian threshold function is

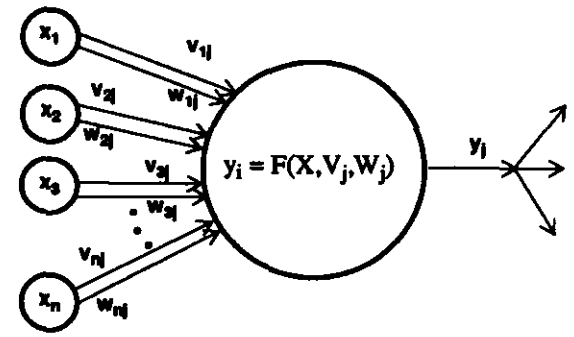
$$g(x) = exp(\frac{-x^2}{2}) \qquad (4)$$

The Gaussian threshold function is described in greater detail in §3.5.5.

3.4.3. Min-Max Connections

Another less common use of dual connections is to assign one of the abutting vectors, say V_j, to become the minimum bound for the class and the other vector, W_j, to becomes the maximum bound for the same class. By measuring the amount of the input pattern that falls within the bounds, a min-max activation value is produced (Simpson, 1990b). Figure 5 illustrates this notion using a graph representation for the min and the max points. The ordinate of the graph represents the value of each element of the min and max vectors and the abscissa of the graph represents the dimensionality of the classification space. The input pattern, X, is compared with the bounds of the class. The amount of disagreement between the classes bounds, V_j and W_j, and input pattern, X, is shown in the shaded regions. The measure of these shaded regions produces an activation value y_j.

Figure 4: Dual Connections



A fuzzy set, A, is defined as a set of ordered pairs, $A = \{x, m_A(x)\}$. A direct analogy with fuzzy sets is found when the min-max class is the collection of points defining some set and the classification function is the membership function. When cast in this framework, each class in a fuzzy min-max network is actually a fuzzy set. The classification value produced from the fuzzy min-max PEs represents the degree to which an input pattern (object) fits within each of the classes (fuzzy sets). Referring once again to Figure 5 and utilizing this fuzzy logic scheme, the max bound, W_j, is the maximum point allowed in class j and the min bound, V_j, is the minimum point allowed in class j. Measuring the degree to which X falls between V_j and W_j can be done by measuring

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

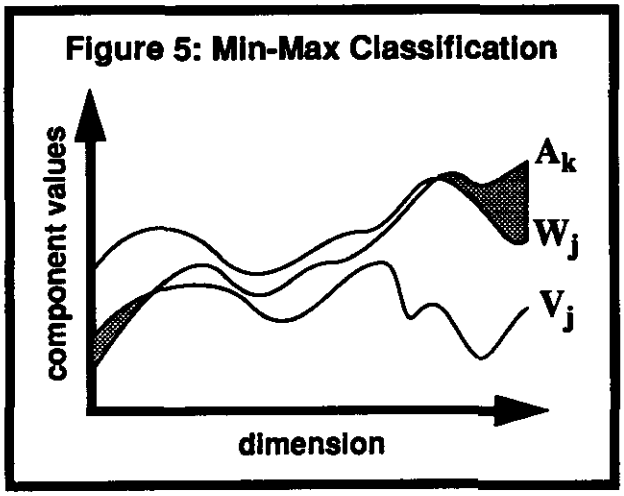


Figure 5: Min-Max Classification

the relative amount of X that falls outside class j. Rescaling the n-dimensional space to lie within the unit cube allows the use of the fuzzy supersethood and subsethood measures to produce classification values. (Kosko, 1986a). The activation value of y_j (the degree to which X belongs to the class j) is defined as the degree to which X is a superset of W_j times the degree to which V_j is a subset of X, yielding the output value

$$y_j = (1 - \text{supersethood}(X, W_j)) \times (1 - \text{subsethood}(X, V_j)) \quad (5)$$

It is easy to show that y_j is bound to the closed interval from 0 to 1. When $y_j = 1$, X lies completely within the min-max bounds. When $y_j = 0$, X falls completely outside of the min- max bounds. When $0 < y_j < 1$, the value describes the degree to which X is contained by the min-max bounds.

3.5. Threshold Functions

Threshold functions, also referred to as activation functions, squashing functions, or signal functions, map a PE's (possibly) infinite domain to a prespecified range. Although the number of threshold functions possible is quite varied, there are five that are regularly employed by the majority of neural networks: (1) linear, (2) step, (3) ramp, (4) sigmoid, and (5) Gaussian. With the exception of the linear threshold function, all of these introduce a non-linearity in the network dynamics by bounding a PE's output values to a fixed range.

3.5.1. Linear Threshold Function

The linear threshold function (see Figure 6(a)), produces a linearly modulated output from the input x as described by the equation

$$f(x) = \alpha x \quad (6)$$

where x ranges over the real numbers and α is a positive scalar. If $\alpha = 1$, it is equivalent to removing the threshold function completely.

3.5.2. Step Threshold Function

The step threshold function, (see Figure 6(b)), produces only two values, β and δ. If the input to the threshold function, x, equals or exceeds the threshold value, θ, then the step threshold function produces the value β, otherwise it produces the value $-\delta$, where β and δ are positive scalars. Mathematically this function is described as

$$f(x) = \begin{cases} \beta & \text{if } (x \geq \theta) \\ -\delta & \text{if } (x < \theta) \end{cases} \quad (7)$$

Typically the step threshold function produces a binary value in response to the sign of the input, emitting +1 if x is positive and 0 if it is not. By making the assignments β=1, δ=0, and θ=0, the step threshold function becomes the binary step function

$$f(x) = \begin{cases} 1 & \text{if } (x \geq 0) \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

which is common to neural networks such as the Hopfield neural network (Amari, 1972; Hopfield, 1982) and the Bidirectional Associative Memory (Kosko, 1988). One small variation of equation (8) is the bipolar threshold function

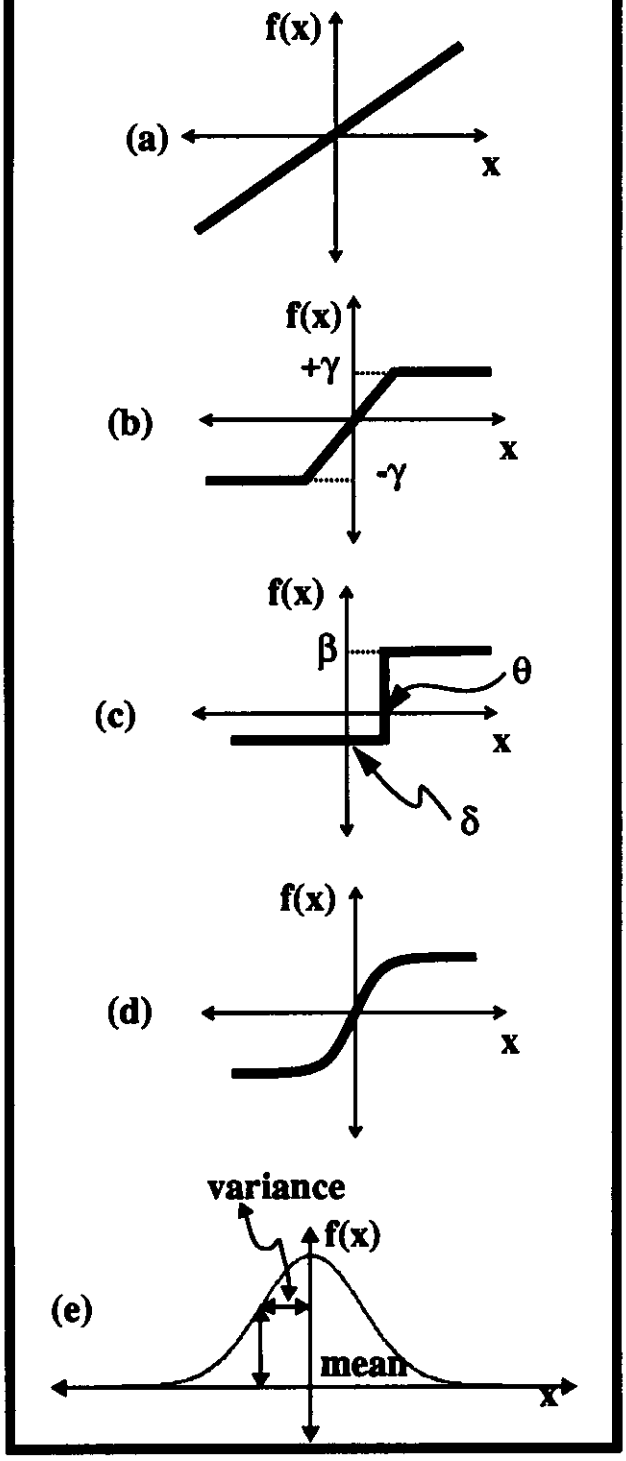
$$f(x) = \begin{cases} 1 & \text{if } (x \geq 0) \\ (-1) & \text{otherwise} \end{cases} \quad (9)$$

which replaces the 0 output value with a -1. In

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

punish-reward systems such as the Associative Reward-Penalty (Barto, 1985), the negative value is used to ensure changes, where a 0 will not.

Figure 6: Threshold Functions



3.5.3. Ramp Threshold Function

The ramp threshold function, (see Figure 6(c)), is a combination of the linear and step threshold functions. The ramp threshold function places an upper and lower bound on the values that the threshold function produces and allows a linear response between the bounds. These saturation points are symmetric around the origin and are discontinuous at the points of saturation. The ramp threshold function is defined as

$$f(x) = \begin{cases} \gamma & \text{if } (x \geq \gamma) \\ x & \text{if } (|x| < \gamma) \\ -\gamma & \text{if } (x \leq -\gamma) \end{cases} \qquad (10)$$

where γ is the saturation value for the function and the points $x = \gamma$ and $x = -\gamma$ are where the discontinuities in f exist.

3.5.4. Sigmoid Threshold Function

The sigmoid threshold function, (see Figure 6(d)), is a continuous version of the ramp threshold function. The sigmoid (S-shaped) function is a bounded, monotonic, non-decreasing function that provides a graded, nonlinear response within a prespecified range.

The most common sigmoid function is the logistic function

$$f(x) = \frac{1}{1 + e^{-\alpha x}} \qquad (11)$$

where $\alpha > 0$ (usually $\alpha = 1$), which provides an output value from 0 to 1. This function is familiar to statistics (as the Gaussian distribution function), chemistry (describing catalytic reactions), and sociology (describing human population growth). Note that a relationship between equation (11) and equation (8) exists. When $\alpha = \infty$ in equation (11), the slope of the sigmoid function between 0 and 1 becomes infinitely steep and, in effect, becomes the step function described by equation (8).

Two alternatives to the logistic sigmoid function are the hyperbolic tangent

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

$$f(x) = \tanh(x) \qquad (12)$$

which ranges from -1 to 1, and the augmented ratio of squares

$$f(x) = \begin{cases} [x^2/(1+x^2)] & \text{if } (x>0) \\ 0 & \text{otherwise} \end{cases} \qquad (13)$$

which ranges from 0 to 1.

3.5.5. Gaussian Threshold Function

The Gaussian threshold function, (see Figure 6(e)), is a radial function (symmetric about the origin) that requires a variance value, $v > 0$, to shape the Gaussian function. In some networks the Gaussian function is used in conjunction with a dual set of connections as described earlier by equation (3) and in other instances (Specht, 1990) the variance is predefined. In the latter instance, the threshold function is

$$f(x) = \exp(\frac{-x^2}{v}) \qquad (14)$$

where x is the mean and v is the predefined variance.

4. NEURAL NETWORK TOPOLOGIES

The building blocks for neural networks are in place. Neural network topologies now evolve from the patterns, PEs, connections, and threshold functions described in §3. Neural networks consist of layer(s) of PEs interconnected by weighted connections. The arrangement of the PEs, connections and patterns into a neural network is referred to as a topology. After introducing some terminology six common neural network topologies will be described.

4.1. Terminology

4.1.1. Layers

Neural networks are organized into layers of PEs. PEs within a layer are similar in two respects: (1) the connections that feed the layer of PEs is from the same source, eg. the F_X layer of PEs in Figure 2 all receive their inputs from the input pattern and the layer of F_Y PEs all receive their inputs from the F_X PEs; and (2) the PEs in each layer utilize the same type of update dynamics, eg. all the PEs will use the same type of connections and the same type of threshold function.

4.1.2. Intralayer vs. Interlayer Connections

There are two types of connections that a neural network employs: intralayer connections and interlayer connections. Intralayer connections are connections between PEs in the same layer. Interlayer connections are connections between PEs in different layers. It is possible to have neural networks that consist of one, or both, types of connections.

4.1.3. Feedforward vs. Feedback Networks

When a neural network has connections that feed information in only one direction, from input to output, without any feedback pathways in the network, it is a feedforward neural network. The network is a feedback network if the network has any feedback paths, where feedback is defined as any path through the network that would allow the same PE to be visited twice.

4.2. Instars, Outstars & the Adaline

The two simplest neural networks are the instar and the outstar (Grossberg, 1982). The instar (see Figure 7(a)), is the minimal pattern encoding network. A simple example of an encoding procedure for the instar would take the pattern, $A_k = (a_{k1}, a_{k2}, ..., a_{kn})$, normalize it, and use the values as the weights, $W_j = (w_{1j}, w_{2j}, ..., w_{nj})$, as shown by the equation

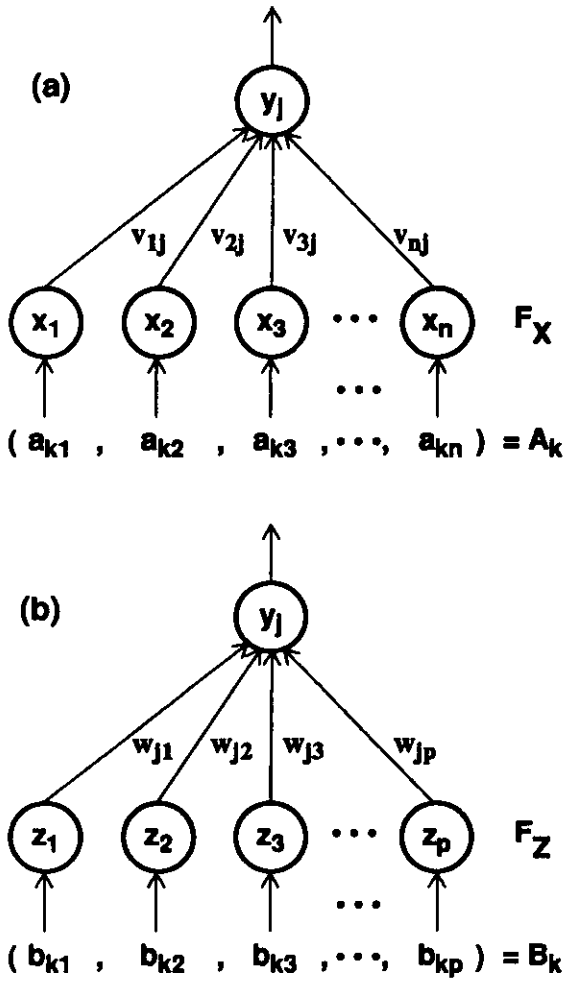
$$v_{ij} = \frac{a_{ki}}{\sum_{i=1}^{n} a_{ki}} \qquad (15)$$

for all i = 1, 2,..., n.

The dual of the instar is the outstar, (see Figure 7(b)). The outstar is the minimal pattern recall neural network. An output pattern is gen-

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Figure 7: Instar & Outstar



erated from the outstar using the equation

$$z_i = y_j w_{ji} \qquad (16)$$

for all i = 1, 2, ..., p, where the weights are determined using equation (15) or one of the learning algorithms described in §5.

The ADALINE, ADAptive LInear NEuron, (Widrow & Hoff, 1960) has the same topology as the instar (see Figure 7(a)), but the weights, V_j, are adjusted using the Least-Mean-Square (LMS) algorithm (see §5.7.1.). In the framework of adaptive signal processing, a similar topology with the same functionality is referred to a finite impulse response (FIR) filter (Widrow & Stearns, 1985). Applications of the FIR filter to noise cancellation, echo cancellation, adaptive antennas, and control are numerous (Widrow & Winter, 1988).

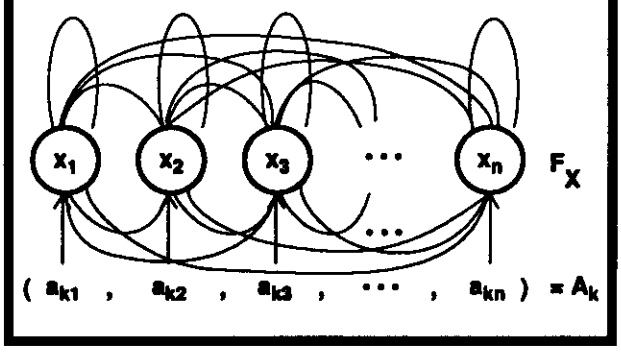
4.3. Single-layer Networks: Autoassociation, Optimization, and Contrast Enhancement

Beyond the instar/outstar neural networks are the single layer intraconnected neural networks. Figure 8 shows the topology of a one-layer neural network which consists of n F_X PEs. The connections from each F_X PE to every other F_X PE and itself, yielding a connection matrix with n^2 entries. The single-layer neural network accepts an n-dimensional input pattern in one of three ways:

- *PE Initialization Only.* The input pattern is used to initialize the F_X PEs and the input pattern does not influence the processing thereafter.
- *PE Initialization and Constant Bias.* The input pattern is used to initialize the F_X PEs and the input remains as a constant valued input bias throughout processing.
- *Constant Bias Only.* The PEs are initialized to all zeroes and the input pattern acts as a constant valued bias throughout processing.

One-layer neural networks are used for pattern completion, noise removal, optimization, and contrast enhancement. The first two operations are performed by autoassociatively encoding patterns and typically using the input

Figure 8: One-layer Neural Network



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

pattern for PE initialization only. The optimization networks are dynamical systems that stabilize to a state that represents a solution to an optimization problem and typically uses the inputs for both PE initialization and as constant biases. Contrast enhancement networks use the input patterns for PE initialization only and can operate in such a way that eventually only one PE remains active. Each of these one-layer neural networks are described in greater detail in the following paragraphs.

4.3.1. Pattern Completion

Pattern completion in a single-layer neural network is performed by presenting a partial pattern initially, and relying upon the neural network to complete the remaining portions. As an example, assume a single layer neural network has stored images of human faces. If half of a face is presented to the neural network as the initial state of the network, the neural network would complete the missing half of the face and output a complete face.

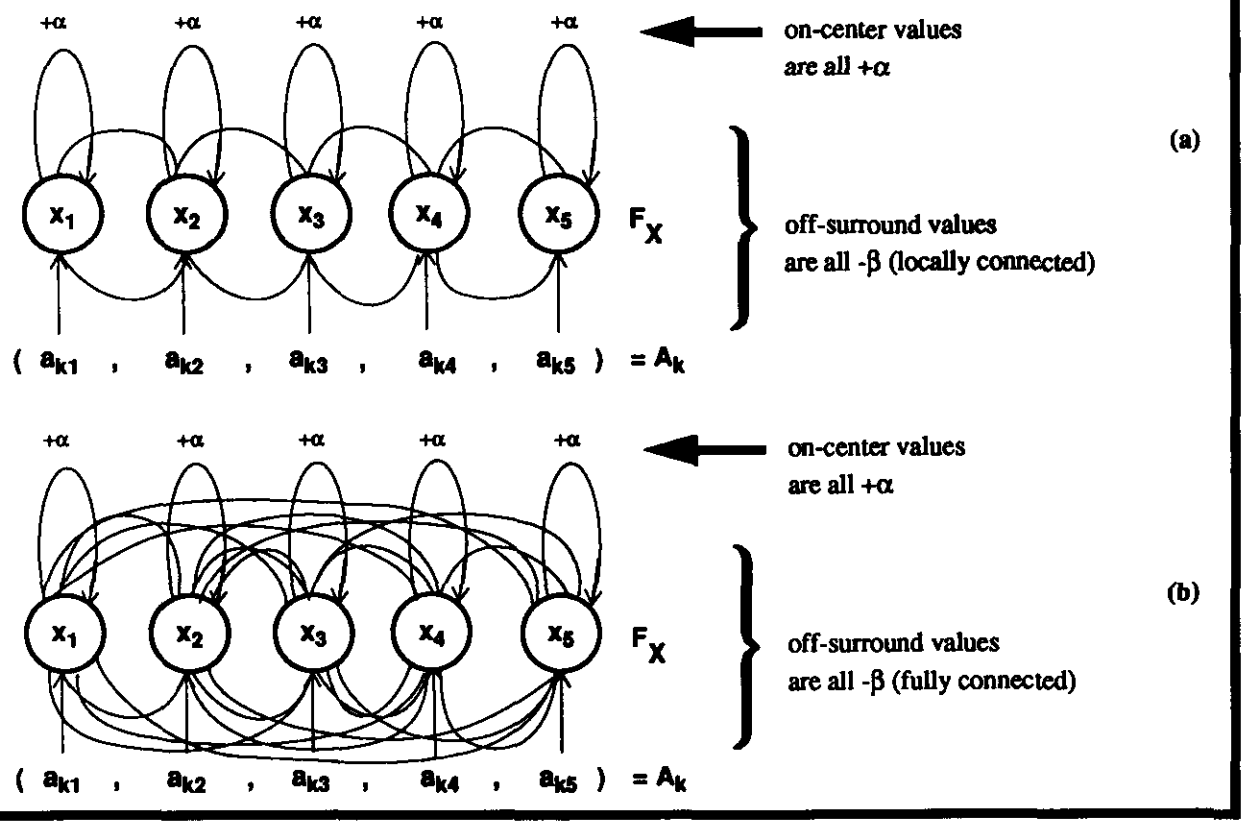
4.3.2. Noise Removal

Noise removal is similar to pattern completion in that a complete, noise-free, response is desired from a pattern corrupted by noise. Fundamentally there is no difference between noise removal and pattern completion. The difference tends to be entirely operational. Using the previous image storage example, if a blurry or splotchy image is presented to the neural network, the output would be a crisp clear image. Single-layer neural networks designed for pattern completion and noise cancellation include the Discrete Hopfield network (Hopfield, 1982), the Brain-State-in-a-Box (Anderson, et al., 1977), and the Optimal Linear Associative Memory (Kohonen, 1984).

4.3.3. Neural Optimization

One of the most prevalent uses of neural networks is optimization (Hopfield & Tank,

Figure 9: Local and Global Contrast Enhancement



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

1985; Tank & Hopfield, 1986). Optimization is a technique for solving a problem by casting it into a mathematical equation that, when either maximized or minimized, solves a problem. Typical examples of problems approached using an optimization technique include scheduling, routing, and resource allocation. The neural optimization approach casts the optimization problem into the form of an energy function that describes the dynamics of a neural system. If the neural network dynamics are such that the network will always seek a stable state when the energy function is at a minimum, then the network will automatically find a solution. The inputs to the neural network are the initial state of the neural network and the final PE values represent the parameters of a solution.

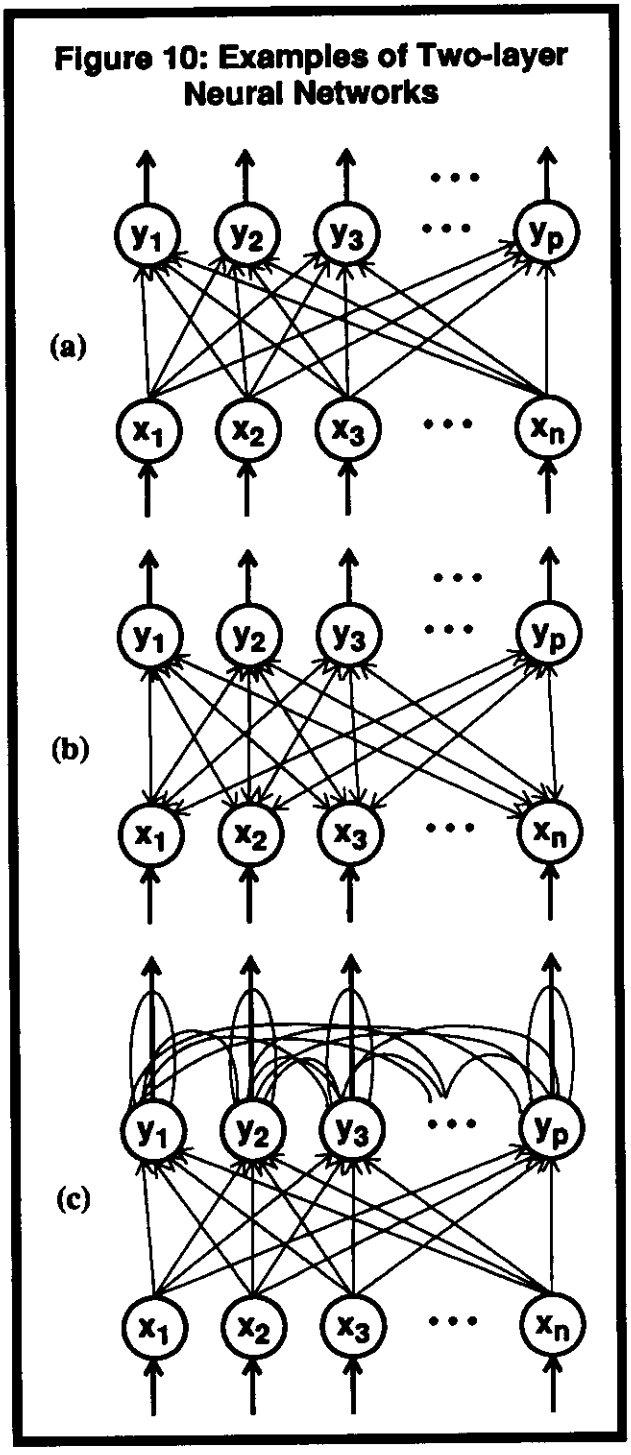
4.3.4. Contrast Enhancement

Contrast enhancement in single-layer neural networks is achieved using on-center/off-surround connection values. The on-center connections are positive self-connections, i.e. $w_{ii} = \alpha$ ($\alpha > 0$) for all i = 1, 2, ..., n, that allow a pattern's activation value to grow by feeding back upon themselves. The off-surround connections are negative neighbor connections, i.e. $w_{ij} = -\beta$ ($\beta > 0$) for all i not equal to j, that compete with the on-center connections. The competition between the positive, on-center, and the negative, off-surround, activation values are referred to as competitive dynamics. Contrast enhancement neural networks take one of two forms: locally connected and globally connected. If the connections between the F_X PEs are only connected to a few of the neighboring PEs (see Figure 9(a)), the result is a local competition that can result in several large activation values. If the off-surround connections are fully interconnected across the F_X layer (see Figure 9(b)), the competition will yield a single winner.

4.4. Two-layer Networks: Heteroassociation and Classification

Two-layer neural networks consist of a layer of n F_X PEs fully interconnected to a layer of p F_Y PEs as shown in Figure 10. The connections from the F_X to F_Y PEs form the n-by-p weight matrix W where w_{ij} represents the

Figure 10: Examples of Two-layer Neural Networks



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

weight for the connection from i'th F_X PE, x_i, to the j'th F_Y PE, y_j. There are three common types of two-layer neural networks: feedforward pattern matchers, feedback pattern matchers, and feedforward pattern classifiers.

4.4.1. Feedforward Pattern Matching

A two-layer feedforward pattern matching neural network maps the input patterns, A_k, to the corresponding output patterns, B_k, k = 1, 2, ..., m. The network shown in Figure 10(a) illustrates the topology of this feedforward network. The two-layer feedforward neural network accepts the input pattern A_k and produces an output pattern, $Y = (y_1, y_2, ..., y_p)$, that is the network's best estimate of the proper output given A_k as the input. An optimal mapping between the inputs and the outputs is one that produces the correct response B_k when A_k is presented to the network, k = 1, 2, ..., m. Most two-layer networks are concerned with finding the optimal linear mapping between the pattern pairs (A_k,B_k) (cf. Widrow & Winter, 1988; Kohonen, 1984), but there are other two-layer feedforward networks that also work with nonlinear mappings by extending the input patterns to include multiplicative combinations of the original inputs (Pao, 1989; Maren, Harsten & Pap, 1990).

4.4.2. Feedback Pattern Matching

A two-layer feedback pattern matching neural network, shown in Figure 10(b), accepts inputs from either layer of the network, either the F_X and F_Y layers, and produces the output for the other layer (Kosko, 1988; Simpson, 1990).

4.4.3. Feedforward Pattern Classification

A two-layer pattern classification neural network, shown in Figure 10(c), maps an input pattern, A_k, to one of p classes. By representing each class as a separate F_Y PE, the pattern classification task is then reduced to selecting the F_Y PE that best responds to the input pattern. Most two-layer pattern classification systems utilize the competitive dynamics of global on-center/off-surround connections to perform the classification.

4.5. Multi-layer Networks: Heteroassociation and Function Approximation

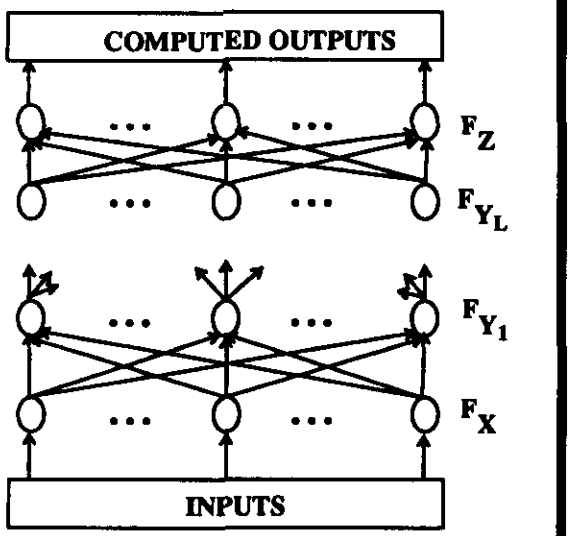
A multi-layer neural network has more than two layers, possibly many more. A general description of a multi-layer neural network is shown in Figure 11, where there is an input layer of PEs, F_X, L hidden layers of F_Y PEs $(Y_1, Y_2, ..., Y_L)$, and a final output layer, F_Z. The F_Y layers are called hidden layers because there are no direct connections between the input/output patterns to these PEs, rather they are always accessed through another set of PEs such as the input and output PEs. Although Figure 11 shows connections only from one layer to the next, it is possible to have connections that skip over layers, that connect the input PEs to the output PEs, or that connect PEs together within the same layer. The added benefit of these PEs is not fully understood, but many applications such as prediction and classification are employing these types of topologies.

Multi-layer neural networks are used for pattern classification, pattern matching and function approximation. By adding a continuously differentiable threshold function, such as a Gaussian or sigmoid function, it is possible to learn practically any nonlinear mapping to any desired degree of accuracy (White, 1989). The mechanism that allows such complex mappings to be acquired is not fully understood for each type of multi-layer neural network, but in general the network partitions the input space into regions and a mapping from the partitioned regions to the next space is performed by the next set of connections to the next layer of PEs, eventually producing an output response. This capability allows some very complex decision regions to be performed for classification and pattern matching problems, as well as applications that require function approximation.

There are several issues that must be

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Figure 11: General Mulit-layer Neural Network



addressed when working with multi-layer neural networks. How many layers is enough for a given problem? How many PEs are needed in each hidden layer? How much data is needed to produce a sufficient mapping from the input layer to the output layer? Some of these issues have been successfully dealt with. As an example, there have been several researchers that have proven that three layers is sufficient to perform any nonlinear mapping (with the exception of a few remote pathological cases) to any desired degree of accuracy with only one layer of hidden PEs (see White, 1989 for a review of this work). Although this is a very important result, it still does not indicate what the proper number of hidden layer PEs is, or if the same solution can be obtained with more layers but fewer hidden PEs and connections overall.

There are several ways that multi-layer neural networks can have their connection weights adjusted to learn mappings. The most popular technique is the backpropagation algorithm (Werbos, 1974; Parker, 1982; Rumelhart, Hinton & Williams, 1986) and its many variants (see Simpson, 1990a for a list). Other multi-layer networks include the Neocognitron (Fukushima, 1988), the Probabilistic Neural Network (Specht, 1990), the Boltzmann Machine (Ackley, Hinton & Sejnowski, 1985), and the Cauchy Machine (Szu, 1986).

4.6. Randomly Connected Networks

Randomly connected neural networks are networks that have connection weights that are randomly assigned within a specific range. Some randomly connected networks have binary valued connections. Realizing that, a connection weight equal to zero is equivalent to no connection being present, binary valued random connections create sparsely connected networks. Randomly connected networks are used in three different ways:

- Initial weights - The initial connection values for the network prior to training are preset to random values within a predefined range. This technique is used extensively in error-correction learning systems (see §5.5 - §5.6. below).
- Pattern preprocessing - A set of fixed random binary valued connections are placed between the first two layers of a multi-layer neural network as a pattern preprocessor. The use of such random connections can be used to increase the dimensionality of the space that is being used for mappings in an effort to improve the pattern mapping capability. This approach was pioneered with the early Perceptron (Rosenblatt, 1962) and has been used recently in the Sparse Distributed Memory (Kanerva, 1988).
- Intelligence from randomness - Early studies in neural networks spent a great deal of effort analyzing randomly connected binary valued systems. The model of the brain as a randomly connected network of neurons prompted this research. These fixed weight, non-adaptive systems have been studied extensively by Amari (1971) and Rozonoer (1969).

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

5. NEURAL NETWORK LEARNING

Perhaps the most appealing quality of neural networks is their ability learn. Learning, in this context, is defined as a change in connection weight values that results in the capture of information that can later be recalled. There are several different procedures available for changing the values of connection weights. After an introduction to some terminology, eight different learning methods will be described. For continuity of discussion, the learning algorithms will be described in pointwise notation (as opposed to vector notation). In addition, the learning algorithms will be described using discrete time equations (as opposed to continuous time). The use of discrete-time equations makes them more accessible to digital computer simulations.

5.1. Terminology

5.1.1. Supervised vs. Unsupervised Learning

All learning methods can be classified into two categories, supervised learning and unsupervised learning. Supervised learning is a process that incorporates an external teacher and/or global information. The supervised learning algorithms that will be discussed in the following sections include error correction learning, reinforcement learning, stochastic learning, and hardwired systems. Examples of supervised learning include; deciding when to turn off the learning, deciding how long and how often to present each association for training, and supplying performance (error) information. Supervised learning is further classified into two subcategories; structural learning and temporal learning. Structural learning is concerned with finding the best possible input-output relationship for each individual pattern pair. Examples of structural learning include pattern matching and pattern classification. The majority of the learning algorithms discussed below focus on structural learning. Temporal learning is concerned with capturing a sequence of patterns necessary to achieve some final outcome. In temporal learning the current response of the network is dependant on previous inputs and responses. In structural learning, there is no such dependance. Examples of temporal learning include prediction and control. The reinforcement learning algorithm discussed below is an example of a temporal learning procedure.

Unsupervised learning, also referred to as self-organization, is a process that incorporates no external teacher and relies upon only local information during the entire learning process. Supervised learning organizes presented data and discovers its emergent collective properties. Examples of unsupervised learning that will be discussed in the following sections includes Hebbian learning, principle component learning, differential Hebbian learning, min-max learning, and competitive learning.

5.1.2. Off-line vs. On-line Learning

Most learning techniques utilize off-line learning. When the entire pattern set is used to condition the connections prior to the use of the network, it is called off-line learning. As an example, the backpropagation training algorithm (see §5.7.2.) is used to adjust connections in multi-layer neural network, but it requires thousands of cycles through all the pattern pairs until the desired performance of the network has been achieved. Once the network is performing adequately, the weights are frozen and the resulting network is used in recall mode thereafter. Off-line learning systems have the intrinsic requirement that all the patterns have to be resident for training. Such a requirement does not make it possible to have new patterns automatically incorporated into the network as they occur, rather these new patterns must be added to the entire set of patterns and a retraining of the neural network must be done again.

Not all neural networks perform off-line learning. There are some networks that can add new information "on the fly" non-destructively. If a new pattern needs to be incorporated into the network's connections, it can be done

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

immediately without any loss of prior stored information. The advantage of off-line learning networks is they usually provide superior solutions to difficult problems such as nonlinear classification, but on-line learning allows the neural network to learn in-situ. A challenge in the future of neural network computing is the development of learning techniques that provide high-performance on-line learning without extreme costs.

5.2. Hebbian Correlations

The simplest form of adjusting connection weight values in a neural network is based upon the correlation of PE activation values. The motivation for correlation-based adjustments has been attributed to Hebb (1949) who hypothesized that the change in a synapses efficacy (its ability to fire, or as we are simulating it in our neural networks, the connection weight) is prompted by a neuron's ability to produce an output signal. If a neuron, A, was active, and A's activity caused a connected neuron, B, to fire, then the efficacy of the synaptic connection between A and B should be increased.

5.2.1. Unbounded PE Values and Weights

This form of learning, now commonly referred to as Hebbian learning, has been mathematically characterized as the correlation weight adjustment

$$w_{ij}^{new} = w_{ij}^{old} + a_{ki} b_{kj} \qquad (17)$$

where: $i = 1, 2, ..., n$; $j = 1, 2, ..., p$; x_i is the value of the i'th PE in the F_X layer of a two layer network; y_j is the value of the j'th F_Y PE; and the connection weight between the two PEs is w_{ij}. In general, the values of the PEs can range over the real numbers and the weights are unbound. When the PE values and connection values are unbound, these two layer neural networks are amenable to linear systems theory. Neural networks like the Linear Associative Memory (Anderson, 1970; Kohonen, 1972) employ this type of learning and analyze the capabilities of these networks using linear systems theory as a guide. The number of patterns that a network trained using equation (17) with unbounded weights and connections is limited to the dimensionality of the input patterns (cf. Simpson, 1990a).

5.2.2. Bounded PE Values & Unbounded Weights

Recently, implementations that restrict the values of the PEs and/or the weights of equation (17) have been employed. These networks, called Hopfield Networks because John Hopfield had excited people about their potential (Hopfield, 1982), restrict the PE values to either binary {0,1} or bipolar {-1,+1} values. Equation (17) is used for these types of correlations.

These discrete-valued networks typically involve some form of feedback recall, resulting in the need to show that every input will produce a stable response (output). By limiting the PE values during processing, nonlinearities are introduced in the system, eliminating some of the linear systems theory analyses that had previously been performed. By adding feedback into the recall process, a discrete valued, nonlinear, dynamical system is formed. The single layer versions of this learning rule are described as Hopfield nets (Hopfield, 1982) and the two-layer versions as the Bidirectional Associative Memory (Kosko, 1988). Some of the earlier analysis of these networks was performed by Amari (1972 & 1977) who used the theory of statistical neurodynamics to show these networks were stable. Later Hopfield (1982) had found an alternative method to prove stability. Also, the number of patterns that neural networks of this form can store is limited (McEleice, et al., 1987).

5.2.3. Bounded PE Values and Weights

Sometimes both the PE values and the weights are bounded. There are two forms of such systems. The first form is simply a running average of the amount of correlation between

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

two PEs. The equation

$$w_{ij}^{new} = \frac{1}{k}(a_{ki}b_{kj} + (k-1)\,w_{ij}^{old}) \qquad (18)$$

describes the average correlation during the presentation of the k'th pattern pair (A_k, B_k), where: $A_k = (a_{k1}, a_{k2}, ..., a_{kn})$; $B_k = (b_{k1}, b_{k2}, ..., b_{kp})$; and k is current pattern number and k = 1, 2, ..., m. The same information that was stored using equation (17) is stored using equation (18), the connection weights are simply bound to the unit-interval in the latter case.

The other example of a correlation neural network learning equation with bounded PE values and bounded weights is the sparse encoding equation defined as

$$w_{ij}^{new} = \begin{cases} 1 \text{ if } a_{ki}b_{kj} = 1 \\ 1 \text{ if } w_{ij}^{old} = 1 \\ 0 \text{ otherwise} \end{cases} \qquad (19)$$

This equation assigns a binary value to a connection if the PEs on each end of the connection have both had the value of 1 over the course of learning. The learning equation is equivalent to performing the logic operation

$$w_{ij}^{new} = (a_{ki} \cap b_{kj}) \cup w_{ij}^{old} \qquad (20)$$

where $\cap$ and $\cup$ are the intersection and union operations, respectively.

Neural networks that have utilized this form of learning include the Learnmatrix (Steinbuch & Piske, 1963) and the Willshaw Associative Memory (Willshaw, 1980). This learning equation has a great deal of potential. By sparsely encoding information in a binary vector (say for example only 32 components out of 1 million were set to 1, the others were set to 0), it is possible to store a tremendous amount of information in the network. The problem lies in creating the code necessary to perform such dense storage (cf. Hecht-Nielsen, 1990).

5.3. Principle Component Learning

There are some neural networks that have learning algorithms designed to produce, as a set of weights, the principle components of the input data patterns. The principle components of a set of data are found by forming the covariance (or correlation) matrix of a set of patterns and then finding the minimal set of orthogonal vectors that span the space of the covariance matrix. Once the basis set has been found, it is possible to reconstruct any vector in the space with a linear combination of the basis vectors. The value of each scalar in the linear combination represents the "importance" of that basis vectors (Lawley & Maxwell, 1963). It is possible to think of the basis vectors as feature vectors and the combination of these feature vectors is used to construct patterns. Hence, the purpose of a principle component network is to decompose an input pattern into values the represent the relative importance of the features underlying the patterns.

The first work with principle component learning was done by Oja (1982). Oja reasoned that Hebbian learning with a feedback term that automatically constrained the weights could extract the principle components from the input data. The equation Oja uses is

$$w_{ij}^{new} = w_{ij}^{old} + b_{kj}(\alpha a_{ki} - \beta b_{kj} w_{ij}^{old}) \qquad (21)$$

where: a_{ki} is the i'th component of the k'th input pattern A_k, i = 1, 2, ..., n; b_{kj} is the j'th component of the k'th output pattern B_k, j = 1, 2, ..., p; k = 1, 2, ..., m; and α and β are positive constants.

A variant of the work by Oja has been developed by Sanger (1989) and is described by the equation

$$w_{ij}^{new} = w_{ij}^{old} + \gamma_k\left(a_{ki}b_{kj} - b_{kj}\sum_{h=1}^{i} y_h w_{jh}\right) \qquad (22)$$

where the variables are similar to those of equation (21) with the exception of the non-zero, time-decreasing learning parameter γ_k. Equations (21) and (22) are very similar, the key dif-

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

ference is equation (22) includes more information in the feedback term and uses a decaying learning rate. There have been many analyses and applications of principle component networks. For a review of this work, see Oja (1989).

5.4. Differential Hebbian Learning

Hebbian learning has been extended to capture the temporal changes that occur in pattern sequences. This learning law, entitled Differential Hebbian Learning, has been independently derived by Klopf (1986) in the discrete time form and by Kosko (1986b) in the continuous time form. The general form, some variants, and some similar learning laws are outlined in the following sections. There are several other combinations that have been explored beyond those that are presented in this section. A more thorough examination of these Hebbian learning rules and others can be found in Barto (1984) and Tesauro (1986).

5.4.1. Basic Differential Hebbian Learning

Differential Hebbian Learning correlates the changes in PE activation values with the equation

$$w_{ij}(t+1) = w_{ij}(t) + \Delta x_i(t-1)\Delta y_j(t) \quad (23)$$

where: $\Delta x_i(t) = x_i(t) - x_i(t\text{-}1)$ is the amount of change in the i'th F_X PE at time t; and $\Delta y_j(t\text{-}1) = y_j(t\text{-}1) - y_j(t\text{-}2)$ is the amount of change in the j'th F_Y PE at time t-1.

5.4.2. Drive-Reinforcement Learning

Klopf (1986) uses the more general case of this equation that captures changes in F_X PEs over that last k time steps and modulates each change by the corresponding weight value for the connection. Klopf's equation is

$$w_{ij}(t+1) = w_{ij}(t) + \Delta y_j \times \sum_{h=1}^{k} \alpha(t-h)|w_{ij}(t-h)|\Delta x_i(t-h) \quad (24)$$

where: $\alpha(t\text{-}h)$ is a decreasing function of time that regulates the amount of change; and $w_{ij}(t)$ is the connection value from the x_i to y_j at time t. Klopf refers to the pre-synaptic changes, $\Delta x_i(t\text{-}h)$, h = 1, 2, ..., k, as drives and the post-synaptic change, $\Delta y_j(t)$, as the reinforcement, hence the name drive-reinforcement learning.

5.4.3. Covariance Correlation

Sejnowski (1977) has proposed the covariance correlation of PE activation values in the equation

$$w_{ij}^{new} = w_{ij}^{old} + \mu\,[\,(a_{ki} - \bar{x}_i)\,(b_{kj} - \bar{y}_j)\,] \quad (25)$$

where the bracketed terms represent the covariance, the difference between the expected (average) value of the PE activation values and the input and output pattern values. The parameter $0 < \mu < 1$ is the learning rate. The overbar on the PE values represents the average value of the PE.

Sutton & Barto (1981) have proposed a similar type of covariance learning rule, suggesting the correlation of the expected value of xi with the variance of y_j as expressed by the equation

$$w_{ij}^{new} = w_{ij}^{old} + \mu\bar{x}_i\,(b_{kj} - \bar{y}_j) \quad (26)$$

5.5. Competitive Learning

Competitive learning, introduced by Grossberg (1970) and Malsburg (1973) and extensively studied by Amari & Takeuchi (1978), Amari (1983) and Grossberg (1982) is a method of automatically creating classes for a set of input patterns. Competitive learning is a two-step procedure that couples the recall process with the learning process in a two-layer neural network (see Figure 12). In Figure 12 each F_X PE represents a component of the input pattern and each F_Y PE represents a class (see also §4.3.4.).

Step 1: Determine winning F_Y PE. An input pattern, A_k, is passed through the connections from the input layer, F_X, to the output layer, F_Y, in a feedforward fashion using the dot product update equation

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

$$y_j = \sum_{i=1}^{n} a_{ki} w_{ij} \qquad (27)$$

where: x_i is the i'th PE in the input layer F_X, i = 1, 2, ..., n; y_j is the j'th PE in the output layer F_Y, j = 1, 2, ..., p; and w_{ij} is the value of the connection weight between x_i and y_j. Each set of connections that abut a F_Y PE, say y_j, as a reference vector $W_j = (w_{1j}, w_{2j}, ..., w_{nj})$ representing the class j. The reference vector, W_j, that is closest to the input, A_k, should provide the highest activation value. If the input patterns A_k, k = 1, 2, ..., m, and the reference vectors W_j, j = 1, 2, ..., p, are normalized to Euclidean unit length, then the following relationship holds

$$0 \le \left(y_j = A_k \bullet W_j = \sum_{i=1}^{n} a_{ki} w_{ij} \right) \le 1 \qquad (28)$$

where the more similar A_k is to W_j, the closer the dot product is to unity (see §3.4.1.). The dot product values, y_j, are used as the initial values for winner-take-all competitive interactions (see §4.3.4.). The result of these interactions is identical to searching the F_Y PEs and finding the PE with the largest dot product value. Using the equation

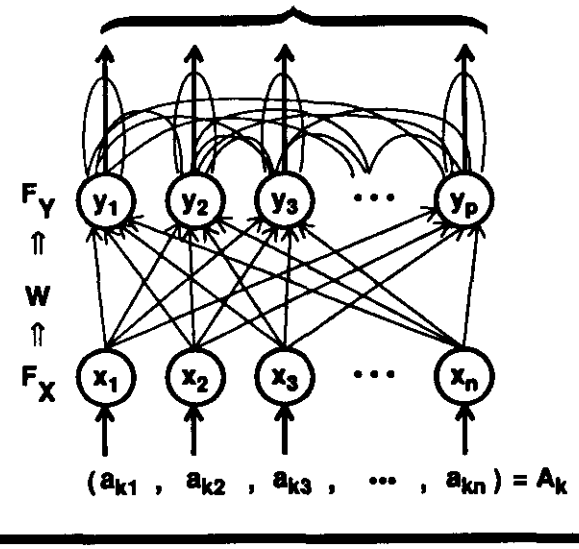
$$y_j = \begin{cases} 1 \text{ if } (y_j > y_k) \text{ for all } (j \neq k) \\ 0 \text{ otherwise} \end{cases} \qquad (29)$$

The F_Y PE with the highest dot product value is called the winning PE. The reference vector associated with the winning PE is the winning reference vector.

Step 2: Adjust winning F_Y PE's connection values. In competitive learning with winner-take-all dynamics like those described above, there is only one set of connection weights adjusted - the connection weights of the winning reference vector. The equation that automatically adjusts the winning reference vector and no others is

$$w_{ij}^{new} = w_{ij}^{old} + \alpha(t) y_j (a_{ki} - w_{ij}) \qquad (30)$$

Figure 12: Competitive Learning Neural Network



where $\alpha(t)$ is a positive, monotonically decreasing function of time. The result of this operation is the motion of the reference vector toward the input vector. Over several presentations of the data vectors (on the order of 10,000 or more), the reference vectors will become the centroids of data clusters (Kohonen, 1986).

There have been several variations of this algorithm (cf. Simpson, 1990a), but one of the most important is the conscience mechanism (DeSieno, 1988). By adding a conscience to each F_Y PE that only allows an F_Y PE to become a winner if it has won an equiprobable number of times. The equiprobable winning constraint improves both the quality of solution and the learning time. Neural networks that employ competitive learning include Learning Vector Quantization (Kohonen, 1984), Self-Organizing Feature Maps (Kohonen, 1984), Adaptive Resonance Theory I (Carpenter & Grossberg, 1987a), and Adaptive Resonance Theory II (Carpenter & Grossberg, 1987b).

5.6. Min-Max Learning

Min-max classifier systems utilize a pair of vectors for each class (see §3.4.3.). For the

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

class j, represented by the PE y_j and defined by the abutting vectors V_j (the min vector) and W_j (the max vector). Learning in a min-max neural system is done using the equation

$$v_{ij}^{new} = \min(a_{ki}, v_{ij}^{old}) \quad (31)$$

for the min vector and

$$w_{ij}^{new} = \max(a_{ki}, w_{ij}^{old}) \quad (32)$$

for the max vector. If the min and max vectors are constrained to lie between 0 and 1 along each dimension, it is possible to think of each reference vector as a fuzzy set (Simpson, 1990b). Within this framework, the fuzzy intersection of two vectors, A_k & V_j, is represented by equation (31) and the fuzzy union of two vectors, A_k & W_j, is represented by equation (32).

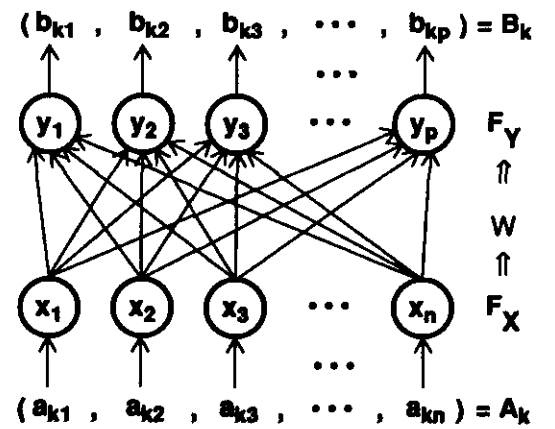
5.7. Error Correction Learning

Error correction learning adjusts the connection weights between PEs in proportion to the difference between the desired and computed values of each output layer PE. Two layer error correction learning is able to capture linear mappings between input and output patterns. Multi-layer error correction learning is able to capture nonlinear mappings between the inputs and outputs. In the following two sections, each of these learning techniques will be described.

5.7.1. Two-Layer Error Correction Learning

Consider the two-layer network shown in Figure 13. Assume that the weights, W, are initialized to small random values (see §4.6.). The input pattern, A_k, is passed through the connections weights, W, to produce a set of F_Y PE values, $Y = (y_1, y_2, ..., y_p)$. The difference between the computed output values, Y, and the desired output pattern values, B_k, is the error. Computing the error for each F_Y PE is done using the equation

Figure 13: Two-Layer Network



$$\delta_j = b_{kj} - y_j \quad (33)$$

The error is used to adjust the connections weights using the equation

$$w_{ij}^{new} = w_{ij}^{old} + \alpha \delta_j a_{ki} \quad (34)$$

where the positive valued constant α is the learning rate

The foundations for the learning rule described by equations (33) and (34) are solid. By realizing that the best solution can be attained when all the errors for a given pattern across all the output PEs, y_j, is minimized, the following cost function can be constructed

$$E = \frac{1}{2} \sum_{j=1}^{p} (b_{kj} - y_j)^2 \quad (35)$$

When E is zero, the mapping from input to output is perfect for the given pattern. By moving in the opposite direction of the gradient of the cost function with respect to the weights, the optimal solution can be achieved (assuming each movement along the gradient, α, is sufficiently small). Restated mathematically, the two-layer error correction learning algorithm is computed as follows

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\left[\frac{1}{2}\sum_{j=1}^{p}\left(b_{kj} - \sum_{i=1}^{n} a_{ki}w_{ij}\right)^2\right]$$

$$= \left(b_{kj} - \sum_{i=1}^{n} a_{ki}w_{ij}\right)a_{ki}$$

$$= (b_{kj} - y_j)\,a_{ki} \qquad (36)$$

Although the cost function is only with respect to a single pattern, it has been shown (Widrow & Hoff, 1960) that the motion in the opposite direction of the gradient for each pattern, when taken in aggregate, acts as a noisy gradient motion that still achieves the proper end result.

The Perceptron (Rosenblatt, 1962) and the Adaline (Widrow & Hoff, 1960), two of the most prominent early neural networks, employed error correction learning. In addition, the Brain-State-in-a-Box (Anderson, et al., 1977) uses the two-layer error correction procedure described above for one-layer autoassociative encoding.

5.7.2. Multi-layer Error Correction Learning

A problem that once plagued error correction learning was its inability to extend learning beyond a two-layer network. By remaining a two-layer learning rule, only linear mappings could be acquired. There had been several attempts to extend the two-layer error correction learning algorithm to multiple layers, but the same problem kept arising: How much error is each hidden layer PE responsible for the output layer PE error? Using the three-layer neural network in Figure 14 to explain, the problem of multi-layer learning (in this case three-layer learning) was calculating the amount of error each hidden layer PE, y_j, should be credited for an output layer PE's error. This problem, called the credit assignment problem (Barto, 1984; Minsky, 1961), was solved through the realization that a continuously differentiable threshold function for the hidden layer PEs would allow the chain rule of partial differentiation to be used to calculate weight changes for any weight in the network. Using the three layer network in Figure 14 to illustrate the multi-layer error correction learning algorithm, the output error across all the F_Z PEs is found using the cost function

$$E = \frac{1}{2}\sum_{j=1}^{q}(b_{kj} - z_j)^2 \qquad (37)$$

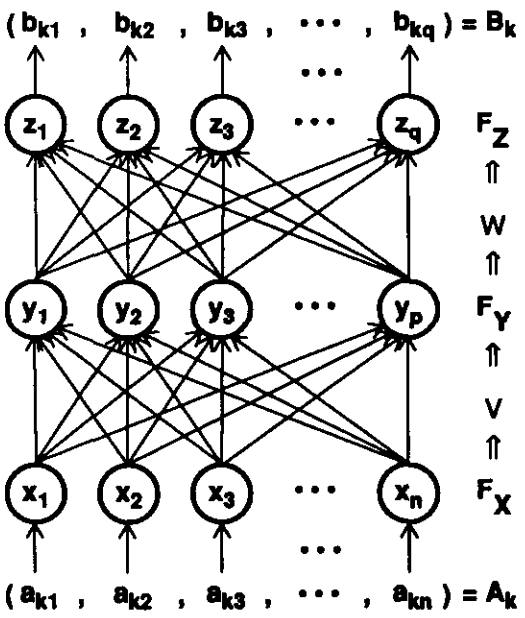
The output of a F_Z PE, z_j, is computed using the equation

$$z_j = \sum_{i=1}^{p} y_i w_{ij} \qquad (38)$$

and each F_Y (hidden layer) PE, y_i, is computed using the equation

$$y_i = f(r_i); \qquad r_i = \sum_{h=1}^{n} a_{kh}v_{hi} \qquad (39)$$

Figure 14: Three-Layer Network



This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

and the hidden layer PE threshold function is

$$f(\gamma) = \frac{1}{1+e^{-\gamma}} \tag{40}$$

Using the same principle as described in the previous section, the weight adjustments will be performed by moving along the cost function in the opposite direction of the gradient to a minimum (where the minimum is considered to be the input-output mapping producing the smallest amount of total error). The connection weights between the F_Y and F_Z PEs are adjusted using the same form of equation derived earlier for two-layer error correction learning, yielding

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\left[\frac{1}{2}\sum_{j=1}^{q}(b_{kj}-z_j)^2\right]$$

$$= (b_{kj}-z_j)\,y_i$$

$$= \delta_j y_i \tag{41}$$

Next, the adjustments to the connection weights between the F_X and F_Y PEs are found using the chain rule of partial differentiation, yielding

$$\frac{\partial E}{\partial v_{hi}} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial r_i}\frac{\partial r_i}{\partial x_h}\frac{\partial x_h}{\partial v_{hi}}$$

$$= \sum_{l=1}^{p}(b_{kl}-y_l)\,y_l w_{hl} f'(r_i) a_{kh} \tag{42}$$

The multi-layer version of this algorithm is commonly referred to as the backpropagation of errors learning rule, or simply backpropagation. Utilizing the chain rule, it is possible to calculate weight changes for an arbitrary number of layers. The number of iterations that must be performed for each pattern in the data set is large, making this off-line learning algorithm very slow to train. Using equation (41) and (42), the weight adjustment equations are

$$w_{ij}^{new} = w_{ij}^{old} - \alpha\frac{\partial E}{\partial w_{ij}} \tag{43}$$

and

$$v_{hi}^{new} = v_{hi}^{old} - \beta\frac{\partial E}{\partial v_{hi}} \tag{44}$$

where α and β are positive valued constants that regulate the amount of adjustments made with each gradient move.

Extending the backpropagation to utilize mean-variance connections (see §3.4.2.) between the F_X and F_Y PEs is straightforward (Robinson, Niranjan & Fallside, 1988). Figure 15 shows the topology of a three-layer mean-variance version of the multi-layer error correction learning algorithm. The hidden layer, F_Y, PE values are computed with the equation

$$y_l = g(r_i);\ r_i = \sum_{h=1}^{n}\left(\frac{u_{hi}-a_{kh}}{v_{hi}}\right)^2 \tag{45}$$

where u_{hi} represents the mean connection strength between the h'th F_X and i'th F_Y PEs, v_{hi} is the variance connection strength between the h'th F_X and i'th F_Y PEs, and the threshold function is the Gaussian function

$$g(x) = e^{-x/2} \tag{46}$$

The output PE, F_Z, values are then formed from the linear combination of the hidden layer Gaussians using the equation

$$z_j = \sum_{i=1}^{p} y_i w_{ij} \tag{47}$$

where w_{ij} is the connection strength between the i'th F_Y and j'th F_Z PEs. Computing the gradients for each set of weights yields the following set of equations

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

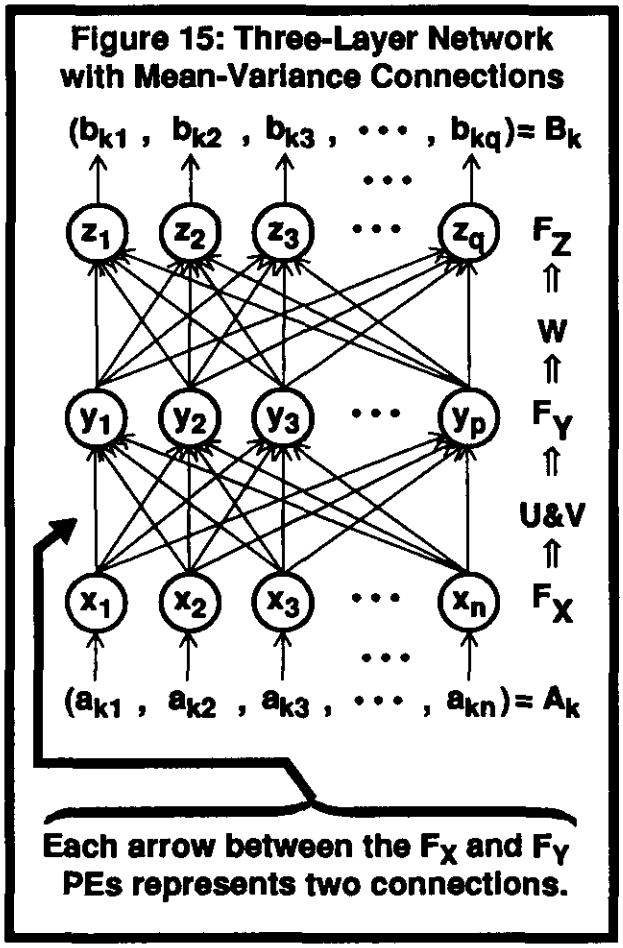


Figure 15: Three-Layer Network with Mean-Variance Connections

$$\frac{\partial E}{\partial u_{hi}} = \frac{\partial E}{\partial z_j}\frac{\partial z_j}{\partial y_i}\frac{\partial y_i}{\partial r_i}\frac{\partial r_i}{\partial u_{hi}}$$

$$= \sum_{j=1}^{q} (b_{kj} - z_j)\, w_{ij} g'(r_i) \left(\frac{u_{hi} - a_{ki}}{v_{hi}^2} \right) \quad (48)$$

$$\frac{\partial E}{\partial v_{hi}} = \frac{\partial E}{\partial z_j}\frac{\partial z_j}{\partial y_i}\frac{\partial y_i}{\partial r_i}\frac{\partial r_i}{\partial v_{hi}}$$

$$= \sum_{j=1}^{q} (b_{kj} - z_j)\, w_{ij} g'(r_i) \left(\frac{u_{hi} - a_{ki}}{v_{hi}^3} \right) \quad (49)$$

$$\frac{\partial E}{\partial w_{ij}} = (b_{kj} - z_j)\, y_i \quad (50)$$

Using these equations, the update equations are then

$$u_{hi}^{new} = u_{hi}^{old} - \alpha \frac{\partial E}{\partial u_{hi}} \quad (51)$$

$$v_{hi}^{new} = v_{hi}^{old} - \beta \frac{\partial E}{\partial v_{hi}} \quad (52)$$

$$w_{ij}^{new} = w_{ij}^{old} - \gamma \frac{\partial E}{\partial w_{ij}} \quad (53)$$

where α, β, γ are positive valued constants that regulate the amount of adjustments made with each gradient move.

The backpropagation algorithm was introduced by Werbos (1974), and later independently rediscovered by Parker (1982) and Rumelhart, Hinton, and Williams (1986). The algorithm presented here has been brief. There are several variations on the algorithm (cf. Simpson, 1990a) including: alternative multi-layer topologies, methods of improving the learning time, methods for optimizing the number of hidden layers and the number of hidden layer PEs in each hidden layer, and many more. Although there are many issues that remain unresolved with the backpropagation of errors learning procedure, such as proper number of training parameters, the existence of local minima during training, the extremely long training time, and the optimal number and configuration of hidden layer PEs, the ability for this learning method to automatically capture non- linear mappings remains a significant strength.

5.8. Reinforcement Learning

The initial idea for reinforcement learning was introduced by Widrow, Gupta & Maitra (1973) and has been championed by Williams (1986). Reinforcement learning is similar to error correction learning in that weights are reinforced for properly performed actions and punished for poorly performed actions. The difference between these two supervised learning techniques is that error correction learning utilizes more specific error information by collecting error values from each output layer PE, while reinforcement learning uses non-specific

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

error information to determine the performance of the network. Where error-correction learning has a whole vector of values that it uses for error correction, only one value is used to describe the output layer's performance during reinforcement learning. This form of learning is ideal in situations where specific error information is not available, but overall performance information is, such as prediction and control.

A two-layer neural network such as the one found in Figure 16 serves as a good framework for the reinforcement learning algorithm. The general reinforcement learning equation is

$$w_{ij}^{new} = w_{ij}^{old} + \alpha (r - \theta_j) e_{ij} \qquad (54)$$

where, r is the scalar success/failure value provided by the environment, θ_j is the reinforcement threshold value for the j'th F_Y PE, e_{ij} is the canonical eligibility of the weight from the i'th F_X PE to the j'th FY PE, and $0 < \alpha < 1$ is a constant-valued learning rate. In error correction learning, gradient descent in error space controlled learning. In reinforcement learning it is gradient descent in probability space. The canonical eligibility of w_{ij} is dependant on a previously selected probability distribution that is used to determine if the computed output value equals the desired output value and is defined as

$$e_{ij} = \frac{\partial}{\partial w_{ij}} \ln g_i \qquad (55)$$

where g_i is the probability of the desired output equalling the computed output, defined as

$$g_i = \Pr (y_j = b_{kj} \mid W_j, A_k) \qquad (56)$$

which is read as the probability that y_j equals b_{kj} given the input, A_k, and the corresponding weight vector, W_j.

Neural networks that employ reinforcement learning include the Adaptive Heuristic Critic (Barto, Sutton & Anderson, 1983) and the Associative Reward-Penalty neural network (Barto, 1985).

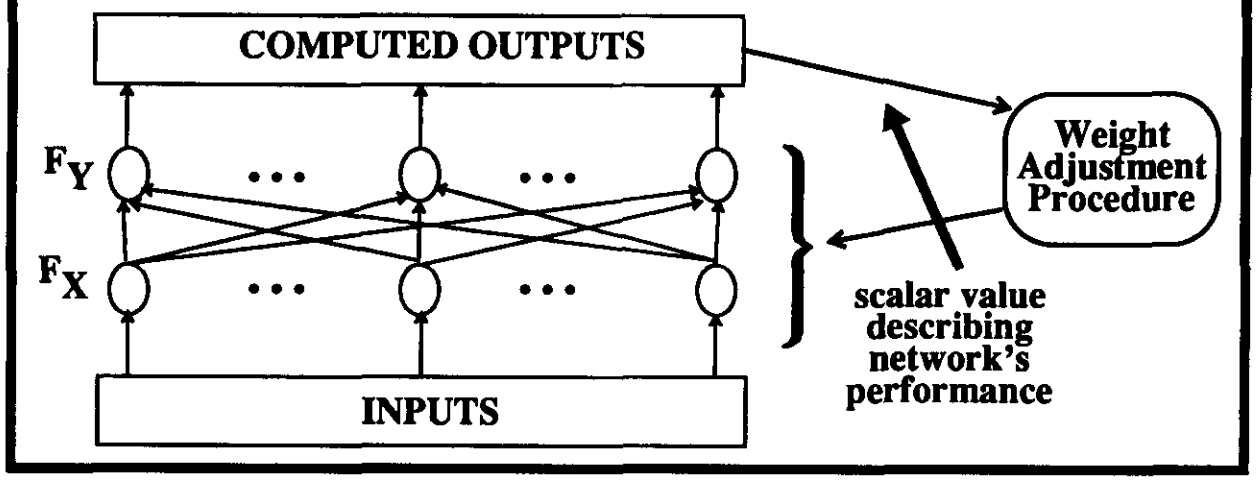
5.9. Stochastic Learning

Stochastic learning uses random processes, probability, and an energy relationship to adjust connection weights in a multi-layered neural network. Using the three-layer neural network shown in Figure 14 to illustrate the learning algorithm, the stochastic learning procedure is described as follows:

1. Randomly change the output value of a hidden layer PE (the hidden layer PEs utilize a binary step threshold function).

2. Evaluate the change using the resulting difference in the neural network's energy as a

Figure 16: Reinforcement Learning Neural Network



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

guide. If the energy after the change is lower, keep the change. If the change in energy is not lower after the random change, accept the change according to a pre-chosen probability distribution.

3. After several random changes, the network will eventually become "stable." Collect the values of the hidden layer PEs and the output layer PEs.

4. Repeat steps 1-3 for each pattern pair in the data set, then use the collected values to statistically adjust the weights.

5. Repeat steps 1-4 until the network performance is adequate.

The probabilistic acceptance of higher energy states, despite poorer performance, allows the neural network to escape local energy minima in favor of a deeper energy minimum. This learning process, founded in simulated annealing (Kirkpatrick, Gelatt & Vecchi, 1983), is governed by a "temperature" parameter that slowly decreases the number of probabilistically accepted higher energy states.

The Boltzmann Machine (Ackley, Hinton & Sejnowski, 1985) was the first neural network to employ stochastic learning. Szu (1986) has refined the procedure by employing the Cauchy distribution function in place of the Gaussian distribution function, resulting in a network that converges to a solution much quicker.

5.10. Hardwired Systems

There are some neural networks that have their connection weights predetermined for a specific problem. These weights are "hardwired" in that they do not change once they have been determined. The most popular hardwired systems are the neural optimization networks (Hopfield & Tank, 1985). Neural optimization works by designing a cost function that, when minimized, solves an unconstrained optimization problem. By translating the energy function into a set of weights and bias values, the neural network becomes a parallel optimizer. Given the initial values of the problem, the network will run to a stable solution. This technique has been applied to a wide range of problems (cf. Simpson, 1990a), including scheduling, routing and resource optimization (see §4.3.3.).

Two other types of hardwired networks include the Avalanche Matched Filter (Grossberg, 1969; Hecht-Nielsen, 1990) and the Probabilistic Neural Network (Specht, 1990). These networks are considered hardwired systems because the data patterns are normalized to unit length and used as connection weights. Despite the lack of an adaptive learning procedure, each of these neural networks are very powerful in their own right.

5.11. Summary of Learning Procedures

There are several attributes of each of the neural network learning algorithms that have been described. Table 1 describes six key attributes of the learning procedures described above:

- Training Time - How long does it take the learning technique to adequately capture information (quick, slow, very slow, and extremely slow)?
- On-Line/Off-Line - Is the learning technique an on-line or an off-line learning algorithm?
- Supervised/Unsupervised - Is the learning technique a supervised or unsupervised learning procedure?
- Linear/Nonlinear - Is the learning technique capable of capturing nonlinear mappings?
- Structural/Temporal - Does the learning algorithm capture structural information, temporal information, or both?
- Storage Capacity - Is the information storage capacity good relative to the number of connections in the network?

The information provided in Table 1 is meant

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

as a guide and is not intended to be a precise description of the qualities of each neural network. For a more detailed description of each neural network learning algorithm, please refer to Simpson, 1990a, Hecht- Nielsen, 1990, or Maren, Harsten & Pap, 1990.

6. NEURAL NETWORK RECALL

The previous section emphasized the storage of information through a wide range of learning procedures. In this section, the emphasis is retrieving information already stored in the network. Some of the recall equations have been introduced as a part of the learning process. Others will be introduced here for the first time. The recall techniques described here fall into two broad categories: feedforward recall and feedback recall.

6.1. Feedforward Recall

Feedforward recall is performed in networks that do not have feedback connections. The most common feedforward recall technique is the linear combiner (see §3.4.1.) followed by a threshold function

$$y_j = f(\sum_{i=1}^{n} x_i w_{ij}) \tag{57}$$

where the threshold function f is one of those described in §3.5.

For a feedforward network using dual connections (see §3.4.2.) where one set of connection weights, W, represents the mean and the other set of connection weights, V, represents the variance, the recall equation is

$$y_j = g(\sum_{i=1}^{n} \left(\frac{w_{ij} - x_i}{v_{ij}}\right)^2) \tag{58}$$

where g is the Gaussian threshold function (see §3.5.5.).

For a feedforward network using dual connections where one set of connection weights, V, represents the min vector and the other set of connection weights, W, represents the max vector (see §3.4.3.), and the system is confined to the unit hypercube, the recall equation is

$$\begin{aligned} y_j = {} & (1 - \text{supersethood}(X, W_j)) \\ & \times (1 - \text{subsethood}(X, V_j)) \\ = {} & \text{subsethood}(X, W_j) \\ & \times \text{supersethood}(X, V_j) \end{aligned} \tag{59}$$

where the supersethood operation is defined as

$$\text{supersethood}(X, Y) = \frac{\sum_{i=1}^{n} \max(0, x_i - y_i)}{\sum_{i=1}^{n} x_i} \tag{60}$$

Table 1: Neural Network Learning Algorithms

Learning Algorithm	Training Time	On-Line/ Off-Line	Supervised/ Unsupervised	Linear/ Nonlinear	Structural/ Temporal	Storage Capacity
Hebbian Learning	Fast	On-line	Unsupervised	Linear	Structural	Poor
Principle Component Learning	Slow	Off-line	Unsupervised	Linear	Structural	Good
Differential Hebbian Learning	Fast	On-line	Unsupervised	Linear	Temporal	Undetermined
Competitive Learning	Slow	On-line	Unsupervised	Linear	Structural	Good
Min-Max Learning	Fast	On-line	Unsupervised	Linear	Stuctural	Good
Two-Layer Error Correction Learning	Slow	Off-line	Supervised	Linear	Both	Good
Multi-Layer Error Correction Learning	Very Slow	Off-line	Supervised	Nonlinear	Both	Very Good
Reinforcement Learning	Extremely Slow	Off-line	Supervised	Nonlinear	Both	Good
Stochastic Learning	Extremely Slow	Off-line	Supervised	Nonlinear	Structural	Very Good
Hardwired Systems	Fast	Off-line	Supervised	Nonlinear	Structural	Good

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Referring to Figure 5, equation (59) measures the degree to which the input pattern A_k falls between the min and max vectors of class j, where a value of 1 means that A_k falls completely between V_j and W_j, and the closer y_j is to 0, the greater the disparity between A_k and the class j, with a value of 0 meaning that A_k is completely outside of the class.

6.2. Feedback Recall

Those networks that have feedback connections employ a feedback recall equation of the form

$$x_j(t+1) = (1-\alpha)\,x_j(t) + \beta \sum_{i=1}^{n} f(x_i(t))w_{ij} + a_{ki} \quad (61)$$

where $x_j(t+1)$ is the value of the j'th element in a single-layer neural network at time t+1, f is a monotonic non-decreasing function (e.g. sigmoid function), α is a positive constant that regulates the amount of decay a PE value has during a unit interval of time, β is a positive constant that regulates the amount of feedback the other PEs provide the j'th PE, and a_{ki} is the constant valued input from the i'th component of the k'th input pattern.

One issue that arises in feedback recall systems is stability. Stability is achieved when a network's PEs cease to change in value after they have been given an initial set of inputs, Ak, and have processed for a while. If the network did not stabilize, it would not be of much use. Ideally, the initial inputs to the feedback neural network would represent the input pattern and the stable state that the network reached would represent the nearest neighbor output of the system.

An important theorem was presented by Cohen & Grossberg (1983) that proved for a wide class of neural networks under a set of minimal constraints, the network would become stable in a finite period of time given any initial conditions. This theorem dealt with systems that had weights that were fixed. In an extension to the Cohen- Grossberg Theorem, Kosko (1990) showed that a neural network could learn and recall at the same time, and yet still remain stable.

6.3. Interpolation vs. Nearest-Neighbor Responses

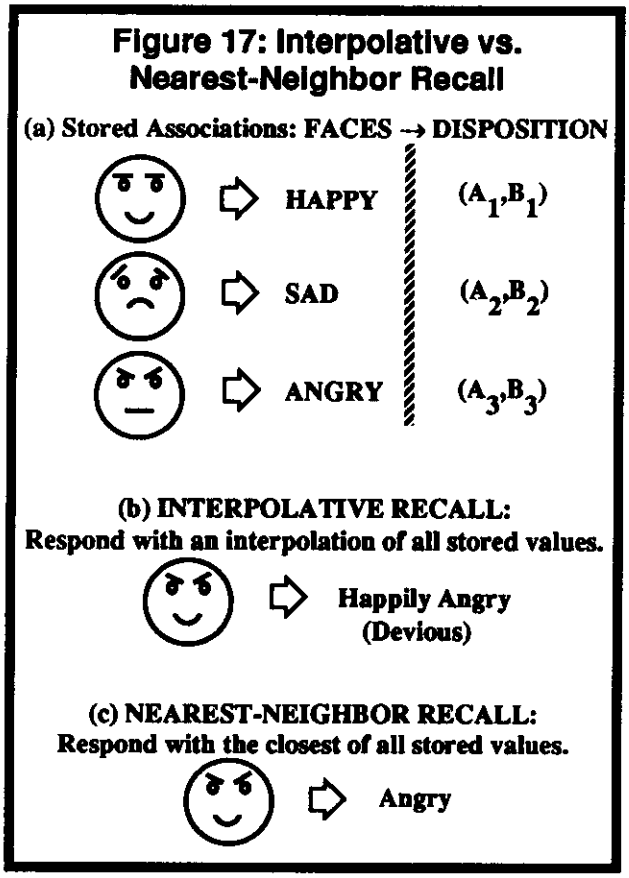
In addition to recall operations being either feedforward or feedback, there is another important attribute associated with recall: output response. There are two types of neural network output response: nearest-neighbor and interpolative. Figure 17 illustrates the difference. Assume that the three face/disposition pairs shown in Figure 17(a) have been stored in a neural network. If an input that is a combination of two of the faces is presented to the network, there are two ways that a neural network might respond. If the output is a combination of the two correct outputs associated with the given inputs, then the network has performed an interpolation (see Figure 17(b)). On the contrary, the network might determine which of the stored faces is most closely associated with the input and respond with the associated output for that face (see Figure 17(c)). The feedforward pattern matching neural networks are typically interpolative response networks (eg. Backpropagation and Linear Associative Memory). The feedforward pattern classification networks (eg. Learning Vector Quantization) and the feedback pattern matching networks (eg. Hopfield Network and Bidirectional Associative Memory) are typically nearest- neighbor response networks.

7. NEURAL NETWORK TAXONOMY

Several different topologies, learning algorithms, and recall equations have been described. Attempts at organizing the various configurations quickly becomes unwieldy unless some simple, yet accurate, taxonomy can be applied. The two most prevalent aspects of neural networks, learning supervision and

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Figure 17: Interpolative vs. Nearest-Neighbor Recall



information flow, seem ideally suited to address this need. Table 2 utilizes these criteria to organize the neural networks described above into a matrix with learning supervision on the ordinate and recall information flow on the abscissa.

8. COMPARING NEURAL NETS TO OTHER INFORMATION PROCESSING METHODS

There are several information processing techniques that have capabilities similar to the neural network learning algorithms described above. Despite the possibility of equally comparable solutions to a given problem, there are several addition aspects of a neural network solution that are appealing including: fault- tolerance through the large number of connections, parallel implementations that allow fast processing, and on-line adaptation that allows the networks to constantly change according to the needs of the environment. The following sections briefly describe some of the alternative methods that are used for pattern recognition, clustering, control, and statistical analysis.

8.1. Stochastic Approximation

The method of stochastic approximation was first introduced by Robbins and Monro (1951) as a method for finding a mapping between inputs and outputs when the inputs and outputs are extremely noisy (i.e. the inputs and outputs are stochastic variables). The stochastic approximation technique has been shown to be identical to the two-layer error correction algorithm presented in §5.7.1. (Kohonen, 1984) and the three-layer error correction algorithm presented in §5.7.2. (White, 1989).

8.2. Kalman Filters

A Kalman Filter is a technique for estimating, or predicting, the next state of a system based upon a moving average of measurements driven by additive white noise. The Kalman Filter requires a model of the relationship between the inputs and the outputs to provide feedback that allows the system to continuous perform its estimation. Kalman filters are primarily used for control systems. Singhal and Wu (1989) have developed a method of using a Kalman filter to train the weights of a multilayer neural network. In some recent work, Ruck, et al. (1990) have shown that the backpropagation algorithm is a special case of the Extended Kalman Filter algorithm and have provided several comparative examples of the two training algorithms on a variety of data sets.

8.3. Linear and Nonlinear Regression

Linear regression is a technique for fitting a line to a set of data points such that the total distance between the line and the data points is minimized. This technique, used widely in statistics (Spiegel, 1975), is similar to the two-layer error correction learning algorithm described in §5.7.1.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Table 2: Neural Network Taxonomies

LEARNING	RECALL INFORMATION FLOW: Feedback	Feedforward
Unsupervised	Hopfield Networks (Amari, 1972; Hopfield, 1982) ART1 & ART2 (Carpenter & Grossberg, 1987a & 1987b) Bidirectional Associative Memory (Kosko, 1988; Simpson, 1990c) Principle Component Networks (Oja, 1982; Sanger, 1989)	Linear Associative Memory (Anderson, 1970; Kohonen, 1972) Associative Reward-Penalty (Barto, 1985) Adaptive Heuristic Critic (Barto, Sutton & Anderson, 1983) Drive-Reinforcement Learning (Klopf, 1986) Learning Vector Quantization (Kohonen, 1984) Fuzzy Min-Max Classifier (Simpson, 1990b) Learnmatrix (Steinbuch & Piske, 1963; Willshaw, 1980)
Supervised	Brain-State-in-a-Box (Anderson, et al., 1977) Neural Optimization (Hopfield & Tank, 1985)	Boltzmann Machine (Ackley, Hinton & Sejnowski, 1985) Neocognitron (Fukushima, 1988) Avalanche Matched Filter (Grossberg, 1969; Hecht-Nielsen, 1990) Sparse Distributed Memory (Kanerva, 1988) Gaussian Potential Function Network (Lee & Kil, 1989) Backpropagation (Werbos, 1974; Parker, 1982, Rumelhart, et al., 1986) Perceptron (Rosenblatt, 1962) Probabilistic Neural Network (Specht, 1990) Cauchy Machine (Szu, 1986) Adaline (Widrow & Hoff, 1960)

Nonlinear regression is a technique for fitting curves (nonlinear surfaces) to data points. White (1990) points out that the threshold function used in many error correction learning algorithms is a family of curves and the adjustment of the weights that minimizes the overall mean-squared-error is equivalent to curve fitting. In this sense, the backpropagation algorithm described in §5.7.2 is an example of an automatic nonlinear regression technique.

8.4. Correlation

Correlation is a method of comparing two patterns. One pattern is the template and the other is the input. The correlation between the two patterns is the dot product. Correlation is used extensively in pattern recognition (Young & Fu, 1986) and signal processing (Elliot, 1987). In pattern recognition the templates and inputs are normalized, allowing the dot product operation to provide similarities based upon the angles between vectors. In signal processing the correlation procedure is often used for comparing templates with a time-series to determine when a specific sequence occurs (this technique is commonly referred to as cross-correlation or matched filters). The Hebbian learning techniques described in §5.2. are correlation routines that store correlations in a matrix and compare the stored correlations with the input pattern using inner products.

8.5. Bayesian Classification

The purpose of pattern classification is to determine which class a given pattern belongs. If the class boundaries are not cleanly separated and tend to overlap, the classification system must find the boundary between the classes that minimizes the average misclassification (error). The smallest possible error relative to a predefined risk is referred to as the Bayes error, and a classifier that minimizes Bayes error is called a Bayesian classifier (Fukunuga, 1986). The Parzen approach to implementing a Bayesian classifier utilizes a uniform kernel (typically the Gaussian function) to approximate the probability density function of the data. A neural network implementation of this approach (see §4.5.) is the Probabilistic Neural Network (Specht, 1990).

8.6. Vector Quantization

The purpose of vector quantization is produce a code from an n- dimensional input pattern. The code is passed across a channel and then used to reconstruct the original input with a minimum amount of distortion. There have been several techniques proposed to perform vector quantization (Gray, 1984), with one of the most successful being the LBG algorithm (Linde, Buzo & Gray, 1980). The Learning Vector Quantization (see §5.5.) is a method of developing a set of reference vectors from a data set and is very similar to the LBG algorithm. A comparison of these two techniques can be found in Ahalt, et al. (1990).

8.7. Radial Basis Functions

A radial basis function is a function that is symmetric about a given mean (e.g. a Gaussian

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

function). In pattern classification a radial basis function is used in conjunction with a set of n-dimensional reference vectors, where each reference vector has a radial basis function that constrains its response. An input pattern is processed through the basis functions to produce an output response. The mean-variance connection topologies that employ the backpropagation algorithm (Lee & Kil, 1989; Robinson, Niranjan, & Fallside, 1988) as described in §5.7.2. are methods of automatically producing the proper sets of basis functions (by adjustment of the variances) and their placement (by adjustment of their means).

8.8. Machine Learning

Neural networks are not the only method of learning that has been proposed for machines (although it is the most biologically related). There are a large number of machine learning procedures that have been proposed over the course of the past thirty years. Carbonell (1990) classifies machine learning into four major paradigms (pg. 2): "[I]nductive learning (e.g., acquiring concepts from sets of positive and negative examples), analytic learning (e.g., explanation-based learning and certain forms of analogical and case-based learning methods), genetic algorithms (e.g., classifier systems), and connectionist learning methods (e.g., nonrecurrent "backprop" hidden layer neural networks)." It is possible that some of the near-term applications might find it useful to combine two or more of these machine learning techniques into a coherent solution. It has only been recently that this type of approach has even been considered.

REFERENCES

Ackley, D., Hinton, G. & Sejnowski, T. (1985). A learning algorithm for Boltzmann machines, Cognitive Science, Vol. 9, pp. 147-169.

Ahalt, S., Krishnamurthy, A., Chen, P. & Melton, D. (1990). Competitive learning algorithms for vector quantization, Neural Networks, Vol. 3, pp. 277-290.

Albert, A. & Gardner, L. (1967). Stochastic Approximation and Nonlinear Regression, MIT Press: Cambridge, MA.

Amari, S. (1971). Characteristics of randomly connected threshold- element networks and network systems, Proceedings of the IEEE, Vol. 59, pp. 35-47.

Amari, S. (1972). Learning patterns and pattern sequences by self- organizing nets of threshold elements, IEEE Trans. on Computer, Vol. C-21, pp. 1197-1206.

Amari, S. (1977). Neural theory of association and concept formation, Biological Cybernetics, Vol. 26, pp. 175-185.

Amari, S. (1983). Field theory of self-organizing neural nets, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-13, pp. 741-748.

Amari, S. & Takeuchi, M. (1978). Mathematical theory on formation of category detecting nerve celss, Biological Cybernetics, Vol. 29, pp. 127-136.

Anderson, J. (1970). Two models for memory organization using interactive traces, Mathematical Biosciences, Vol. 8, pp. 137-160.

Anderson, J. (1990). Knowledge representation in neural networks, AI Expert, Fall 1990.

Anderson, J., Silverstein, J., Ritz, S. & Jones, R. (1977). Distinctive features, categorical perception, and probability learning: Some applications of a neural model, Pyschological Review, Vol. 84, pp. 413-451.

Barto, A. (1984). Simulation experiments with goal-seeking adaptive elements, Air Force Wright Aeronatical Laboratory, Technical Report AFWAL-TR-84-1022.

Barto, A. (1985). Learning by statistical cooperation of self- interested neuron-like computing units, Human Neurobiology, Vol. 4, pp. 229-256.

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Barto, A., Sutton, R. & Anderson, C. (1983). Neuron-like adaptive elements that can solve difficult learning control problems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-13, pp. 834-846.

Carbonell, J. (1990). Introduction: Paradigms for machine learning, In Machine Learning: Paradigms and Methods, Carbonell, J. (Ed.), pp. 1-10. MIT/Elsevier, Cambridge, MA.

Carpenter, G. & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine, Computer Vision, Graphics, and Image Understanding, Vol. 37, pp. 54-115.

Carpenter, G. & Grossberg, S. (1987b). ART2: Self-organization of stable category recognition codes for analog input patterns, Applied Optics, Vol. 26, pp. 4919-4930.

Cohen, M. & Grossberg, S. (1983). Absolute stability of global pattern formation and parallel storage by competitive neural networks, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-13, pp. 815-825.

Eberhart, R. (1990). Standardization of neural network terminology, IEEE Transactions on Neural Networks, Vol. 1, pp. 244-245.

Elliot, D., Ed. (1987). Handbook of Digital Signal Processing: Engineering Applications, Academic Press: San Diego, CA.

Fukunuga, K. (1986). Stastical pattern classification, In Handbook of Pattern Recognition and Image Processing, Young, T. & Fu, K. (Eds.), pp. 3-32. Academic Press: San Diego, CA.

Fukushima, K. (1988). Neocognitron: A hierarchical neural network capable of visual pattern recognition, Neural Networks, Vol. 1, pp. 119-130.

Grossberg, S. (1969). On the serial learning of lists, Mathematical Biosciences, Vol. 4., pp. 201-253.

Grossberg, S. (1970). Neural pattern discrimination, Journal of Theoretical Biology, Vol. 27, pp. 291-337.

Grossberg, S. (1982). Studies of Mind and Brain, Reidel: Boston.

Hebb, D. (1949). Organization of Behavior, John Wiley & Sons: New York.

Hecht-Nielsen, R. (1990). Neurocomputing, Addison-Wesley: Reading, MA.

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

Hopfield, J. & Tank, D. (1985). 'Neural' computation of decisions in optimization problems, Biological Cybernetics, Vol. 52, pp. 141- 152.

Kanerva, P. (1988). Sparse Distributed Memory, MIT Press: Cambridge, MA.

Kirkpatrick, S., Gelatt, C. & Vecchi, M. (1983). Optimization by simulated annealing, Science, Vol. 220, pp. 671-680.

Klopf, A. (1986). Drive-reinforcement model of a single neuron function: An alternative to the Hebbian neuron model, In J. Denker (Ed.), AIP Conference Proceedings 151: Neural Networks for Computing, (pp. 265-270). American Institute of Physics: New York.

Kohonen, T. (1972). Correlation associative memory, IEEE Transactions on Computer, Vol. C-21, pp. 353-359.

Kohonen, T. (1984). Self-Organization and Associative Memory, Springer-Verlag: Berlin.

Kosko, B. (1986a). Fuzzy entropy and conditioning, Information Sciences, Vol. 40, pp. 165-174.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Kosko, B. (1986b). Differential Hebbian learing, In J. Denker (Ed.), AIP Conference Proceedings 151: Neural Networks for Computing, (pp. 277-282). American Institute of Physics: New York.

Kosko, B. (1988). Bidirectional associative memories, IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-18, pp. 42- 60.

Kosko, B. (1990b). Unsupervised learning in noise, IEEE Transactions on Neural Networks, Vol. 1, pp. 44-57.

Lawley, D. & Maxwell, A. (1963). Factor Analysis as a Statistical Method, Butterworths: London.

Lee, S. & Kil, R. (1989). Bidirectional continuous associator based on Gaussian potential function network, Proceedings of the IEEE/ INNS International Joint Conference on Neural Networks, Vol. I, pp. 45-54.

Malsburg, C.v.d. (1973). Self-organization of orientation sensitive cells in the striate cortex, Kybernetik, Vol. 14, pp. 85-100.

Maren, A., Harston, C. & Pap, R. (1990). Handbook of Neural Computing Applications, Academic Press: San Diego.

McEliece, R., Posner, E., Rodemich, E. & Venkatesh, S. (1987). The capacity of the Hopfield associative memory, IEEE Transactions on Information Theory, Vol. IT-33, pp. 461-482.

Oja, E. (1982). A simplified neuron model as a principal component analyzer, Journal of Mathematical Biology, Vol. 15, pp. 267-273.

Oja, E. (1989). Neural networks, principle components, and subspaces, International Journal of Neural Networks, Vol. 1, pp. 61-68.

Pao, Y. (1989). Adaptive Pattern Recognition and Neural Networks, Addison-Wesley: Reading, MA.

Parker, D. (1982). Learning logic, Stanford University, Dept. of Electrical Engineering, Invention Report 581-64, October.

Robbins, H. & Monro, S. (1951). A stochastic approximation method, Annals of Mathematics Statistics, Vol. 22, pp. 400-407.

Robinson, A., Niranjan, M. & Fallside, F. (1988). Generalizing the nodes of the error propagation network, Cambridge University Engineering Department, Technical Report CUED/F-INENG/TR.25.

Rosenblatt, F. (1962). Principles of Neurodynamics, Spartan Books: Washington, DC.

Rozonoer, L. (1969). Random logic networks I, Vol. 5, pp. 137-147; Random logic networks II, Vol. 6, pp. 99-109; Random logic networks III, Vol. 7, pp. 129-136, Automatic Remote Control.

Ruck, D., Rogers, S. Kabrisky, M., Maybeck, P. & Oxley, M. (1990). Comparative analysis of backpropagation and the extended Kalman filter for training multilayer perceptrons, IEEE Transactions on Pattern Analysis and Machine Intelligence, in review.

Rumelhart, D., Hinton, G. & Williams, R. (1986). Learning representations by back-propagating errors, Nature, Vol. 323, pp. 533-536.

Sanger, T. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network, Neural Networks, Vol. 2, pp. 459-473.

Sejnowski, T. (1977). Storing covariance with nonlinearly interacting neurons, Journal of Mathematical Biology, Vol. 4, pp. 303-321.

Simpson, P. (1990a). Artificial Neural Systems: Foundations, Paradigms, Applications and Implementations, Pergamon Press: Elmsford, NY.

Simpson, P. (1990b). The fuzzy min-max neural networks for clustering, in review.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Simpson, P. (1990c). Higher-ordered and intraconnected bidirectional associative memories, IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-20, pp. 637-653.

Singhal, S. & Wu, L. (1989). Training multilayer perceptrons with the extended Kalman algorithm, In D. Touretzky (Ed.), Advances in Neural Information Processing Systems 1, (pp. 133-140). Morgan Kaufmann: Palo Alto, CA.

Specht, D. (1990). Probabilistic neural networks, Neural Networks, Vol. 3, pp. 109-118.

Spiegel, M. (1975). Schaum's Outline of Theory and Problems of Probability and Statistics, McGraw-Hill Book Company: New York.

Steinbuch, K. & Piske, U. (1963). Learning matrices and their applications, IEEE Transactions on Electronic Computers, Vol. EC- 12, pp. 846-862.

Sutton, R. & Barto, A. (1981). Toward a modern theory of adaptive networks: Expectation and prediction, Psychological Review, Vol. 88, pp. 135-171.

Szu, H. (1986). Fast simulated annealing, In J. Denker (Ed.), AIP Conference Proceedings 151: Neural Networks for Computing, (pp. 420-425). American Institute of Physics: New York.

Tank, D. & Hopfield, J. (1986). Simple 'neural' optimization networks: A/D convertor, signal decision circuit, and a linear programming circuit, IEEE Transactions on Circuits and Systems, Vol. CAS-33, 533-541.

Tesauro, G. (1986). Simple neural models of classical conditioning, Biological Cybernetics, Vol. 55, pp. 187-200.

Werbos, P. (1974). Beyond Regression, PhD Dissertation, Harvard University, Cambridge, MA.

White, H. (1989). Learning in neural networks: A statistical perspective, Neural Computation, Vol. 1, pp. 425-464.

White, H. (1990). Neural network learning and statistics, AI Expert, Fall 1989.

Widrow, B. & Hoff, M. (1960). Adaptive switching circuits, 1960 WESCON Convention Record: Part IV, pp. 96-104.

Widrow, B. & Stearns, S. (1985). Adaptive Signal Processing, Prentice-Hall: Englewood Cliffs.

Widrow, B. & Winter, R. (1988). Neural nets for adaptive filtering and adaptive pattern recognition, IEEE Computer Magazine, March, pp. 25-39.

Willshaw, D. (1980). Holography, associative memory, and inductive generalization, In Anderson, J. & Hinton, G. (Eds.) Parallel Models of Associative Memory (pp. 103-122). Lawrence Erlbaum Associates: Hillsdale, NJ.

Young, T. & Fu, K., Eds. (1986). Handbook of Pattern Recognition and Image Processing, Academic Press: San Diego, CA.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

A Neural Network Design Methodology: Considerations and Issues for Design and Project Management

BY

B. Archie Bowen, Ph. D., P. Eng.,
President, CompEngServ Ltd., 19 Fairmont Avenue, Ottawa, Ontario

ABSTRACT

An artificial neural network (ANN) is a software implementation of a neural paradigm, and, therefore, such projects yield to many of the disciplines of software engineering. On the other hand, many issues that must be faced, as the project proceeds, are unique and require specialized knowledge to address.

This paper is concerned mainly with the management of such projects, however in order to propose the management issues, it seems necessary to understand, at least superficially, the process of the design and implementation of a neural-based system. This paper therefore begins with a proposal for a methodology for the conduct of a project involving the choice, design, and implementation of a neural-based system. It outlines the issues that should be considered and resolved at each step of the project.

Based on this methodology, a project management plan can be put in place. Such a plan calls for a set of milestones and design reviews for various levels of management (and the customer) and a corresponding document set designed to prove a milestone has been reached, and, finally, that the original requirements have been met.

1.0 INTRODUCTION

This paper brings together past experience in the development of software systems, including expert systems and neural nets, in an attempt to formulate a system design methodology for neural net projects. This is an important requirement for both the customer and the developer if such projects are to become a professional activity and commercially feasible.

As with the early days of expert system projects there seems to be a host of issues unique to neural computing which would suggest that the rules of good project design and management can be ignored. It is the thesis here that these rules cannot be ignored and that there is little excuse for 'hacking' towards a solution. There are, in fact, critical issues to be resolved and there are appropriate times to face these issues, and there is also a minimum level of knowledge and experience necessary to resolve them and proceed. It is towards the structuring of these issues and the evolution of a design methodology for facing the issues when required, and for providing a mechanism for providing evidence that they have been faced, that this work is dedicated. When all of these things are understood, it is then possible to develop a methodology for such projects, and from this a project management approach.

The present work has been strongly influenced by a general systems design methodology established by the author [A-1], by extensive experience in doing battle with real neural network applications, and by later work, specific to neural computing, by Robert Hecht-Nielson [A-2] and by Bailey & Thompson [A-3]. It is also influenced by the procedures and reporting mechanisms defined in the United States Department of Defense Military Standards (MIL STDS) 2167A and 490.

In Section 2 an overview of the methodology is given, followed by four sections devoted to an examination of the procedures that should be executed and issues that should be faced at each step. In Section 7, a plan for project management is proposed. This plan shares many features with any plan to manage the production of software systems. The details are based on the structure and the resulting milestones of the proposed methodology.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

2.0 THE METHODOLOGY - AN OVERVIEW

The creation of an artificial neural network is essentially a software project with a special set of rules and issues that must be observed and addressed by the design team. It is important to distinguish here between a project in which no specific goals or deliverables are expected, such as a familiarization exercise, and a project with a defined level of effort, deliverables and a budget. We are specifically concerned with the later.

Neural engineering is not as advanced as other aspects of software engineering, however, it would be folly to believe that creating and managing a project in neural engineering is different from any other software undertaking. A methodology is therefore required, and proper project management and control essential to the successful completion of anything but a toy project. The methodology is strongly influenced by the concepts developed in the DND standards for software systems development. That system produces three specifications: called the Level A, B, and C Specification. The Level A specification (the A Spec.) describes the end-user problem to be solved and provides the functionality and performance and that must be achieved and the constraints which must be satisfied. The Level B Specification (the B spec.) is prepared by the design team and describes the top-level design of the system to be built. This specification is usually reviewed at a Preliminary Design Review (the PDR). The Level C specification is a detailed design document which described the system to be built. This document is reviewed at a Critical Design Review (CDR).

In addition to design reviews attended by the customer, there are lower level design reviews conducted by the design team usually conducted on a regular basis. These internal reviews keep the project on track and are invaluable preparation for the more public reviews. A good methodology contains an intrinsic modularity at which the state of a project can be assessed, reviewed and corrective action taken to ensure and maintain convergence to the original requirements, if necessary.

The methodology proposed here has four major phases:

Requirements Analysis
Logical Design
Implementation
Integration and Maintenance

Each of these phases mark a major milestone at which the project can be evaluated and decisions made as to progress and continuation.

The **requirements analysis** is sometimes referred to as 'functional specification development' and results in the definition of the equivalent of the Level A Specification. This phase provides the interface from the original problem to the functional specifications. During this phase, the desired functionality and performance of the final system is specified. In addition, the user and system interfaces of the final system should be outlined. As part of this analysis, it is important to define the constraints (functional, economic and otherwise) that the design team must consider during the design phase. In general, the approach is to consider the system to be built and specify how is should appear to the user and, if appropriate, how it will interface to other portions of a total system. An important aspect of this phase is to determine the available data sets and how the final system will be evaluated for acceptance.

The **logical design** phase involves selecting the appropriate set of neural paradigms, designing the network and finally the training regime. Each of these sub-phases constitute an ideal point for an intermediate design review and project milestone meeting. Part way through this phase, a Preliminary Design review (PDR) should produce the equivalent of the Level B Specification. This is normally reviewed by the customer in terms of the original requirements laid out in the A specification. The conclusion of this phase should provide the implementation team with the equivalent of the C Specification - a clear set of specifications for implementation.

The **implementation phase** is when the neural system is created, trained and tested. An important part of this phase is the choice of the implementation platform, the detailed training and the testing (and often the debugging) of the network. This phase demands the most 'on the bench'

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

experience, since the gulf between theory and practice is, in some aspects of neural systems engineering, very wide indeed. The result of this phase is the product, ready for integration and delivery.

The final system must be delivered **integrated and maintained** over its life time. This phase reinforces the need for an agreed upon acceptance plan and a document set that will permit maintenance. Experience will confirm that these details should be considered at the beginning, rather than at the end of the project.

The management of any project is intrinsically bound to the methodology being followed by the design team. A good methodology has many attributes which simplify the overhead associated with its management. Of great importance is its modularity, which yields milestones at which progress can be measured and control exerted. Other factors are peculiar to the particular software paradigm, and these will be compared and explored in the final section.

3.0 REQUIREMENTS ANALYSIS

3.1 Introduction

Requirements analysis forms the first step in any project, however, it is often overlooked in projects designed to exploit new technologies, and often a statement of need is mistaken for a definition of requirements.

The detailed mechanisms and the documentation of the requirements will depend on the formality of the project organization, however, even for in-house projects, time spent on requirements will benefit the project by finding a common ground for the project team.

It is suggested here that the following form the minimum considerations that should precede a neural net project: bound the problem, bound the project, define the acceptance tests, and finally define the total deliverables package. These ideas are not profoundly different from any other project, and will bring a focus to the project which will prove to be invaluable. In a global sense the issues are 'What are we trying to build?', 'Under what set of constraints are we trying to build it?' and 'What demonstration will we require to prove it has been built?'

3.2 Bound The Problem - What are we Trying to Build?

A necessary preamble to the final definition of requirements is to obtain from the user(s) a clear idea of what it is that will satisfy their needs. This is often difficult because the operational language of the users may not be at a sufficiently technical level to be easily translated into comprehensible technical jargon. Never-the-less, unless this step is clarified in some detail, almost always what is produced will not be what was expected. This situation does not apply only to neural networks, as your experience will confirm.

3.2.1 Statement of Required Functionality

The need here is for a concise statement of what functions the end-product will execute. This is often stated in the user's vocabulary, and must be eventually translated into technical jargon by the requirements analysis team. The user should be encouraged to define 'what' is required in as few words as possible. This discipline tends to focus the need and removes concepts of 'how' it should be done from the discussion.

3.2.2 Solution Requirements

In this section a more detailed statement of the solution is given. This should include the type of solution, the accuracy acceptable to the output, and the time constraints, if any, that are necessary.

3.2.3 Data Sources

The data definitions should include the data available for training and testing the system as well as the data input to the final system if the format is different.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

3.2.4. Define the Interfaces

System interfaces include all hardware and software interfaces, including the requirements of operation with specific operating systems and existing software.

Often overlooked is the need to specify high and low level protocols for control synchronization and the format and structure of data passed into and out of the surrounding system.

This machine-human interface is often the most crucial in the user's final acceptance of the system. It is also the most difficult to specify in its entirety. In the end, every screen interface and control protocols for mode changing, screen manipulation and data passing must be specified.

3.3 Bound the Project

The project is bounded by specifying the total budget which fixes the level of effort. In addition, however, time is an often overlooked constraint. There are two aspects to timing constraints: project time and performance time. If a solution has to be available in a certain time frame, this imposes constraints which should be understood at the beginning. If the neural network must fit into a larger system response time may form a constraint. This will drive a host of considerations from the neural topology to the execution platform.

3.4 Define Acceptance Tests

Neural networks are trained to respond to a set of data elements which are alleged to define the input space. Because of the data dependence of the success of a project, it is of critical importance that the final set of tests that are formulated to determine success or failure, and hence acceptance of the final product, be specified in detail. From the contractor's point of view a test set which is not representative of the training set can spell disaster.

All projects start with the accumulation of a data set which must be representative of the problem and must eventually be used for training and testing. The problem is to guarantee that the training and the test set can be considered representative.

3.5 Define the Deliverables

3.5.1 Documentation

In order to maintain the neural net a complete set of documentation is required. A description of a neural net consists of a description of the paradigm, and the implementation topology. In addition the training and test set should be documented. It is most useful when retraining the system to have a knowledge of the training parameters and the details of the training regime. Finally any interface software, and restrictions on the execution platform.

3.5.2 Code

The most notable difference between the documentation of a neural paradigm and classical software is the 'black box' character of neural nets. The concept of documented code is not applicable since the character of the neural net response is buried in the topology and the weights of the neurons. Furthermore, as discussed elsewhere, maintenance and extensions of the neural net is different in concept than classical software. The neural net is often thought of as a black box.

The documentation must be sufficient to permit the reconstruction of the neural net topology and the weights of each neuron. This can take the form of a description of the paradigm and the topology and a printed list of the weights (despite its length in some cases). With this data the network can be reconstructed, and retraining because of minor input sets undergoing change can often be shortened by beginning with the trained set.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

4.O LOGICAL DESIGN

4.1 Introduction

The logical design phase begins the process of translating the requirements into a proposal for implementation. In this phase all the capabilities of neural computing paradigms should be examined to determine the best approaches to satisfying the requirements.

This phase is often a preliminary step in a bid/no-bid situation. Inappropriate requirements formulated by a potential customer can lead to a no-win situation if a neural paradigm is demanded, accepted for design and delivery, and is intrinsically inappropriate.

There are no guarantees in this field, however, a few preliminary considerations will enhance the probability of the right choice.

4.2 Confirm the Application

The logical design team should begin the design process by reconfirming that a neural computing paradigm is suitable for the problem. In general if an expert system solution will satisfy the requirements then it should be chosen before a neural solution, and by extension if a classical software algorithm will fulfill the needs, it should be chosen. The design team should look at not only the functionality of neural computing but at the availability of data and the impact of the other requirements.

4.2.1 Characteristics of Successful Applications

Successful neural applications have the following characteristics:

1. The algorithm to solve the problem is unknown or expensive to discover.
2. Heuristics or rules to solve the problem are unknown or perhaps difficult to enunciate.
3. The application is data intensive and a variety of data sets are available which can be identified as correct or describes specific examples.

Several classes of problem have these characteristics at this time: Pattern recognition, pattern completion or pattern classification, Statistical mapping.

Of these classes, applications include: Character Recognition, Image classification, Forecasting, Incident Detection, Signature Identification, robot control, signal processing.

In general, it should be determined that:

1. Conventional computer technology is unsuitable or inadequate.
2. The application requires qualitative or complex quantitative reasoning.
3. The solution is derived from inter-dependent or correlated factors which are difficult or impossible to quantify.
4. Data is available and corresponding known solutions can be derived.

4.2.2 Characteristics of Poor Applications

Poor applications include:

In general, those -

1. For which algorithms or rule-based solutions are possible.
2. That require deduction and a logical approach are not suitable.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

3. That require explanations of procedures.
4. That are essentially mathematical computations or transformations.

In particular, those -

1. Requiring precise mathematical computations.
2. In which answers must be explained or the steps documented.
3. An adequate an representative data set is not available for training and testing.

4.2.3 Choosing a Software Paradigm

If the final system is to operate embedded in the original development system, the issue of a software paradigm is relatively unimportant. In situations in which the system must interface to a variety of data bases, graphics displays, and surrounding software, the representation of the whole system and indeed the underlying language may become an important issue to resolve. Obviously the issue is the induced overhead in creating the software interfaces to link the system, and the potential consequences on performance.

4.3 Select the Neural Paradigm

This step involves selecting a potential set of neural paradigms which match the application requirements. The issues here are the size, training and time constraints, output type. Table 1 contains a comparison of the capabilities of a variety of neural paradigms, which could be updated as newer technologies become proven. The designer should choose a potential set of paradigms which match the requirements, and prioritize the most likely candidates. In a constrained environment (time and money) the highest priority candidate is is started first. However, the other candidates may have to be called upon if unforeseen events prevent training convergence or performance is not as expected.

4.3.1 Network Paradigm

The network choices include, the number of layers or slabs, the number and type of nodes, the size of the hidden layers, the number and type of output nodes, and the connectivity of each neuron and layer.

4.3.2 Output Type

Choosing the Size of the Output Layer: Choosing the number of output neurons depends on the paradigm being used and on the type of output being generated. There are two broad categories of outputs: hetro- and auto-associative. Auto-associative networks have the same number of outputs as inputs, whereas hetro-associative generally implies less. These categories are far too broad, and a further division into the various expected outputs is useful. These depend on the application and can be categorized as: Classification, Images or Patterns, Optimizations, and Numbers.

Classification: The outputs are interpreted as categories or attributes. The output is either a binary vector or a real number. Generally classification is indicated by a binary vector of all zeros expect the class of the input data which is a one. In some cases, real numbers are used to indicate further information as, for example, the confidence of the classification.

Images or Patterns: In this application the outputs are interpreted as an image or pattern generated in response to the input. The number obviously depends on the detail of the expected patterns.

Optimization: The output size depends on the optimization problem and the information required to interpret the results of the class of input data being optimized.

Numbers: Numbers are a subset of the other categories, however, in general numbers are used when the output represents a number, such as power levels or switch settings, etc.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

4.3.3 Training Method

Neural training falls into three categories: supervised, unsupervised and reinforced. The choice depends on many factors but is strongly influenced by the availability of data. Supervised learning requires pairs of data vectors consisting of the input pattern and the correct output. The training data must therefore contain the solution the network is expected to provide. Generally this training mode demands extensive data sets, and can consume a long time to achieve the correct responses.

Unsupervised learning classifies input data patterns according to some form of nearness criteria. The classification will depend on the structure of the training data, and the training time is usually much shorter than unsupervised learning.

Reinforced learning is a compromise between the two. It requires only the input data and and indications of the goodness of the response (a reward signal). Reinforced learning can consume much longer times than the other two, but training data requirements are less stringent, although the goodness criteria must be attached to each response.

4.3.4 Time Constraints

All aspects of the training and operation of neural networks are computationally intensive, since every neuron performs a sum-of-products calculation often utilizing floating point operations. Training time is usually not counted as part of the operational timing constraints, however, from a project point of view, training times can be very large on an inadequate platform. This time is pure delay, which tends to limit the iterations that can be tried in a fixed time-frame, and can of course finally influence the delivery time table.

If the network must fit into a hybrid software system then the operational response should be specified. If it is part of a diagnostic or prediction system, the response may be not critical. In any event, response should be considered as a constraint which can influence the size of the network and eventually influences the cost of the execution platform.

4.4 Network Design

The design of the network involves three basic issues: the node, the network topology, and the training details.

4.4.1 Node Level

The node or neuron design is constrained by the type of input to be used, the transfer function and the nearness computation. The input data format has already been specified and is usually unalterable. The nearness function is usually an inner product (a sum of products) however others can be used such as a vector difference. The transfer function is the nonlinearity following the nearness computation. This can be linear, signum, sigmoid, and hyperbolic tangent. The selection is determined by the characteristics of the region boundaries and in the case of backpropagation training by the necessity of a differentiable function. The calculation of the nonlinearity affects the computational complexity of each neuron and the simplest possible should be chosen.

4.4.2 Network Level

At the network level of design, the topology of the interconnection of the neurons must be decided. This involves the number of layers or slabs within a layer, the number and type of nodes, the size of the hidden layers, the number and type of output nodes, and finally the detailed interconnectivity of all the neurons. Several paradigms have a fixed topology in that the number of layers is predefined, e.g., Hopfield nets, Kohonen self-organization maps, etc.

In backpropagation nets, hidden layers act as levels of abstraction. Adding hidden layers will increase the ability to abstract characteristics of the input classes, however, training will take longer and in the end the training of multilayer networks by backpropagation become very tedious and convergence is not necessarily guaranteed in practice. The number of neurons in a hidden layer affects the ability to generalize the characteristics of the input data. Generalization and memorization becomes a critical issues in selection the number of neurons in the hidden layer.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The two opposing schemes for back propagation topology achieve memorization of the input training set or achieve generalization of the features of the training set to identify examples never before seen. In general, increasing the number of neurons in the hidden layer offers sufficient memory for the network to memorize the test set. Conversely reducing the number of neurons, up to a point, forces generalization.

Determining the exact number of neurons to achieve generalization is not a solved problem, and is often achieved by experimentation. The difficulty lies in the affect on the test set. A network trained to memorize will achieve very good results if the test set is equivalent to the training set: and conversely the performance can be very poor if the test set includes new examples outside the training set.

4.4.3 Training Issues

The issues to be addressed before training begins are both strategic and tactical. Strategically the training falls into three phases (as in chess) with a beginning, a middle and an end game. In each of these phases, training parameters can be varied to hasten or encourage convergence. The plan should outline the training parameters for each phases and some measurements which will suggest when each phase has been completed. On the other hand, measurements should be determined to decide when training is not converging, and the time has come to back-up and try a new set of parameters. In practice it is often difficult to predetermine these measurement exactly and sometimes a certain amount of synergy is necessary to observe lack of progress and to suggest corrective actions. The need for a theoretical background, experience, and good judgement in combining the theory and experience become evident during this phase.

These issues will be discussed in more detail in Section 5.4.

5.0 IMPLEMENTATION

5.1 Introduction

The implementation phase is the crucial phase in the development of a neural project. Despite all the preparation, it is not always possible to guarantee convergence of the training, however, following a well established methodology [B-1] will enhance the probability.

The key activities are: Characterize and Prepare the Input Data Set, Choose the Development System, Train the Network, and be prepared to Debug and Test the Network.

5.2 Characterize the Input Data Set

5.2.1 Assemble and Prepare the Input Data Set

This phase consists of two major activities: assembling the data set and preparing it for training and eventual testing of the network.

The input data set refers to all the data that will be used both for training and testing the network. Initially the concern is with the quality of the data. Under some circumstance the data can be ambiguous, error ridden, come from multiple sources and formats, and in some cases have conflicting judgements on its classification.

Preparing the data refers to two major activities: accommodating the input formats of the development environment, and preprocessing the data to enhance its training potential. Accommodating the input formats suggests the potential need for code conversions, and normalization scaling. Preprocessing, such as creating ratios or some form of filtering, is sometimes useful in enhancing training or the meaningfulness of the results. Obviously all training and test data sets must be brought to the same format before being used.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Of critical importance during this phase is the definition of the acceptance test set. This is the final data input which will define if the system is preforming with sufficient accuracy to be useful to the end-user (and it may determine if the final invoice is accepted).

Typically the fnal test set is not made available to the development team. Since many neural nets can be made to memorize a given set of input data, it is clear that the acceptance test set should be a set which at least includes samples that have not been used in training.

On the other hand the development team needs a test set to be used to evaluate the effectiveness of the training regime. A large set of data covering all interesting cases should be made available to the development team from which they can choose the optimal training and test sets. The goal is to force the neural net to generalize the characteristics of the input data classes based on the test set so that appropriate responses to the test sets will be derived.

5.2.2 Select the Training Set

The selection of the training set for neural paradigms is the most critical decision that affects the final outcome. While it is easy to say, the set must represent the total range of inputs in a relative density of occurrences to represent the final desired results. This is not easy to accomplish, since the n-dimension volume of the total space is impossible to define exactly, choice of examples for training (and testing) is difficult.

A training set should be assembled as a subset of the total data set. In a real sense all the data assembled is a potential candidate as a training set. Including all this data, however, will profoundly affect the training time, and the cost of the project.

The training set can be considerably smaller than the total data set and should be chosen to achieve generalization across the various classes of the problem. The training set should represent the key features of the problem. A representative set should cover the breadth of the problem to be solved. For example in a pattern recognition problem, the set should cover the range of problems in the classes of images. In a decision or control problem, it should cover all the significant cases.

In some cases it is possible to partition the training set into routine, difficult and border line cases. This partition will be most useful in determining convergence conditions and, in particular, lack of training convergence, if this occurs.

5.2.3 Select a Test Set

The test set should provide evidence that the system will be useful to the customer. The customer and the contractor share a responsibility to ensure that this set reflects the customer's perception of an adequate test, and the contractor's technical understanding of the relationship between the training and the test conditions. The test set should reflect the distribution of input vectors similar to the training set. A test set with parameters outside the training set can lead to failed tests.

5.3 Choose the Development System

Experience suggests that the choice of the development platform will have the most profound affects on the success of the project. A wide variety of software development systems have appeared over the last few years, some of which are useful as experimental learning tools for a University Laboratory, and others which provide a leaner environment for the skilled professional. The development system includes the software simulator, the operating system and the hardware platform.

Many simulation platforms are now available to facilitate the process of training, testing, debugging, and creating and displaying the system and human interfaces. These platforms operate on a variety of workstations and PCs. The characteristics of the platform will profoundly affect the level of effort needed to set up, train, debug and test the system. Depending on the financial resources committed to the project, a system should be judged based on:

The Vendor
- Support and Training
- Fielded Systems

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

- Price
- Functionality
  - The Neural Paradigms Implemented
  - User-Programmable Paradigms
  - Training and Debugging Facilities
- Interfaces
  - Graphics and Displays
  - System software Interfaces
  - Graphics Interfaces
  - Database Interface
  - Language Interfaces
- Support Platforms Needed or Required
  - Operating System
  - Hardware Platform
  - Multiple Screen Processing (Windows)

The relative importance of these factors will depend on the project and the team's experience. The final system can either operate in a stand-alone mode or be part of a larger system. In either case, the development environment may have to be suitably modified in order to integrate the operational network into the final system, thus portability may also be an important consideration. Finally in the choice of a new system, the whole system capability should be carefully traded-off against not only the learning curve required to begin work, but the learning curve to become really proficient.

5.4 Training the Network

5.4.1 Training Phases

Many training paradigms have a sequence of phases. In each phase, the training parameters can be optimally adjusted to speed the process.

5.4.2 Selecting Training Parameters

Once the paradigm, the structure of the neuron and the network topology have been decided, a choice of training parameters is required. In backpropagation training, for example, the initial weights, the learning rate, and the momentum must be selected before training begins. The implementation team should have considered the choice of these parameters and if appropriate considered the variation of parameters as training proceeds and convergence begins to occur (or otherwise).

5.4.3 Convergence and Nonconvergence

Despite the theoretical proofs of convergence, experience suggests that neural training often results in a hung situation in which the network will not converge. This can be caused by many factors: for example, a poor choice of the training set, inappropriate training parameters, a stabilization occurring in a local minima, by overtraining some of the neurons, or by network paralysis. Aside from experience, which might suggest corrective approaches, it is in this situation that a powerful simulation platform to assist in the debugging will be most appreciated.

5.5 Debug and Test

In the broadest sense, the test set is chosen to achieve some level of acceptable response. As discussed in Section 5.1.1, the test set should be partitioned into routine, difficult and boundary cases.

This approach is termed 'black box' testing and is the most common. There are however, other testing procedures which are important in some cases and can be important indicators of bad training and/or redundant layers or nodes. Some simulation packages for example permit the viewing of the weights, and the on-line computation and presentation of errors, etc.

Each of these approaches will be discused in the following. Finally, there is the question of what to do if the network fails the acceptance tests. This will be discussed in the final section.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

5.5.1 The Black Box Approach

Generally a neural net is tested by comparing input and output for the appropriate response. The network remains essentially a black box with its internal coding and data transformations being undecipherable. Under these conditions the test set should, if possible, be divided into easy, difficult and boundary sets. Acceptance criteria should be developed for each set. Attention should be paid to comparisons to human responses when this is possible. Finally, in boundary cases, it is useful to predefine the threshold acceptance level of the output responses.

5.5.2 Node and Layer Redundancy

Eliminating nodes and indeed whole layers can substantially reduce the computational complexity of the final system. In some cases removing these redundancies will increase the convergence of the training process.

By examining the weights of each node, those with low values of weights make a negligible contribution to the final output and can likely be eliminated. Such pruning should be followed by continued training to determine if an improved accuracy can be achieved, or to insure that the incremental contribution that has been removed is restored.

A rule of thumb suggests that weights below about 0.1 are probably redundant and can be removed.

At the other extreme, nodes that have weights much in excess of others should be suspect, for it may indicate over training and contribute to a lack of generalization. Such a situation may suggest a repeat of the training process, or indicate that some of the test set will fail.

5.5.3 Input Node Activation Sensitivity

In some applications it is possible to determine inappropriate behaviour by carefully selecting a test set to positively reinforce an expected output at a given node.

5.5.4 Responses to Failed Test-Procedures

If after successful training, a network fails to respond to the test set with acceptable results, there is a whole sequence of considerations that must be considered in a rational order by the design team. In general these are:

The training and test set
The learning algorithm
The network design
The system interfaces

Training and Test Sets: The first thoughts are about the training and test set. The test and training set should be re-examined for quality, representativeness and accuracy. The training set must be chosen with the same characteristics as the training set. A test set with input members different from the training set will invariably lead to testing failures.

The Learning Algorithm: The learning algorithm constants should be examined.

The Network Design: The network nodes characteristics, architecture and connectivity.

The System Interfaces: All interfaces should be examined including those between the training set and the network, the user and the network and any other interconnected software.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

6.0 DELIVERY AND MAINTENANCE

6.1 Introduction

This phase of the project resembles any software delivery phase. The principle difference is the mechanisms for demonstrating performance and functionality.

6.2 The Acceptance Test Plan

The functionality and performance tests should have been formulated as part of the requirements analysis. The plan to satisfy these requirements should have been formulated and accepted at the preliminary design review.

6.3 System Integration

Neural networks tend to be regarded as stand-alone systems, however, they can be integrated into an overall system in two configurations: loosely coupled, or tightly coupled.

Loosely coupled neural networks (probably the most common at this time) communicate by passing data files. Such systems function either as preprocessors, post processors, or as a distributed system. Preprocessing networks prepare data for examination or processing by other software modules. Post processing networks are often used to remove noise, classify patterns or make predictions. A distributed system passes data to a neural net for analyses or to interface to another system.

Tightly coupled systems are more full integrated, relying on data sharing to pass data. In a fully integrate the neural net tends to loose its identity and become another module in a larger system.

6.4 System Performance Evaluation

The evaluation of performance usually includes both functionality and computational complexity.

Performance is judged by the successful treatment of the testing data set. Computational complexity includes both memory requirement of the program and the time needed to fulfill its function.

6.5 Maintenance Plan

The maintenance plan should consider three major issues: Environmental Modifications, Structural Modifications and Interface Modifications.

Environmental modifications suggests that the character of the the input data has changed. This could occur from a wide variety of causes, however, the result is the need for a redefinition of a training set and the complete retraining of the network.

Structural modifications suggests that the role of the neural net is found to be unsatisfactory or it must be changed to accommodate newer roles in the system. A structural modification suggests in the worst case a complete reconsideration of the design methodology or at best a reorganizing and training of the existing network.

Interface modifications suggests software changes to the human or system interfaces. These could occur from the need for more useful data presentation to humans, changes to the format of the data base, or protocol changes to the surrounding system. Depending on the original requirements, plans should be produced to show how these potential exigencies will be approached.

We note that a maintenance plan, as in other software projects, depends on a document set that will support the actions needed over the life of the system.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

7.0 PROJECT MANAGEMENT

7.1 Introduction

In this section the basic project development methodology will be pulled together to exhibit a management approach.

7.2 Management Overview

A characteristic of any design methodology is the intrinsic mechanisms for project monitoring and control. Project plans and methodologies all exhibit some structure and modularity at which states can be measured and either forward or backward influence exerted to ensure convergence on the original objectives or to adapt to changes caused by influences outside the project, or by unforeseen events occurring in the project. These points in the methodology are characterized by milestones which occur at the conclusion of a predictable set of activities at which reports can be prepared and progress measured. Milestones also are points at which control can be exerted to account for slippages caused by inappropriate predictions of the level of effort, the level of difficulty, or by changes in the requirements.

In general there is a distinction preserved between major and minor milestones. There are many ways of defining such a distinction depending on the environment and the project. For our purposes a major milestone signifies a point in the design process were a significant goal has been achieved; typically the completion of a set of tasks marking a logically complete step in the overall process. A useful criteria is to assume that at a major milestone, a different team will take over the project and must be provided with a set of specifications for their task. Thus, major milestones are points in the project where significant documentation and evaluation occurs. Minor milestones are important events which are typically part of a larger logical task.

Staffing a neural network project depends on many factors. Project Leaders should have experience in software development, and preferably a working knowledge of the capabilities and limitations of neural computing paradigms. Since a wide variety of skills are necessary, it is not necessary for all team members to be experts in neural computing, however, one team member should have the capability to do the system analysis, and to determine the appropriate neural net paradigm, as well as to judge the training and test sets. Programmers with conventional skills may be required depending on the human and system interfacing requirements. Finally since vast amounts of data are usually required, experience with data structures and data manipulative software is very useful.

The methodology as outlined contains such milestones, and the issues that should be addressed and activities that should be executed, and hence the reports that can be prepared by the project team to provide evidence of reaching the milestone (or otherwise). In this section we will outline these milestones and suggest the form of reporting and actions that are appropriate for managers.

7.3 Project Milestone Reviews

7.3.1 Major Milestones

The proposed methodology is characterized by four major milestones: Requirements Analysis, Logical Design, Implementation, and Integration and Maintenance. These mark major milestones in the project life-cycle. Each is characterized by attaining certain goals and each can be validated by determining progress and achievements as discussed in the previous sections. Essentially at each major milestone, the project team can be required to prepare reports outlining how each issue has been addressed and the reasons for the choice of each alternative. The project leader is typically charged with the responsibility of sign-off for the document set attesting to the conclusion of the milestone.

Major Milestone #1: Requirements Analysis

In typical software projects, the requirements analysis is completed by the customer's team based on the perceived needs of the end-user and results in a set of requirements specifications (the A-Level Specifications). The design team has the problem of analyzing these specifications and

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

reducing these to a document set suitable for their purposes. In either case, evidence should be presented that the following questions have been answered or the issues have been addressed:

What is:

The required functionality?
The performance?
The human and system interface definitions?
The acceptance tests?
The operational constraints?
The non-functional constraints on the final system?
The maintenance and documentation requirements?

Finally "Is there an adequate supply of useable data for training and testing?"

Major Milestone # 2: Logical Design

The conduct of this phase has been outlined in Section 4. The conclusion of this phase is marked by the presentation of evidence to demonstrate either the consideration and/or attainment of the following:

1. The application has been confirmed, in the sense that the team is confident that a neural computing approach will yield results.

2. The neural paradigm has been selected including estimates or starting data for: the network paradigm(s) including the network size, the output type, and the training method, Time Constraints

3. The network alternatives have been designed, including the node level, and the network level. The training issues and parameters have been considered and allocated.

Major Milestone # 3: Implementation

The level of effort and the amount of time required to implement the neural net is very difficult to predict, due to the uncertainty of the training procedures. It is, however, during this phase that detailed management is required to ensure that the project is converging and that appropriate steps are being taken to maintain estimates of time and effort.

The expected result is, of course, that a trained neural net is presented as evidence of success, however, the team should at minor milestones present evidence that they have:

1. Characterized the input data set by assembling and preparing the input data set, selected the training and test set.
2. Chosen the development system, including the operating system and the hardware platform.
3. Achieved training to the standards required and successfully passed the preliminary test requirements.
4. Created all the user and system interfaces, and have completed any test and demonstration required by the original specifications.

The final requirement in this phase is the preparation of all the deliverable documentation, and the preparation of the demonstrations required by the factory acceptance tests.

If an integration of the system into a larger environment is required this should be completed and tested on-site, if possible. If the delivery requires integration into a system on the customer's site, the interface documents should be carefully pursued and the integration plan finalized.

Major Milestone #4: Delivery and Maintenance

This phase should mark sign-off of the project. The delivery of the required document set, and the test plan for acceptance should be prepared.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

7.3.2 Preliminary Design Review

A preliminary design review should occur after the finish of the logical design phase. At this point the complete logical plan for attacking the problem should be in place.

7.3.3 Critical Design Review

The critical design review should be held part way through Milestone 3, at the point where the major decisions have been made with respect to platforms and the detailed training plans are in place.

7.3.4 Pre-Delivery Preparation

Predelivery activity should include the instantiation of the user and system interfaces, as required, as well as the pretesting preparation for the final test. The documentation should be subject to quality assurance, and the list of deliverables checked-off.

7.4 Documentation and Configuration Control

It is truism to say that configuration control (or the lack of it) has contributed to more failures and cost overruns in software projects than any other single cause. In neural engineering the need for configuration control is even more urgent. In addition to the normal software documentation, a complete record of the training portion of the project becomes critical in judging progress, and maintaining convergence in time.

Of particular importance is a detailed record of each parameter, and the changes effected during training runs, the number of iterations, the rate of convergence, and any other tuning efforts based on the heuristics of the team.

This record will prove valuable not only in situations where convergence is slow to occur, but will be essential in post-delivery if modifications are to be effected in the field.

7.5 Disasters - Recovery and Containment

During phase three, a host of difficult problems can arise that require experience and often basic knowledge of neural paradigms to surmount. The response to these problems will be based on the quality of the implementation team.

In addition there may be fundamental problems which are caused by decisions made during earlier phases of the design methodology. Some of these may be very difficult to fix 'on-the-fly' and can contribute to project failure or, at best, over-runs of time and money. These pathological errors may occur at any stage of the project and can be classified as either global or detailed. Global errors occur during the logical design phase, and usually indicate a restart of the project.

Global errors include:

Wrong choice of neural paradigms
Wrong choice of test and training data
Wrong Simulation System
Inadequate Performance

Detailed errors refer to inappropriate choices made during the implementation phase, and are manifested as lack of convergence, lack of performance, or failure of the test set.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

8.0 SUMMARY AND CONCLUSIONS

8.1 Summary

The Management Structure:

The thesis underlying this presentation has been that a project involving the development of a neural computing system is subject to the same general rules of conduct as any other software project. The management structure must account for the special features of such project, since there is considerable difference in detail in arriving at a successful neural network compared to other more familiar software systems. The management structure and the managers must therefore be familiar with these difference and the issues which they raise if they are to understand the importance of addressing these issues and the alternatives which should be considered.

The Methodology:

The methodology proposed follows the classical four major activities of: Requirements Analysis, Logical Design, Implementation, and finally Delivery and Maintenance. Each of these mark a major milestone in a project and each has a set of activities, issues and specialized expertize required to successfully traverse the required activities. And most importantly, each can be reported by preparing a document set which addresses the relevant issues, and the solutions and/or alternatives found necessary to project a successful conclusion. This reduces the project and its management to a an understood set of activities which are close to those normally found in a software project. There are however some critical differences.

Project Differences: There are perhaps four major differences with more conventional software projects: First the success is dependent on the availability of existing data; Second, the logical design consists of choosing the set of paradigms most likely to yield success (rather than evolving a systems algorithm); Third, there is no guarantee that the training regimes will converge, and that the test sets will provide the evidence necessary for validation, and finally, the documentation and configuration management must be adapted to the neural paradigm.

Software Project Failures: Failures generally tend to be based very intangible details of the training and test sets, and the wrong choice of paradigm, and/or the lack of convergence of the training regime. A whole new set of skills and techniques are necessary to rescue a failing project.

Selecting the Paradigm: The field of neural computing is in a rapid state of expansion, however, there are now what could be called classical approaches. It should be possible to map an application onto a set of potential paradigms with a fair degree of confidence.

Selecting the Simulation Platform: The simulation platform is a critical choice, as in most software projects. Aside from the choice of paradigms, the most critical item is assistance in influencing the training regimes, monitoring the training progress and, if necessary, in debugging the failed system.

Training and Test Set Selection: The most critical concern of both the contractor and the customer should be the selection of the training and the test sets. The training set must reflect the full scope of potential inputs to the final system, and the test set must reflect the structure of the system as learned through the training set. Failure to select these sets will cause project failure, in most cases for the wrong reasons.

Training Failures: The most distressing feature of neural computing is the lack of convergence of the training procedure. As discussed there are many causes, and the field is rife with heuristics and procedures for rescuing the situation. The main recourse is the experience of the team, a good simulation platform, and sometimes raw luck.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

8.2 Conclusions

The Neural Computing Field:

At the present time, the movement of the theories developed by neural science to the practice of neural engineering is progressing along the same line that software did twenty years ago, and in the same manner that expert systems software did over the last decade. The field is in a tremendous growth phase with new theories, implementation platforms, and successful applications appearing daily. In addition there is a certain amount of the bandwagon syndrome appearing in the commercial world, resulting in many claims of neural computing expertise based on very limited experience with the realities of hard experience. In such an environment it is easy to be misinformed and misdirected. Many of the developed systems have been level-of-effort projects with an open budget; this situation must evolve towards a more commercial development project with the standard management, documentation and control procedures, if the field is to mature into a professional discipline.

Software or Hardware:

In the end, a neural network will be considered as one of the alternatives to solving a problem. Its inclusion in a hybrid system [C-1,2] composed of classical algorithmic software and rule-based software will depend on the nature of the problem and the capabilities of the different paradigms. This trend is already noticeable in hybrid systems of algorithms and rule-based systems. The implementation of neural paradigms will follow the path of special purpose software which has been relegated to microcode for such applications as input/output drivers, and will be implemented on special purpose coprocessor boards.

Contract Award:

If contracts are to be let, a review of the design methodology and the project management plan should be an integral part of the assessment procedure of the received bids. While there is still a level of uncertainty in the convergence of most neural network projects, there are good engineering design approaches which will minimize this risk and often contribute to the success of the whole project.

Project Management:

In a larger view of the development of a software solution, the need for a neural network would evolve, as part of the logical systems design, in response to the demands on the functionality, the input data, and other knowledge. The methodology proposed here has begun with the implicit assumption that a neural solution has been decided upon. It, never the less, proposed a distinct set of phases in which progress can be measured, and issues faced at the appropriate time. This certainly provides management control and leaves the development team with a set of guidelines for ensuring that all options are explored in a systematic manner. It also suggests when things may be going astray and convergence may not be occurring. The steps proposed may be traversed quickly with an experienced project team on familiar ground, however, an awareness on the part of the team and management of the logical sequence of considerations and issues lends order and structure to the project.

The Near Future:

The neural computing field is in a state of rapid change: in theories, in new architectures, and in computational platforms (software and hardware). Design and implementation teams will need a constant infusion of updated concepts and information to make full use of this technology. This statement is also true for those with applications that might benefit from the use of this technology.

Finally we have not considered the development of neural paradigm using new hardware neurons [D-1,2,3]. It is clear that large, high performance neural networks will be implemented using a variety of silicon or perhaps optical (perhaps, even biological) devices to simulate the neuron. Some of these will be trainable and some will accept weights derived from software simulation. This combination will offer an interesting challenge as the software and hardware engineers join their methodological requirements to achieve very large neural networks.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

9.0 REFERENCES

A. Design Methodologies

1. B. Archie Bowen & William R. Brown, "System Design," Prentice Hall, Englewood Cliffs New Jersey, 1985.

2. Robert Hecht-Nielsen, "Neural Computing," Addison-Wesley, Don Mills, Ontario, 1990.

3. David L. Bailey & Donna M. Thompson, "How to Develop Neural-Network Applications," AI Expert, June 1990, pp 38-47, & September 1990, pp 34-41.

B. Training Techniques

1. Maureen Caudill, " Neural Network Training Tips and Techniques," AI Expert, Vol. 6, No. 1, January 1991, pp 56-61.

2. G. David Garson, "Interpreting Neural Net Connection Weights," AI Expert, April 1991, Vol. 6, No. 4, pp 46-51.

C. Hybrid Systems: Neural Networks, Classical Software, and Expert Systems

1. David V. Hillman, "Integrating Neural Nets and Expert Systems," AI Expert, Vol. 5, No. 6, June 1990, pp 54-59.

2. Maureen Caudill, "Using Neural Networks: Hybrid Expert Networks," AI Expert, Vol. 5, No. 11, November 1990, pp 49-54.

3. Maureen Caudill, "Embedded Neural Networks," AI Expert, April 1991, Vol. 6, No. 4, pp 40-45.

D. Hardware Neural Networks

1. Tom J. Schwartz, "A Neural Chips Survey," AI Expert, Vol. 5, No. 12, December 1990, pp 28-32.

2. Maureen Caudill, "Embedded Neural Networks," AI Expert, Vol. 6, No. 4, April 1991, pp 40-45.

3. Jessica Keyes, "AI on a Chip," AI Expert, Vol. 6, No. 4, April 1991, pp 32-38.

E. Standards and Documentation

1. Denny Rock, Jerome Asarewicz, Rick Klobuchar, & Jennifer Oshin, "AI and the Military: Time For A Standard," AI Expert August 1990, Vol. 5, No. 8, pp 56-64.

DESIGN FACTOR / PARADIGM	TRANSFER FUNCTION	NUMBER OF LAYERS	TYPE OF INPUT ACCEPTED	SIZE OF HIDDEN LAYERS	CONNECTIVITY	LEARNING ALGORITHMS	LEARNING PARAMETERS
BACK PROPAGATION	SIGMOID HYPERBOLIC TANGENT	2 OR MORE	CONTINUOUS	SMALL TO MEDIUM	FULLY INTER-CONNECTED, RANDOM, SLABS	GENERALIZED DELTA RULE	LEARNING CONSTANT, MOMENTUM
COUNTER PROPAGATION	KOHONEN & SIGMOID	2 or 4	CONTINUOUS	SAME AS KOHONEN	FULLY INTER-CONNECTED	KOHONEN & GROSSBERG	KOHONEN & OUTSTAR
MADALINE	SIGNUM	2	BIPOLAR	SMALL TO MEDIUM	FULLY INTER-CONNECTED	DELTA RULE & MAX, MIN, MAJORITY	LEARNING CONSTANT
OUTSTAR	SIGMOID	1	BINARY CONTINUOUS	N/A	VARIES BY APPLICATION	OUTSTAR	DECAY TIME, ATTACK FUNC.
ART2	SIGMOID	1	GREY SCALE	INCREASES BY DATA CATEGORY	FULLY INTER-CONNECTED	ART2	VIGILENCE, GAIN
KOHONEN NETWORK	COMPETITIVE LEARNING	1	CONTINUOUS	EQUALS # OF DATA CATE-GORIES	FULLY INTER-CONNECTED	KOHONEN	NEIGHBORHCOD SIZE,ALPHA,BETA
BOLTZMANN MACHINE	VARIES	2 OR MORE	BINARY CONTINUOUS	SMALL TO MEDIUM	FULLY INTER-CONNECTED, RANDOM	BOLTZMANN	TEMPERATURE
HOPFIELD NETWORK	HARD LIMITING	1	BINARY BIPOLAR	N/A	FULLY INTER-CONNECTED	HOPFIELD	NONE
ADALINE	SIGNUM	1	BIPOLAR	N/A	FULLY INTER-CONNECTED, RANDOM	DELTA RULE	LEARNING CONSTANT
BAM	CLAMPED LINEAR	1	BIPOLAR	N/A	FULLY INTER-CONNECTED	BAM	RETENTION, GAIN
PERCEPTRON	PERCEPTRON	1	CONTINUOUS	N/A	RANDOM	PERCEPTRON CONVERGENCE, DELTA RULE	LEARNING CONSTANT

Table 1: Characteristics of Neural Paradigms

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

PARADIGM	ASSOCIATIVE MEMORY		TRAINING METHOD		TYPE OF OUTPUT				TRAIN-ING TIME	EXE-CUTION TIME	DECISION INFO		INFO. CONTENT		UTILIZ-ATION	
	AUTO	HETERO	SUP.	UNSUP.	PATTERN	REAL #	CLASS	OPT.			LIN.	COMP	LOW	HIGH	LOW	HIGH
BACK PROPAGATION			●		●	●	●		SLOW	FAST		●		●		●
COUNTER PROPAGATION			●	●	●	●	●		MED.	FAST		●		●		●
MADALINE			●			●	●		MED.	FAST		●		●		●
OUTSTAR			●		●		●		MED.	FAST		●		●	●	
ART2	●			●	●		●		FAST	MED.		●		●		●
NECOGNITRON				●	●				MED.	MED.		●		●	●	
KOHONEN NETWORK		●		●	●				MED.	FAST		●		●		●
INTERACTIVE ACTIVATION									NONE	FAST				●	●	
BOLTZMANN MACHINE	●	●	●		●	●	●		SLOW	SLOW		●		●		●
CAUCHY MACHINE	●	●	●		●	●	●	●	MED.	SLOW		●		●	●	
HOPFIELD NETWORK	●		●		●		●	●	FAST	MED.	●		●		●	
BAM			●		●		●	●	FAST	FAST	●		●		●	
SINGLE-LAYERED LMS			●		●	●	●		MED.	FAST	●		●		●	
MULTI-LAYERED PERCEPTRON			●			●	●		MED.	FAST		●		●	●	
PERCEPTRON			●			●	●		MED.	FAST	●		●		●	

Table 2: Design Issues of Neural Paradigms

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

PROCESSING COMPLEXITY OF TWO APPROACHES TO OBJECT DETECTION AND RECOGNITION

Todd Gutschow

HNC, Inc.
5501 Oberlin Drive
San Diego, CA 92121
619-546-8877

Robert Hecht-Nielsen

HNC, Inc.
and
Dept. of Electrical & Computer Engineering
University of California, San Diego
La Jolla, CA 92093

Summary

The computational complexity of a processing function is a driving factor in the implementation of that function in an operational system. Artificial neural networks offer the potential for significant improvements in the computational complexity of a number of guidance and control functions. To illustrate such an improvement, this paper considers a comparison between two different approaches to object detection and recognition: a traditional approach employing a wide field of view and constant spatial resolution throughout the image sensing and processing chain, and a foveal approach utilizing a roving "eyeball" circularly symmetric sampling grid with a radially variant resolution in the processing chain. The rationale and characteristics of these two approaches are described and compared. Quantitative evaluations of the processing loads and data transfer rates are then carried out for both approaches. These processing requirements are then compared and the operational implications of this comparison are discussed. While this paper does not explicitly discuss the efficacy of the foveal approach, references to relevant research results in this regard are provided.

1 Introduction

Object detection and recognition are image analysis operations that are of central importance for the guidance and control of many types of modern weapons. Unfortunately, except for the simplest types of objects (e.g., "hot blobs" in infrared and radar imagery) and the simplest operational scenarios, the computational and data transfer requirements connected with these operations are orders of magnitude beyond current real time on-board processing capabilities. Thus, even though competent image object detection and recognition systems can be built, most such systems cannot be employed for guidance and control because of size, mass, power, and cost limitations. What is needed is a new approach that can significantly reduce the computational burden and data transfer requirements associated with object detection and recognition.

Most current approaches to image object detection and recognition employ sensors that are designed following the tradition of television. This is logical, since an enormous technological infrastructure exists for such devices. However, most current systems continue the analogy all the way through the entire processing chain. In other words, at each stage of processing, the image pixels or features that are used are sampled at regular intervals across the entire image or subimage. While this seems particularly natural (because of our television mentality), it is not necessarily an optimal or cost-effective approach for object detection and recognition. In fact, this approach clearly ignores the design principles employed in biological vision systems.

Unlike television systems, the visual systems of animals are optimized for object detection and recognition — not for image rendering. No example of a constant resolution image sensor or image processing system exists among the vertebrates (some insects have such systems). Vertebrate animal visual systems are based upon foveal sensors and foveal processing. Such systems provide the advantage of high visual acuity within a small central field, with resolution that drops off rapidly with radial distance from the center. Such *foveal* vision systems must employ eyeballs to allow the high central resolution of the foveal sensor to be rapidly moved to different locations within the scene. The primary thesis of this paper is the claim that military object detection and recognition

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

systems built upon this foveal eyeball concept deserve intensive investigation.

The next section provides detailed descriptions of two different image object detection and recognition architectures: a constant resolution architecture, and a foveal architecture. In Section 3, these two architectures are compared. Finally, in Section 4, the potential military operational implications of this comparison are discussed.

2 Two Architectures

In this section, the designs for two hypothetical object detection and recognition systems (a constant resolution system and a foveal system) are discussed. To focus the discussion, we shall assume an image-based object detection and classification system having a 1024 x 1024 pixel imaging sensor looking down at the ground obliquely from an airborne platform which always flys at about the same altitude above the ground. It is further assumed that the range is such that the number of pixels on each object is reasonably large. The analysis in this section will concentrate on estimating the processing required to carry out object detection and classification for a single frame of this imagery. It is assumed that there are 40 object classes of interest and an average of 12 objects per frame. The next section compares the results obtained in this section for the two system concepts.

2.1 A Constant Resolution System

This subsection describes an object detection and recognition system concept that uses constant resolution imagery and constant resolution processing. The system employs a two-stage processing approach to reduce the computational burden while maintaining high probability of detection and classification rates (see Figure 1). The first processing stage performs object detection using a small number of features computed across the entire image. The result of this processing stage is a set of potential object locations. At each potential object location, the second stage of processing eliminates false alarms and classifies the true objects. This stage of processing uses a larger number of features than the first stage.

We begin with a brief discussion of the features that are used at both processing levels. Next, the two processing stages are described in detail.

2.1.1 Feature Extraction

Both the primary and secondary feature extractors use Gabor logons (see Figure 2) as the feature set. Gabor logons, originally introduced in the context of uncertainty theory for information [9], have been widely used in image processing and machine vision since Daugman extended the original work to two-dimensions [8]. Examples of Gabor logons in image processing include image compression [4], image reconstruction [13], texture segmentation [4], feature extraction and pattern recognition [3,19].

The primary advantage of Gabor logons is that they provide local spatial frequency information which has been demonstrated to be sufficient for many types of object detection and classification. A logon is constructed from a sinusoidal grating function weighted by a two-dimensional Gaussian. The sinusoid portion of the logon introduces a "waviness", whereas the Gaussian portion localizes the logon to a region of the image that surrounds the location corresponding to the mean of the Gaussian. The extent of the Gaussian and subsequently the logon is determined by the variance of the Gaussian. The mathematical form of a logon can be written as

$$G(x,y) = e^{-[(x-x_0)\alpha-(y-y_0)\beta]^2-i[u_0(x-x_0)+v_0(y-y_0)]}$$

where (x_0, y_0) are position parameters which localize the function to a region of the image, (u_0, v_0) are modulation parameters which orient the function to a preferred direction and spatial frequency, and (α, β) are scale parameters which determine the spatial extent of the logon.

As demonstrated in [8], the two-dimensional Gabor logons are not orthogonal functions. Therefore, the decomposition of an image into a set of logon coefficients cannot be performed by simply projecting the image onto the logons. Daugman [4] has developed a neural network-based method for decomposing an arbitrary image into a set of logons. This method uses a relaxation process to achieve a minimum mean squared error fit of the image to the set of logons.

While this method works well, it is very computationally intensive. Therefore, the object detection and recognition systems described below use the projection of the image onto each logon. The "cross-talk" in the resulting logon coefficients is ignored.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

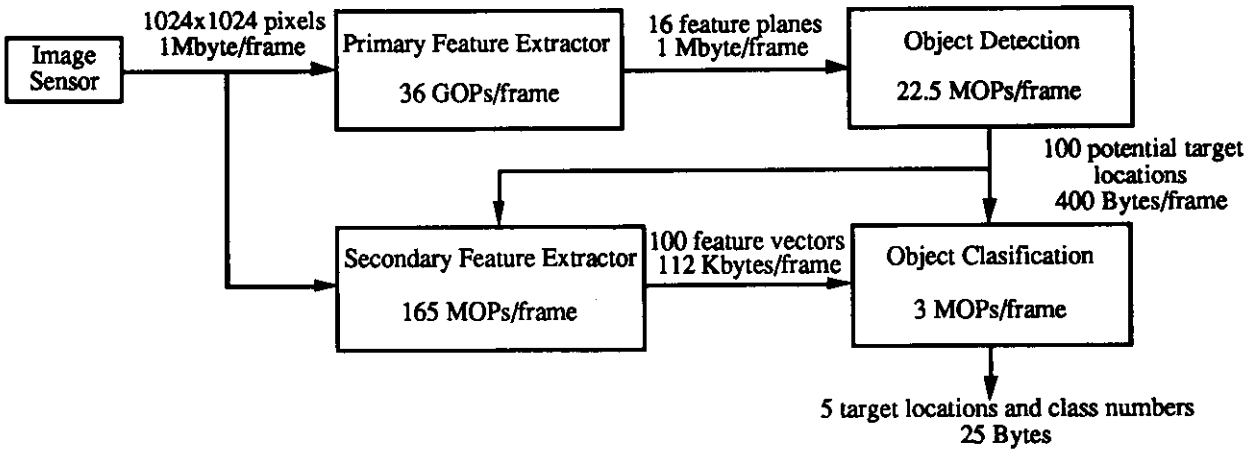


Figure 1: Constant resolution image object detection and recognition system design.

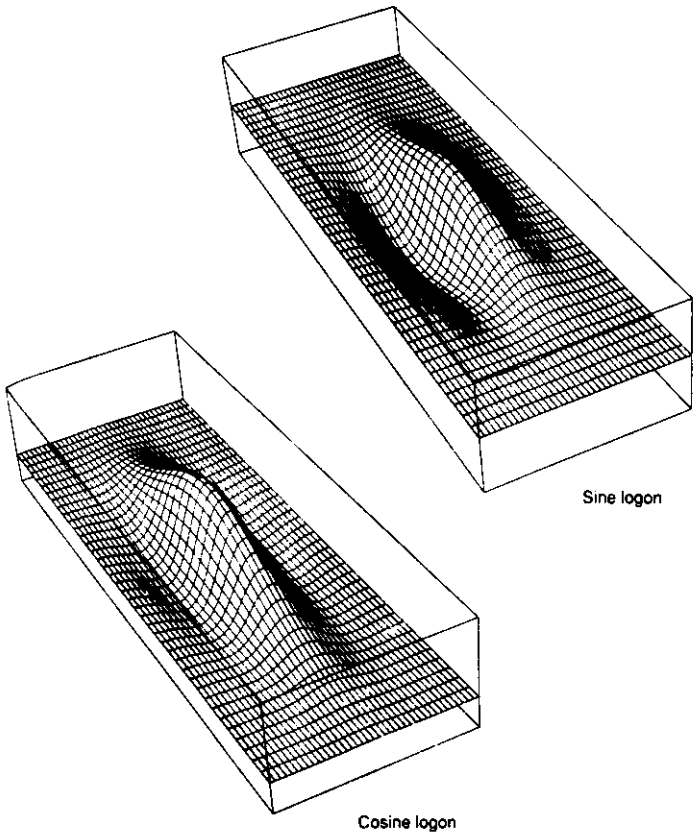


Figure 2: Spatial frequency detection kernels – sine and cosine Gabor logon wavelet features.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

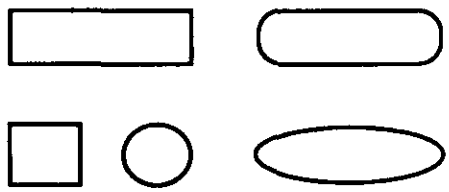


Figure 3: Examples of geometric shapes used for object detection.

2.1.2 Object Detection

The first stage of processing is object detection. The approach [20] consists of calculating a set of low resolution Gabor logon coefficients and comparing these coefficients with those derived from a set of simple geometric shapes. The use of low resolution logons provides resistance to noise and background clutter, while comparison with simple geometric shapes reduces false alarms. In order to ensure that all objects in the image are detected, the object detection process is applied throughout the image on a sampling grid of every fourth pixel.

In Figure 1, the primary feature extractor calculates 16 complex Gabor coefficients corresponding to two spatial scales, eight orientations, and two phases at every fourth pixel location. This results in 1,048,576 coefficients and requires 36 billion arithmetic operations per image. These coefficients are passed to the object detection module which compares the 16 coefficients at each pixel location to the coefficients derived from each of five geometric shapes (see Figure 3).

The comparison is performed using a normalized similarity function derived from [3]:

$$S(i,j) = \frac{\underline{G}(i,j) \cdot \underline{G}(p)}{||\underline{G}(i,j)||\,||\underline{G}(p)||} \min\left(\frac{||\underline{G}(i,j)||}{||\underline{G}(p)||}, \frac{||\underline{G}(p)||}{||\underline{G}(i,j)||}\right)$$

where $S(i,j)$ is the similarity function, $\underline{G}(i,j)$ is the Gabor feature vector at point (i,j) in the image and $\underline{G}(p)$ is the Gabor feature vector of the matching geometric shape. This similarity function is normalized to the interval (0,1).

The similarity values at each sampled pixel are then compared with a threshold. Those pixels with similarity values above the threshold are considered potential object locations and are passed on to the second stage of processing. In general, a very large fraction of the pixels will be below the threshold and therefore will not be processed further, resulting in a significant reduction in processing bandwidth between the first and second stages. This bandwidth reduction is accomplished while maintaining a low object miss rate.

2.1.3 Object Recognition

The second stage of processing is object recognition. The processing at this stage consists of extracting a number of higher resolution Gabor logon coefficients and inputting these coefficients to a backpropagation neural network [11]. The backpropagation network has been trained to classify its input into one of the 40 object classes or the "no object" class. This processing is applied only at those pixel locations that were above threshold in the object detection stage. The following discussion assumes that there are 100 such points.

At each potential object location, the secondary feature extractor calculates 56 complex Gabor coefficients corresponding to seven spatial scales, eight orientations, and two phases. The spatial scales include the two scales used in the detection processing as well as five additional higher resolution scales. In addition to these 56 coefficients, the secondary feature extractor calculates 56 coefficients at each of 4 adjacent locations for a total of 280 coefficients. These adjacent locations are typically within a few tens of pixels of the potential object location, and result in a more robust classifier that is insensitive to the precise position of the potential object location on the object.

The magnitude of each complex coefficient is calculated and the resulting 280-dimensional vector is presented to a backpropagation classification network. This network is trained to classify its input vector into one of 40 object classes or the not-an-object class. Through training on actual examples of objects and false alarms, the network is able to achieve a low false alarm rate and a high probability of correct classification. It is worth noting that this constant resolution method is itself much more economical from a computational standpoint than most classical approaches which often require yet another order of magnitude more processing per image.

2.2 A Foveal Rosette System

In the recent past, a number of researchers [14,15,16,17,18,19,10] have advocated a fundamentally new approach to image object

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

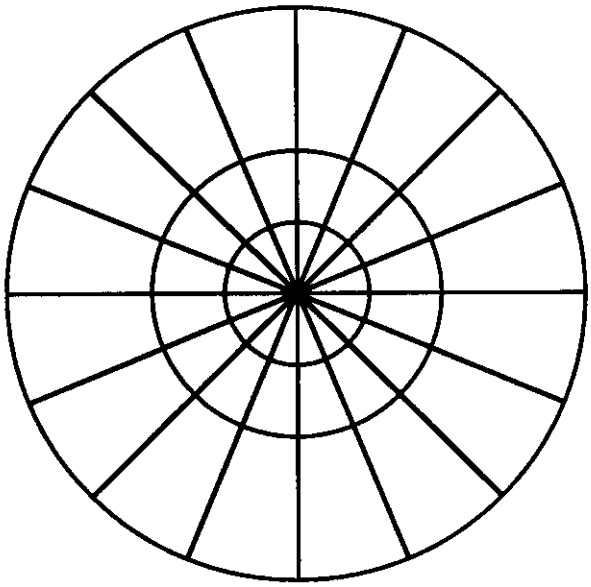


Figure 4: A foveal image feature sampling pattern rosette with 3 rings and 16 spokes. The radius of each ring is twice that of the previous ring. Spatial frequency features are gathered at the central *fixation point* of the rosette and at the points of intersection of the rings and spokes. For many objects, these spatial frequency feature sets provide a unique signature – assuming that the central point of the rosette is placed at an approximately repeatable position on the object. Rules and neural networks for moving the rosette (a *saccade generation system*) ensure that the rosette moves repeatably to similar fixation points on similar objects in different images. A saliency detector neural network can be used to determine when an object-identification-relevant fixation point (a *nexus point*) has been found.

detection and classification. This approach, which we shall call "eyeball" vision, is based upon a crude analogy with mammalian vision systems. The idea is to utilize many of the successful methods already developed in machine vision research, and modify these methods to work with a much smaller set of multiresolution wavelet features that are sampled in a non-uniform foveal pattern (See Figure 4). The idea of foveal sampling is that of having an agile, readily movable sensor that moves intelligently from fixed point to fixed point in the image to carry out the object detection, classification, tracking, and measurement functions.

As shown by Zeevi [19], Rybak [14,15,16,17], and von der Malsburg [2,3,12], the operations required to carry out most object acquisition and object recognition operations in images can be carried out using a relatively small ensemble of spatial frequency and image intensity features. In fact, for a foveated image sampling pattern, Rybak [14] proposes that as few as 833 real-valued, local image features are sufficient for carrying out many practical object acquisition and recognition functions. The work of Zeevi [19] and von der Malsburg [2,3,12] supports Rybak's conclusions. In this paper we will discuss a slightly modified version of Rybak's foveal rosette system [14].

The point on the image that lies at the center of the foveal sampling pattern (the rosette) is called a *fixation point*. As in biological vision, the movement of the rosette from one fixation point to the next is known as a "saccade". Saccades are generated primarily by exploiting feature data gathered at the sparse peripheral sampling points of the rosette. No information processing occurs during a saccade. Processing only occurs during pauses of the rosette at fixation points.

The goal of saccade generation is to ultimately move the center of the rosette to a repeatable position on each object of interest within the scene. Once an object is approximately centered in the rosette, it is classified utilizing the features gathered in the high acuity central region of the rosette. At least this is the case for compact objects (which will be the focus of this paper). Extended objects (objects larger than the two central rings of the rosette – see Figure 4) can only be classified by linking information gathered at multiple fixation points located on the object. Building an eyeball vision system for detecting and classifying such extended objects will probably be more difficult than for compact objects. Since almost all military object detection and classification problems can be solved within the confines of a compact object restriction, we will consider only compact objects.

In the presentation below, we begin with a discussion of the foveal rosette and a basic set of features that are derived from the image at each of the rosette sampling points. The feature set presented here, while sufficient for an initial development effort, should probably be expanded for an operational system to include additional important saccade generation clues such as color gradients and frame-to-frame motion cues. The issue of exactly how the foveal rosette features can be physically extracted from the scene is also discussed. Following the discussion of the foveal

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

rosette and the feature set, methodologies for object detection and recognition are reviewed. Finally, methods for developing a feature vector library for use in classification are presented.

2.2.1 The Foveal Rosette

The foveal rosette (see Figure 4) is nothing but a sampling frame. At each intersection of a radial line and a ring (or of the radial lines themselves — namely, the center), a set of features is gathered. The essential element of the rosette is the density of the features, not their regular spacing. In fact, randomly located sampling points could just as well be used, as long as their average density were to fall off properly with radial distance from the center of the rosette. Regular spacing simply makes the system easier to describe and work with. It also facilitates the efficient mathematical comparison of the feature sets gathered at different fixation points on different images.

The features that are extracted at each point of the rosette could be almost anything. For example, we might measure the local spatial frequency of the image at one or more scales. Another possibility would be to detect local image flow or measure color gradients. To allow comparison with the constant resolution system described above, we shall concentrate solely on the use of local spatial frequency features (specifically, Gabor logons, as shown in Figure 2).

The specific feature set that we will discuss in this paper is based upon the concepts of Rybak and his colleagues [14,15,16,17]. Rybak's idea is that the spatial frequency measurements at each sample point are made with both sine and cosine Gabor logon correlation kernels at eight different angles equally spaced between 0 degrees (vertical) and 157.5 degrees (the opposite azimuths are covered by the symmetry of the kernels) – see Figure 5.

We assume that the objects of interest have a spatial frequency structure such that the objects can be uniquely and easily classified by means of spatial frequency measurements at two scales that are a fixed percentage of the overall object size (and the same for all object types). Further, we assume that the object's size can differ no more than a factor ranging from 1/2 to 2 from some mean. While these assumptions may seem quite limiting, they really are not. Surprisingly, as Rybak has shown [14], the foveal spatial frequency features used here are capable of being reversed to reconstruct an immediately recognizable

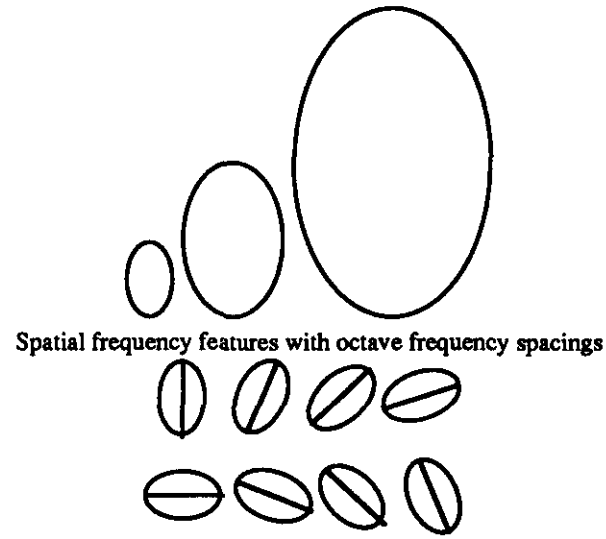


Figure 5: Spatial frequency kernels with different spatial frequencies are used at different sampling positions in the rosette. The spatial frequencies get smaller (i.e., the kernels get physically larger) by a factor of 1/2 on each successively larger ring. Sixteen orientations of each sine and cosine kernel are used at each sampling position (only eight orientations are shown here, the others are derived by means of symmetry).

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

approximate version of the portions of the original image that were sampled. In most cases the reconstruction is quite sufficient to readily visually recognize the objects in the image. Rybak obtained this result with only 833 features per rosette.

Experience from the DARPA Neural Network ATR project and on other image analysis projects suggests that most military objects can be classified by measuring spatial frequency content at no more than two spatial frequencies that are a fixed percentage of the object's size. In fact, almost all objects have this property. With the advent of inertial navigation systems, GPS, laser ranging, etc., almost all military imagery provides detailed information about the approximate scale of objects within a specific image. Thus, by means of either optical lenses/telescopes or digital image processing, the sizes of objects of interest within an image can be controlled to within a factor of 1/2 to 2 of a desired mean. This is usually simple to arrange in almost any application (e.g., a missile seeker, a reconnaissance system, an imaging radar, etc.). If necessary, the range of object sizes over which the system can function can be increased. However, this would add cost.

An important issue regarding eyeball vision is the nature of the sensing and feature extraction hardware. Clearly, the necessary sensing and feature extraction operations can be carried out using an ordinary television-type camera and digital image processing. While this will work, it may not be the most cost-effective solution in the long run. Specialized sensors that directly extract foveal rosette sampled features from a scene, such as Zeevi's CCD delay line scheme [19], may ultimately provide a more cost-effective solution. In the discussion that follows, we will not concern ourselves with the specific details of how the set of features is derived from the scene. We shall simply assume the existence of the rosette sampling pattern and the associated spatial frequency features (although we shall count the calculations required to extract them).

The specific features we will discuss are shown in Figures 2 and 5 (see [4,5,6,7,8] for details). At each sample point, we calculate Gabor logon wavelet features of a single spatial frequency at eight different orientation angles, using both sine and cosine logons. The scale of the spatial frequency features at each ring is 2 times the scale of the corresponding features at the ring just inside of it. The spatial frequency of the features used on the first ring are the same as those used at the center. The rings themselves have radii that increase by a factor of 2 between successive rings. This feature set, with some further tuning and refinement, is probably adequate for many object detection and classification problems.

2.2.2 Object Detection and Saccade Generation

In the eyeball vision concept, object detection involves two processes:

- Movement of the rosette to positions where objects are likely to be found.
- Determination that an object of interest lies at or very near the center of the rosette.

Movement of the rosette to positions where objects might be located is carried out via a set of neural networks and rules. These networks and rules determine (based upon feature information gathered at the current rosette position and at previous rosette positions) whether an object of interest is likely to be in a particular direction. For example, one rule that Rybak has explored is to follow a prominent extended edge and look for areas of concentrated "line activity" at a specific point. Such points of concentrated line activity (i.e., multiple strong line processes at different angles located at approximately the same location) are known as *nexus points*. In a typical military object detection and classification problem, objects of interest have one or more nexus points, whereas most other objects in the environment do not. Another rule might be that, if a particular edge process is followed in search of nexus points, one might later revisit this same edge process and search for it in the opposite direction. In the instance of such a rule, the periphery of each rosette would be carefully searched for evidence suggesting an extended edge process. This would then be used in formulating future saccades to examine. A saliency detection neural network can also be used to augment the rule set to determine whether or not a particular fixation point is a nexus point on an object of interest.

Obviously, some kind of feature classification process must be used in saccade generation. One option is to have a separate feature analyzer for each distinct set of saccade generation rules. For example, the extended edge following rule might use a classifier that looks for and locates extended edges in the scene using feature data from each

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

rosette. Another process might involve looking for edge intersections to determine the locations of potential nexus points. In either case, a set of rules for moving the rosette is required. Saccade generation is clearly the area of eyeball vision that has the greatest need for additional research. Notwithstanding the need for more research on saccade generation, even the current crude systems work remarkably well (see [15] for an impressive example).

2.2.3 Object Recognition

As the foveal rosette is moved about the scene by the saccade generation rules, a nexus point saliency detection neural network (a mapping network trained on points chosen by humans as being good nexus points) is used at each step to determine if an object of interest is present near the center of the rosette. The inputs to this network are the same multiresolution wavelet features used by the saccade generation rule base. This nexus detection element is used to decide whether full classification of the specific set of rosette features is called for. The ultimate goal is to compare each nexus point rosette feature set with a stored library of catalogued features.

The comparison or matching operation needs to be carried out in such a way that the system is insensitive to scale changes in the object by as much as a factor of 1/2 to 2 from the baseline scale, rotations of the object within the plane of the image around the center of the rosette, and small changes in the spatial frequency content of the object. Methods for carrying out such matching operations are known. One method is graph matching [2,3,12]. In terms of our specific features, the essence of graph matching is to take the unknown feature set and compare it with each of the known feature sets at a variety of scale and rotation offsets. For example, we might take the unknown feature vector and compare it (using an abridged Euclidean distance measurement) with a collection of auxiliary feature vectors derived from a single library feature vector. The auxiliary vectors are created by taking the library vector and rearranging the feature values to correspond to rotations of the foveal rosette by 22.5 degrees increments and scale changes of the rosette (by factors or divisors of 2) across scales of 1/2 to 2. The Euclidean distance measurement is abridged so that components which would correspond to rings that do not exist in the scaled rosette are ignored. The outer ring is also often ignored, because its features are used primarily for saccade generation.

Instead of using Euclidean distance, another approach would be to use a neural network comparison module that has been trained on a large volume of known image feature data. The output of the module is the determination of whether or not the unknown feature vector and one of the rotation/scale altered versions of the library feature vector match sufficiently or not. The use of a neural network for this function would seem promising, since the subtleties of the matching operation probably will allow a method that utilizes more of the feature content to do better than simple Euclidean distance comparison.

One of the challenges of the eyeball vision method is to find a way of matching an enormous number of library vectors with a particular unknown feature vector in a small amount of time. Cluster trees and other hierarchical indexing or content-addressable memory techniques may be useful for this purpose.

2.2.4 Feature Vector Library

The creation of a feature vector library for a particular set of objects of interest might seem very difficult, but it need not be. All that is needed is a labeling of nexus points on objects of interest in a reasonably large set of images. During the training process, the rosette movement rules are allowed to generate saccades and move the rosette around the images. Human observation of the rosette's behavior can be utilized to improve and expand the rule base. Neural networks can also be trained by humans to make expeditious saccade commands. Whenever the center of the rosette touches a labeled object of interest near a nexus point, the rosette feature vectors are captured and added to the library with a tag specifying the class of the object with which the vector is associated in the image. Rybak's work suggests that most objects will have multiple nexus points. All of the feature vectors from these points would typically be gathered and stored.

2.2.5 Foveal Object Detection and Recognition Architecture

Figure 6 shows a hypothetical foveal object detection and recognition system architecture. This system is now described. In the next section it is compared with the traditional system.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

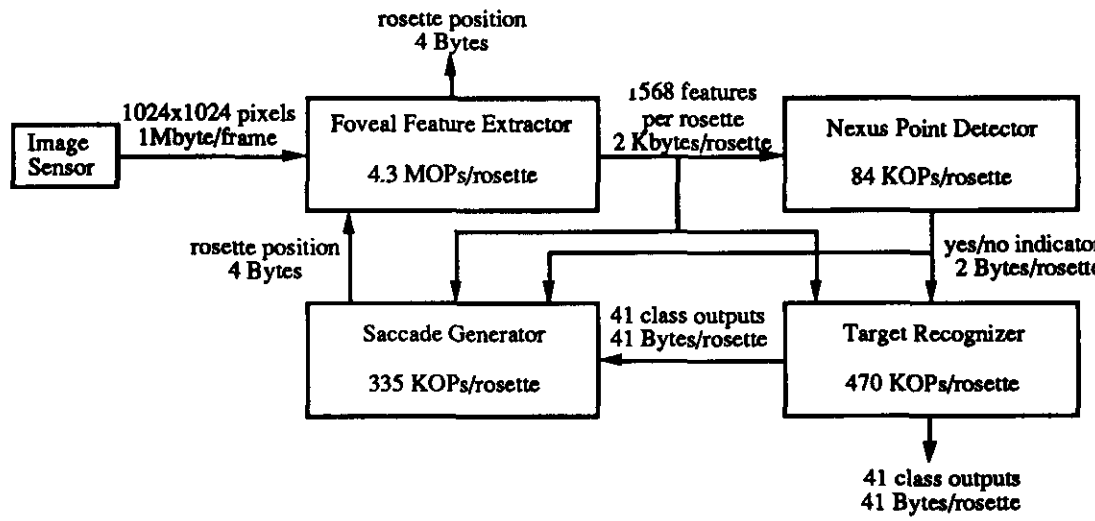


Figure 6: A foveal rosette image object detection and recognition system design.

As shown in Figure 6, the same image as used in the traditional fixed-resolution system is foveally sampled using a rosette with three rings, 16 spokes, and a center point (49 sampling points). 16 sine and 16 cosine Gabor logon wavelet features are extracted at each sample point. If we assume that digital processing is used, then each wavelet must be computed by multiplying each pixel of a wavelet template mask by each corresponding pixel value beneath the mask. The calculational burden associated with these operations is shown in the table of Figure 7. The current position of the foveal window is also emitted (this position is obtained from the saccade generator). The output of the foveal feature extraction module is a set of 32 features at each of the 49 sample points for a total of 1568 features (one byte each).

Following feature extraction, the nexus point detector module uses the foveal features to determine if the current fixation point is a nexus. This operation is assumed to be carried out by a multilayer perceptron neural network [11] with 1568 inputs, 50 first hidden layer units, 50 second hidden layer units, and two output units (one each for yes and no). While the size of this network is just a guess, experience with similar problems (such as object detection using regularly sampled spatial frequency features) suggests that a network of this size should work for a typical image object detection application. This network requires 83,652 operations to determine the nexus point classification for a single fixation point ($1569 \times 50 + 2 \times 51 \times 50 + 51 \times 2 = 83{,}652$, including bias inputs).

If the fixation point is judged to be a nexus point (a rare event), the object recognizer module is activated. The object recognizer uses a search procedure (such as a tree search) to search through a large feature vector library. It is assumed that 100 comparisons, each requiring $3 \times 1568 = 4704$ arithmetic operations, are needed to complete the search. This is reasonable, since trees can be designed to keep the search time to a low multiple of $\log N$, where N is the number of example feature vectors stored in the feature vector library (including redundant rotated and scaled versions).

Following each nexus point detection operation, the saccade generator module selects a new fixation point (unless it judges that the image search has been completed). The operation of this module is assumed to involve a combination of both rules and neural networks having a combined total computational burden four times as great as the saccade generation module, or 334,608 operations per fixation point (this is a guess based upon the saccade generation methods of Giefing [10], Rybak [15], and Schimidhuber [18]).

Let us assume (as we did with the constant resolution system considered in the previous subsection) that there are 12 objects in the image and assume that there are 2 nexus points for each object (i.e., half of the nexus points are judged by the recognition module to not be objects of any of the 40 classes of interest). This is reasonable,

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Ring	Wavelet Size (in pixels)	Arith. Ops (per pixel)	Number of Wavelets	Number of Samples Pts.	Total Ops
Center Pt.	400	2	16	1	12,800
1	400	2	16	16	204,800
2	1,600	2	16	16	819,200
3	6,400	2	16	16	3,276,800
				Total Ops.	4,313,600

Figure 7: The calculations associated with derivation of the 1568 features of a single foveal rosette.

because the nexus point detector will not be able to do as detailed an analysis as the recognition module. Let us further assume that there are a total of 100 fixation points explored in the image. We then get a total computational burden of roughly 500 million arithmetic operations per image ($100 \times 4{,}313{,}600 + 100 \times 83{,}652 + 12 \times 2 \times 470{,}400 + 100 \times 334{,}608 = 484{,}475{,}600$). Note that the calculational burden associated with extraction of the foveal features is about 90% of the total required computations. This illustrates why it would be highly advantageous if a sensor that directly extracts these features could be built.

3 Computational Complexity Comparison

In this section the real time object detection and classification system described at the beginning of Section 2 is used to compare the constant resolution and foveal approaches.

3.1 The Guidance and Control Scenario

We shall assume that the airborne object detection and classification problem described at the beginning of Section 2 is being used for guidance and control of weapon systems on-board the platform and/or of the platform itself. We shall assume a need to process 5 frames of imagery per second. To make the comparisons simple, we shall imagine that all of the data flows shown in Figure 1 and Figure 6 occur on a single shared data bus within the information processing subsystem.

3.2 Processing and Data Transfer Comparisons

In the case of the constant resolution system we have a total processing load of approximately 181 billion operations per second. The foveal system will have a total processing load of 2.4 billion operations per second. Thus, the foveal system is almost two orders of magnitude faster than the constant resolution system, assuming that both systems are implemented in approximately the same sort of hardware (see Figure 8).

In terms of data transfer, if we ignore the image input (which is the same for both) the rates for the constant resolution and foveal system operating at 5 frames per second are 5.6 MBytes per second and 1.5 MBytes per second, respectively. Here again, the foveal system is better.

4 Operational Implications

The operational implications of the comparison carried out in Section 3 are now briefly discussed.

4.1 System Envelope Parameters

The 2.5 billion operations per second processing load of the foveal system is within reach of existing or near-term processors, as is the associated 1.5 MByte per second data bus information transfer rate. Thus, although it is still in need of validation in terms of its performance, the foveal approach is well within the computational and data transfer rate envelope that can be reasonably postulated for near-future military systems.

In contrast, the constant resolution system, with its 181 billion operation per second processing load and 5.6 MByte per second data bus information transfer rate, will be more difficult to implement in real time hardware in the near future.

5 Conclusions

Clearly, the eyeball vision concept impacts more than just cost. It introduces the possibility of using knowledge regarding the spatial appearance and

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Constant Resolution System		
Module	MOPS per Frame	MOPS per Second
Primary Feature Extractor	36,000	180,000
Object Detection	22.5	112.5
Secondary Feature Extraction	165	825
Object Classification	3.1	15.5
Total	36,200	181,000

Foveal Rosette System		
Module	MOPS per Frame	MOPS per Second
Foveal Feature Extractor	430	2,150
Nexus Point Detector	0.8	4.0
Saccade Generator	33.5	167.5
Target Recognizer	11.3	56.5
Total	475.6	2,378

Figure 8: The computational requirements of the constant resolution and foveal rosette systems.

characteristic detailed internal structure of objects of interest.

Obviously, at this stage eyeball vision is little more than a concept. However, it seems worthy of further investigation, if for no other reason than the potential for computational cost savings.

References

[1] Bienenstock, E., and von der Malsburg, C., "A neural network for invariant pattern recognition", *Europhysics Letters*, **4**, 121-126, 1987.

[2] Buhmann, J., Lades, M., and von der Malsburg, C., "Size and distortion invariant object recognition by hierarchical graph matching", *Proceedings of the International Joint Conference on Neural Networks*, II 411 - II 416, IEEE Press, New York, 1990.

[3] Buhmann, J., Lange, J., and von der Malsburg, C., "Distortion invariant object recognition by matching hierarchically labeled graphs", *Proceedings of the International Joint Conference on Neural Networks*, I 155 - I 159, IEEE Press, New York, 1989.

[4] Daugman, J. G., "Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression", *IEEE Trans. Acoustics, Speech & Signal Processing*, **36**, 1169–1179, 1988.

[5] Daugman, J. G., "Pattern and motion vision without Laplacian zero crossings", *J. Optical Soc. Am. A*, **5**, 1142–1148, 1988.

[6] Daugman, J. G., and Kammen, D. M., "Image statistics, gases, and visual neural primitives", *Proc. of the Int. Conf. on Neural Networks*, IV 163 - IV 175, IEEE Press, New York, 1987.

[7] Daugman, J. G., "Image analysis and compact coding by oriented 2-D Gabor primitives", *SPIE Proc.*, **758**, April 1987.

[8] Daugman, J. G., "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters", *J. Optical Soc. Am. A*, **2**, 1160–1169, 1985.

[9] Gabor, D., "Theory of communication," *J. Inst. Elec. Eng.*, **93**, 429-459, 1946.

[10] Giefing, G.-J., Jansen, H., and Mallot, H., "A saccadic camera movement system for object recognition", in: Kohonen, T., et al [Eds.], **Artificial Neural Networks** (Proceedings of ICANN-91, Helsinki, Finland, June 1991),

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

I 63 – I 68, Elsevier North-Holland, Amsterdam, 1991.

[11] Hecht-Nielsen, R., **Neurocomputing**, Corrected Printing, Addison-Wesley, Reading, MA, 1991.

[12] von der Malsburg, C., "Considerations for a visual architecture", in: Eckmiller, R. [Ed.], **Advanced Neural Computers**, 303-312, Elsevier North Holland, Amsterdam, 1990.

[13] Porat, M. and Y. Y. Zeevi, "The generalized Gabor scheme of image representation in biological and machine vision," *IEEE Trans. on Anal. and Mach. Intel.*, **PAMI-10**, 452-468, July, 1988.

[14] Rybak, I.A., Golovan, A.V., Gusakova, V.I., and Shevtsova, N.A., "Modeling of a neural network system for active visual perception and image recognition", research report, Institute for Neurocybernetics, Rostov State University, Rostov-on-Don, USSR, April 1991.

[15] Rybak, I.A., Shevtsova, N.A., Podlatchikova, L.N., and Golovan, A.V., "A visual cortex domain model and its use for visual information processing", *research report*, Institute for Neurocybernetics, Rostov State University, Rostov-on-Don, USSR, 1990. An edited version of this report was recently published in *Neural Networks*, **4**, 3-13, 1991.

[16] Rybak, I.A., Shevtsova, N.A., and Sandler V.M., "Temporal and spatial discrimination of image features in the visual cortex", in: Novak, M., and Pelikan, E. [Eds.] **Theoretical Aspects of Neurocomputing**, 259-276, World Scientific Publishers, Singapore, 1991.

[17] Rybak, I.A., Shevtsova, N.A., and Sandler V.M., "The model of the neural network visual preprocessor", research report, Institute for Neurocybernetics, Rostov State University, Rostov-on-Don, USSR, 1990.

[18] Schimidhuber, J., and Huber, R., "Using adaptive sequential neurocontrol for efficient learning of translation and rotation invariance", in: Kohonen, T., et al [Eds.], **Artificial Neural Networks** (Proceedings of ICANN-91, Helsinki, Finland, June 1991), I 315 - I 320, Elsevier North-Holland, Amsterdam, 1991.

[19] Zeevi, Y.Y., and Ginosar, R., "Neural computers for foveating vision systems", in: Eckmiller, R. [Ed.], **Advanced Neural Computers**, 323-330, Elsevier North Holland, Amsterdam, 1990.

[20] Zhou, Y. T., and Crawshaw, R., "Contrast, size, and orientation invariant target detection in infrared imagery", *Proc. of SPIE Conf. on Image Analysis*, Orlando, Florida, March 1991.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

NEURAL NETWORKS FOR TARGET RECOGNITION

Bernard ANGENIOL
MIMETICS
5 Centrale Parc
Avenue Sully Prud'homme
92298 Chatenay Malabry Cedex
France

0 Introduction: why neural networks are interesting in target recognition problems

Modern strategic surveillance or autonomous weapons systems have performance requirements that imply the use of new and innovative data processing techniques. The ever increasing number and sophistication of modern threats, the availability of large amount of data coming from large numbers of transportable and moving sensor platforms, the extremely strong real-time defense system requirements, have resulted in increased demands on data and signal processing systems, often overwhelming conventional processing technologies.

The existence of larger numbers of threats in a cluttered environment, the existence of many false alarms, implies the use of real-time adaptive algorithms. Classical approaches have led to often costly, inflexible, algorithm intensive data processing systems; they can only meet the performance requirements through high-cost developments of co-processors.

More precisely, target recognition imply very adaptive developments, the nature of targets being different from one situation to another, the targets themselves varying in time, for example during the life of a weapon system. Various pattern recognition, from the perspective of sensor signal classification processing, are necessary, for example to detect and classify specific target signatures buried in noisy, clutter-rich signals.

Neural networks techniques, because learning from examples is a crucial phase are well suited for problems requiring an adaptive behaviour; by applying the same architectures to learn various database, on can obtain developments at relatively low costs; moreover, good fault tolerance is obtained which is particularly useful for signal processing on clutter and noisy signals. Finally, neural networks are intrinsically parallel algorithms, which allows execution on parallel neural networks processors, which may provide the answers to some of today's most

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

formidable defense system processing requirements.

1 Input signals and databases

Essentially four types of signals are used for target recognition, radar signals, infra-red images, sonar signals and TV images.

In each type of signal, several subtypes can be described, corresponding in particular to the functionalities of the system; for example, radar signals for panoramic surveillance ar every different from radar signals used in target detection in weapon systems; moreover sensors have particularities in executing the reception phase, which includes filtering, amplification, and demodulation of the signals, these procedures being generally analog.

For the needs of neural networks applications, big databases are necessary for the learning phase. Here comes the first real difficulty, because these databases have to be really representative of the problem to be solved. Two issues are then possible; either one uses data obtained from simulations, or one uses real data registered either in past conflicts, or in experimentations made by the army or the industrial groups interested in the project. In both cases, some questions are raised.

If one uses simulations, the advantages are generally that one has as many data as needed, that their cost remain reasonable, generally the cost of the development of the software simulation, that it is easy to make a database that is statistically representative of the data to be processed, and also to take into account some particular cases that appear as rather exceptional. This leads you to a software that solves pretty well the target identification problem for signals coming from the simulator. The question is then: what about real data? Are the data generated by the simulator close enough to real data to ensure good performance on real data? The answer to these questions clearly depends of the particular characteristics of the problem; one can however say that it is relatively easy to make simulations with shapes close to real targets shapes, but that the main difficulty remains in the simulation of noises and clutter; the experiments prove that resistance to artificial noises does not necessarily imply resistance to real cluttering.

If one uses real time data, the advantage is of course that the database used for learning will have characteristics close to the data used in real tests. The inconvenient is generally that, except when for the addressed problem, real databases have been recorded for years, you have to record new data to complete your database, and this may imply very high costs. Moreover, it may be merely impossible to obtain a database being representative of all exceptional patterns that may occur in your data. So, there is then little chance that the system will be able to handle these exceptional cases that he never met before.

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The best solution is most of the time to use data coming from simulations in the development of prototypes, then, to make real applications, to start from an existing or reasonable cost database, and to complete this real database by data coming from simulation. This solution is often the one offering the best price/performance ratio.

It must also be enhanced that the possibility of complementary learning phases remains open and that, consequently, if it will always be possible to enhance the performance of the system facing some particular situations that had not been forecasted originally.

2 What preprocessing?

The second problem that has to be addressed is the choice of the preprocessing. First, is preprocessing really needed? It is clear that neural networks, in many applications perform very well on raw data. This is particularly true in image processing, less in signal processing. However, if one wants to deal with raw data, one may have to make a numeric representation of the signal with very high frequency; this implies very big memory size for the system, and very very long learning time. So, to obtain equivalent results at reasonable cost, one needs some preprocessing.

But, again this really depends of the signal. For example, no preprocessing is really necessary for recognizing targets in TV images, while preprocessing seems unavoidable in most problems using sonar signals.

Most of the time, the problem of target recognition has been studied for long time using various classical methods. Adapted preprocessing was then used, and the experiments prove that the best preprocessings for classical methods are also most often the best preprocessing for neural methods. For example, in the case of radar signals, usual numeric preprocessing such as pulse compression, doppler filtering, normalization, or thresholding with constant rate of false alarms have proven to improve the performance of neural recognition.

Again, the choice of preprocessing in itself depends of the problem; for example, if you want to distinguish between various helicopters, the frequency of blades is one of the most discriminating patterns, so that you will need a doppler filtering.

But neural networks have proven to be useful either in the choice of the preprocessing, or in the preprocessing itself. Here come a few examples:

The way how neural networks can be used to choose a preprocessing has been studied in [16]. In this paper a two-stage original architecture is described: in the first stage, a first neural network with input the raw signal makes a pre-classification, identifying the type of input signal, and yielding a good choice of signal processing method; in a second stage, this preprocessing technique is applied to the signal to feed a second neural network which performs the precise

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

classification. In this example, the "preclasses" are classes such as transient sounds, surrounding noise around, quasi-stationnary noises.

An example of preprocessing using neural networks concerns texture analysis in infra-red images. Such a procedure is described in [6]. For target detection in infrared images, texture analysis is a very useful tool (while for example, because of the low dynamic, and low signal/noise ratio, contour detection is not successful). To perform the discrimination between textures, two sorts of preprocessing are used: multiresolution analysis by wavelet transform to provide interscale level energies, and the grey-level distribution. A multi layers perceptron then performs the classification.

Another example of preprocessing is described in [12]. To improve the performance in pulse radar detection, pulse compression techniques, which involve the transmission of a long duration wide bandwidth signal, and the compression to a narrow pulse, are generally employed. A neural network has been trained to perform this compression, with computational speed faster than those of the traditional approaches.

3 Extraction of features

In all pattern recognition problems, features extraction has always been a key problem. If you are able of finding discriminating characteristics of patterns in signal or in image, then making the classification is generally rather an easy task. Before the introduction of neural networks, there was essentially two ways of extracting features for a classification problem, linear algebra and experience.

The only available mathematical method was linear regression, which is still the best method to be used when the characteristic features can be obtained in a linear way from the parameters coming from the sensor; but this means the problem is easy.

In other cases, the best help for extracting features is probably to use the experience of experts in the domain. They generally are used to look for particular patterns in the signal, their approach has proven to be successful, so why not try to identify these particular patterns. Even when you use after neural networks, this has proven to save lot of time for learning. Moreover, a good choice of the features may bring to you some invariance properties that are adequate to your application. In target recognition, one generally wants to have some translation, rotation or scaling invariance; a convenient choice of features may bring this property. This is done in [9], (see § 5.2 below).

In some cases, various preprocessing and features extraction have been applied to a same problem; performances can then be compared. This is the case in [18], for automatic identification of pulse sonar noises. The first approach is based on a joint use of autoregressive modeling and wavelets transform to obtain a reduced set of parameters to feed the classifier neural network. The second is based on a two

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

dimension signal (time-scale) representation by compactly supported wavelets as inputs for the network.

If backpropagation is certainly the most popular algorithm in neural networks, a key reason is its ability to extract automatically features. In fact, you can consider the first layer of multi layer perceptron as being a feature extraction program, dedicated to the addressed problem. Moreover, the procedure of shared weights allows to impose translation or even scaling (with convenient preprocessing) invariance to these features.

In [5], an example of an extraction of visual features for lofar images is given. The identification of underwater acoustic noises is actually made essentially by human operators, either by listening directly to the noise, or by looking at the spectrogram of the noise (lofar). A backpropagation neural network has been used to extract visual features from the lofar diagrams.

Most of the time, the features that have been automatically extracted have their justification in the performances of the classification that follows them. But sometimes, specific signal features extracted by hidden units of the network can be given an interpretation. A good example is given in [11]. The problem addressed there is to classify sonar returns from an undersea metal cylinder and a cylindrically shaped rock of comparable size. It can be shown that certain hidden units correspond to an aspect-angle independent classification, while others correspond to an aspect-angle dependent strategy, encoding in particular specific spectral peaks or nulls.

4 Neural classification techniques

Backpropagation is certainly the most popular algorithm for target recognition problems, as it is for most classification problems. In the examples we are giving in §8, backpropagation is used in [4], [5], [11], [12], [13], [15], [18], [19], [20]. The main reason for that, as was said previously, is that backpropagation still works if preprocessing or features extraction that have been made before the classification are not perfect. So, it is the easiest way of making an application, main problems generally occuring in the optimization of the learning time.

For example, in [4], backpropagation has been applied to the problem of the detection of moving targets in severely cluttered environments from medium pulse repetition frequency Doppler radar signal. Performances, when compared with conventional filter bank method, proved to be much better especially in highly cluttered environments.

Another example is given in [15] for the passive detection of target-like signals in underwater acoustic fields. The input to the Neural Network is an intensity modulated signal which a measure of the power of the signal at different frequencies as time varies. The first stage of the system is an autoassociative memory whose function is to eliminate the noise. The output of this first stage is input to the

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

second stage which is a multilayers perceptron. Performances are quite promising.

Stochastic algorithms such as Boltzmann machine are used more exceptionally, generally when the characteristics of the application imply that the cost function that is used for the classification has several local minima one wants to escape from. The inconvenient for these algorithms is that they are generally computer time-consuming, so that their study is often coupled with hardware implementation.

This is the case in [2], where Synchronous Boltzmann machines are implemented on a Connection machine, for classification of boat outlines extracted from infra-red images.

In some other cases, randomness can be used to escape from flat portions of the energy landscape, as it is the case in [22], where a stochastic variant of backpropagation improves convergence rates for a sonar target recognition problem. ..

Learning Vector Quantization is a typical classification algorithm, probably the most efficient when used properly; but it has to use perfectly adapted features as inputs; in some problems, best results were obtained by making a first classification using backpropagation, then by applying a Learning Vector Quantization to the intermediate hidden units of the backpropagation. In [9], Learning Vector Quantization is applied to features that have been manually extracted to insure translation, rotation and scaling invariance (see §5.2 below)

Neocognitron is a very powerful algorithm, able to extract automatically features, even with some invariance properties. But, it is not so popular because the architecture of the network may be rather complicated, and the results very dependent of the chosen parameters. Most often, the architecture of the network corresponds to a decomposition into functionalities . In [10], the neocognitron is applied to detection, recognition, and identification of targets in infra-red images. It is proven that a neocognitron can distinguish between tanks, cows and haystacks, a difficult task when they are viewed by an infrared sensor.

Kohonen Topological maps is the most commonly used algorithm for unsupervised target recognition problems. In fact, the target recognition problems are not so often unsupervised, so that Topological maps are rarely used. One can however see an example of its use in [14] (see description §5.4 below)

Finally, Widrow's Adaline is used in some cases, even if in most cases, backpropagation is preferred (see [17] for example)

5 The key points

In most of the target recognition applications, some common difficulties arise; on can quote four:

- Multi resolution recognition
- Invariance by translation, rotation, scaling
- Movement detection

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

- Global situation analysis

5.1 Multi resolution recognition

Targets may be far or close, big or small, the accuracy of the signal may change due to noise or cluttering, so that the scale to which one has to use the signal may vary. The most popular tools for taking into account these sorts of problems is the use of Gabor functions, or wavelets functions.

In [7], a multiresolution segmentation technique is developed for signals and images, combining wavelets and neural networks. Multiresolution analysis allows localization of different contours in different scales. Thanks to this localization which characterizes the smoothness of the contour , one can hope to distinguish objects with different resolution.

A hierarchical organization of feature vectors constructed from Gabor convolutions with infra-red images at different orientations and resolutions is used in [8] for tanks recognition.

5.2 Invariance by translation, rotation, scaling

Invariance by translation, rotation, scaling is important since targets are moving objects to be recognized whatever their position, distance or orientation is. Invariant feature extraction is thus an important factor. Even if shared weights backpropagation can bring a partial answer to this, abstract features are often defined.

In [9], invariant target recognition is performed. The features are defined a priori; for example the total number of pixels with value 1, the sum of the products of pixels which are at the same distance from a designated origin, but 90 degrees apart, .. are some of these features. Kohonen's Learning Vector Quantization 2 technique is then applied to these features and gives very good performances for identifying silhouettes images of targets.

5.3 Movement detection

Targets are moving. Sensors are generally giving a picture, including position of various targets. But recognition tasks are much easier if correlation between these positions is done from one picture to the next picture. This task of tracking, or extracting trajectories is always important, and is more difficult if the frequency of picture is low, compared to the speed of the targets, as in the case of some radars, for example.

In [1], visual information about the motion of objects in an image is obtained, including the description of the trajectories. A neural networks implementation of the so-called novelty filter allows to detect motion of objects in a scene and to record corresponding trajectories.

5.4 Global situation analysis

Another difficulty to use all the information obtained is that, in many cases, isolated information concerning one target is not enough. A decision of

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

attacking a target may depend of the existence of other targets around it; the spatial relations between several targets often give important indications about their intentions.

A global analysis tool is still something prospective; however, some prototypes are developed on the subject, such as in [14], where recognition and reconstruction of spatially related grouping of various objects is addressed. The recognition and reconstruction properties are invariant under input patterns that are translated, distorted, incomplete and rotated by 30 degrees with respect to the training patterns. The algorithm is a combination of Fukushima's neocognitron, and Kohonen's multi-layered multi-topological feature maps.

6 Integration

The biggest difficulty of integrating new technologies in big systems has always been integration. This is true as well for weapon systems and neural networks. In fact, two levels of the difficulty of integration appear: the integration in the information processing part of the system, and the integration in the whole system itself. A third level of difficulty, is not addressed here, but has to be quoted: as neural networks programs are made by learning from examples, the software engineering cycle imposed by military administrations, as well as the usual validation procedures are not applicable. New agreements have to be found on this subject between military administrations and weapons systems industry.

6.1 Integration of various neural and non neural modules

Various functionalities have to be performed in the computers of weapons systems. Some of them, as seen earlier, may be well performed by using neural networks. But, all this would be of no use without integration capabilities of neural networks developments between themselves, and with other modules. Fortunately, lessons from expert systems have been learnt, and integration is a high priority for most of the neural networks tools.

An example of integration of various neural algorithms used in panoramic surveillance is given in [13]. In this application, several multi-layers perceptron trained using backpropagation are used for image prediction, pattern and image classification, image compression. Also, a model deriving from simulated annealing solves the tracking problem.

A combination of classical and neural algorithms, from noise removing, to identification is presented in [21]. A preprocessing stage removes noise from the imagery using data fusion, and performs automatic detection to obtain a range slice of the object. The object is then normalized for scale, rotation, and translation in the field of view. Oriented receptive fields are applied to extract edge strengths, followed by a neural network that does boundary completion. The object shape thus obtained is then the input of a neural network based classification stage that identifies the object.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

6.2 Integration in systems

When one wants to define a neural networks module, as seen in § 1, a strong constraint is the availability of databases. This may lead to choices that are not always compatible with the functionalities of the whole system, as most of the time the available data has not been recorded especially for the neural networks module.

Two good examples of a good integration of neural modules within the functionalities of the whole weapon system are given: in [19], a backpropagation module is used to insure the load limitation of a radar plot extractor system. The network differentiates between true and false plots before the tracking function is performed. This allows to reserve the tracking function, which is computer time consuming to the true plots.

In [20], a target recognition system based on neural networks is described, as well as the integration in the system. In this system, target recognition is performed on infra-red images in two steps.In the first step, potential targets are classified in targets or false alarms, to reduce again the computer consuming; in the second step, classification of targets as planes, helicopters or missiles, allows to adapt the tracking algorithms, to give priorities to the various targets, and to give a better evaluation of threat.

Finally, concerning integration, the real-time constraints justify the use of parallel dedicated neural hardwares. Up to now, the technology of realizing neural hardwares, has been more developed in research laboratories than in operational integration teams, so that,

7 Conclusion

Neural networks are certainly a very promising technique for target recognition, because of their adaptability, their fault tolerance and their real-time potential due to their parallelism. If, as in most of the applications of neural networks, database availability, choice of preprocessing, and features extraction are important to keep the amount of time necessary for learning within reasonable limits, the key factors for the success of the applications are multi resolution recognition capabilities, invariance of recognition by translation, rotation, scaling , movement detection capability. The integration of neural modules in weapon systems requires new validation processes, as well as a careful study to make the neural modules compatible with the sequence of functionalities of the system.

Backpropagation is certainly the most often used neural algorithm, because of its ability of extracting features. Various comparisons of performance with classical methods have been made on some examples. One is given in [17], where neural networks outperform classical algorithms for some problems of classification of natural underwater sounds. But there is no general rule, and in fact, most of the time performances mainly depend on the representativity of the database.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

8 Bibliography

[1]ARDIZZONE E., CHELLA A., SORBELLO F., (University of Palermo) Application of the Novelty filter to motion analysis INNC-90 Paris proceedings
A network learns to stress novelties occuring in a pattern representation so that visual information about the motion of objects can be obtained, including the description of trajectories and various information on motion

[2]AZENCOTT R., DOUTRIAUX A., YOUNES L., (Ecole Normale sup. Paris, CESTA Toulon) Synchronous Boltzmann machines and outline based image classification INNC-90 Paris proceedings
Application of synchronous Boltzmann machines, implemented on a Connection machine, to the classification of boat outlines extracted from infra-red images

[3]CASTELAZ P.F., (Hughes Aircraft) Neural Networks in Defense Applications IJCNN San Diego 1988
Overview of general advantages of Neural networks in defense applications (speed, fault tolerance, development effort); brief description of an application on multi-sensor passive track initiation

[4]CHIA-JIU WANG, CHWAN-HWA WU, ZIEMER R., (University of Colorado) Application of Neural Networks to Pulse-Doppler Radar systems for moving Target Indication IJCNN 90 Washington II
Detection of moving targets in severely cluttered environments from pulse Doppler radar signal, using backpropagation

[5]DE BOLLIVIER M., LEMER A., TANGUY J., (Thomson-CSF) Reconnaissance de bruits acoustiques sous-marins par réseaux multi-couches. Neuro-Nîmes 1988 Proceedings
Extraction of visual features in a lofar image using backpropagation

[6]DERYCKE N., DESMOUCEAUX J., (Thomson-CSF) Discrimination de textures infra-rouges par réseaux de neurones. Revue technique Thomson n° 23, 1991, ed. Gauthier-Villard
Texture analysis, an important preprocessing for infrared images before target recognition is made by combining wavelets transforms, greylevel distribution study, and a back propagation analysis

[7]FEAUVEAU J.C., VIENNET E., (Thomson-CSF) Multiresolution segmentation by neural networks INNC-90 Paris proceedings
A multiresolution segmentation technique is developed for signals and images, combining wavelets and neural networks

[8]FLATON K.A., TOBORG S.T., (Hughes Aircraft) An approach to image recognition using sparse filter graphs IJCNN-89 Washington proceedings
Application to tanks recognition in infra-red images of image recognition techniques based on hierarchical organisation of feature vectors constructed from Gabor convolutions with the image at different orientations and resolutions

[9] FULLER J., FARSAIE A. (Naval Surface Warfare Center) Invariant target recognition using feature extraction IJCNN 90 Washington II-595
Rotation, scaling, and translation invariant target recognition is made by combining original features extraction, and apllication of the Learning Vector Quantization algorithm

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

[10] GILMORE J.F., CZUCHRY J., (Georgia Tech Research Institute) An application of the Neocognitron in target recognition INNC-90 Paris proceedings
Use of the neocognitron for early detection, recognition and identification of targets in infrared images

[11] GORMAN R.P., SEJNOWSKI T.J., (Allied signal Aerospace Technology Center, John Hopkins University) Analysis of hidden units in a layered network trained to classify sonar targets
Application of backpropagation to the classification of sonar returns. with interpretation of specific signal features extracted by hidden units of the network

[12] HONG KEUNG KWAN, CHI KIN LEE (University of Windsor) Pulse radar detection using a multi-layer neural network IJCNN-89 Washington proceedings
Application of backpropagation to pulse compression techniques used in pulse radar detection , with good signal/noise ratio and short processing time

[13] JACQUET F., NOEL H., DERYCKE N., DESMOUCEAUX J., BUREL G., (Thomson-CSF) Application des réseaux de neurones à la veille panoramique infra rouge. Revue technique Thomson n° 23, 1991, ed. Gauthier-Villard
Various backpropagation algorithms are used for image prediction, pattern, image classification, and image compression to realize "blobs" detection and tracking in infrared panoramic surveillance

[14] JAKUBOWICZ O.G. (State University of NewYork) Multi-layer multi-feature map architecture for situational analysis IJCNN-89 Washington proceedings
Applications of Kohonen topological maps and Neocognitron to recognition and reconstruction of spatially related grouping of various objects

[15] KHOTANZAD A., LU J.H., SRINATH M.D., (Southern Methodist University of Dallas) Target detection using a neural network based passive sonar system IJCNN-89 Washington proceedings
Succesful use of Multilayers perceptron to the passive detection of target-like signals in underwater acoustic fields

[16] LEFEBVRE T., NICOLAS J.M., DEGOUL P., (Thomson-CSF) Numerical to symbolical conversion for acoustic signal classification using a two-stage neural architecture INNC-90 Paris proceedings
Combination of two neural algorithms for acoustic signal classification, one making a preclassification and choosing the preprocessing method; the second then makes the complete classification

[17] LEGITIMUS D., SCHWAB L., (Thomson-CSF) Natural underwater sounds identification by the use of Neural Networks and Linear Techniques INNC-90 Paris proceedings
Comparison of various neural and classical techniques of classification to identify natural underwater sounds

[18] NICOLAS J.M., LEMER A., LEGITIMUS D., (Thomson-CSF) Identification automatique de bruits impulsifs en acoustique sous-marine par réseaux multi-couches. Neuro-Nîmes 1989 Proceedings
Automatic identification of pulse sonar noises, using backpropagation and two sorts of preprocessing: extraction of parameters using regression, or wavelets representation of the signal

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

[19] POTIER C., (Thomson-CSF) Application des réseaux de neurones à la classification de plots radar. Revue technique Thomson n° 23, 1991, ed. Gauthier-Villard

Use of backpropagation to distinguish "true" and "false " plots to insure the load limitation of a radar plot extractor system

[20] REGEF J.L., QUIGNON J.,(Thomson-CSF) Applications des méthodes Réseaux de neurones à la classification automatique de cibles. Defense Conference Proceedings, 1991

Use of backpropagtion for automatic classification of targets in infrared images , identifying false blobs and various categories of targets (helicopters, planes, missiles,..); integration in the weapon system is addressed

[21] VAN ALLEN E., MENON M.M., DICAPRIO P. (M.I.T.) A modular architecture for object recognition using neural networks INNC-90 Paris proceedings

A combination of classical and neural algorithms to process, noise removing, detection, normalization, edge extraction, boundary completion, and identification of objects in noisy images

[22] VENUGOPAL K.P., PANDYA A.S., SUDHAKAR R. (Florida Atlantic University) Sonar target recognition using Neural Networks Neuro-Nimes 1990 Proceedings

Undersea target recognition from sonar signals by applying an original variant of multilayer perceptron based on a stochastic training algorithm

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

VISION SYSTEMS FOR GUIDANCE AND CONTROL: A TUTORIAL OVERVIEW

BY

B. Archie Bowen, Ph. D., P. Eng.,
President, CompEngServ Ltd., 19 Fairmont Avenue, Ottawa, Ontario

ABSTRACT

Vision systems are finding wide-spread use in such areas as autonomous robotics and in more mundane situations for the interpretation and/or identification of objects in images generated by various sensors. This tutorial presents an overview of the various areas in which such systems have proven successful and an introduction to the underlying theory.

The human vision system seems to be composed of a set of pre-attentive filters located in the retina which do an immediate data reduction by computing a set of features (a feature vector). These features are transmitted to the brain for interpretation as images. Synthetic vision systems are based on the same functional decomposition of feature extraction followed by interpretation.

The use of pre-attentive filters for synthetic vision systems has gained wide acceptance and produced some impressive results. The concept of pre-attentive filters is introduced and the Gabor and the Fourier-Mellin filter are shown as typical examples.

Several types of neural nets, given the appropriate input data, can be trained as interpreters to classify, complete and identify patterns. Several architectures are explored for these applications.

The first class of applications exploits the mapping characteristics of neural networks. This ability leads to a set of applications in pattern classification, pattern completion and pattern recognition. The second is in the more difficult field of object (target) recognition. Experimental results in image compression and target identification are drawn from the literature.

It is suggested that the techniques for creating vision systems appear to be applicable to very large class of problems not normally associated with 'seeing' as we normally consider it.

INTRODUCTION

The goal of replicating the capabilities of the human vision system, or perhaps more ambitiously the vision systems of various other animals with superior capabilities, is undergoing some form of realization at this time. Electronic vision systems with some of the capabilities of animals are being routinely accomplished.

While a complete electronic vision system that simulates the capabilities of animals may seem a desirable goal, in most cases some specific subfunction is all that is required. Robots, for example, need only 'see' what is required to perform their function. This may only demand the identification of a hole in a casting into which some part is to be inserted. In other cases, only predetermined shapes or objects need by identified. Thus, in most cases, while researchers may seek biologically emulated electronic systems, a vastly lower order of functionality is usually what emerges in practice.

Animal vision systems are composed of two main functional partitions. The first, in the eye, consists of a vast array of pre-attentive filters located in the retina which are either genetically coded or trained, early in life, to recognize certain attributes of the light energy they receive. The output from the filters forms a feature vector (a coded representation of the image), which is transmitted to the brain. The brain interprets this code and creates an image. The visual richness of the resulting image depends on the evolutionary demands that have been placed on the species. A frog, for example, seems to see only motion qualified by some indication of mass. The interpretation of these images is very simple; small things you try to eat, and large things you try to escape. The human system, we assume, has responded with the most complex and valid representation of the external world both through our coding mechanisms and our interpretative capabilities. On the other hand, it is very conceivable that we are missing many subtleties in the surrounding world.

Research in vision systems seems to have been concentrated in three general areas: understanding and proposing models of the animal system; modelling the generation of feature vectors, and, training neural networks to recognize certain attributes of an image. It is the latter two we are interested in. The modelling approach attempts to create feature vectors which represent the image with such fidelity that it can be reproduced (this is most useful in transmission and storage),

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

or which enhances certain attributes useful for classification or for object recognition. This later capability is perhaps of most interest to those concerned with guidance and control.

This paper is organized in four main parts: In the first we will review a model of animal vision and from this propose a model for electronic vision systems. In the second, we will review neural computation from the point of view of image processing. In the third, we address the use of neural networks to classify or identify objects presented to the input. Finally, in the fourth, we will address the use of pre-attentive filters used to more closely simulate animal vision.

VISION SYSTEMS

Animal Vision

A model of an animal vision system is shown in Figure 1. In this model, the image is decomposed by a large array of sensors which become trained to recognize attributes of the environment (such as vertical strips or bars) based on the characteristics of the received light. These sensors have been shown to have a response which is similar to a two dimensional sinusoid, damped with a (two dimensional) Gaussian decay function. This function originally proposed by Gabor has the unique property of a minimal space-time dimensionality under a Fourier transform. The functions are called Gabor-Logons, after Gabor [B-1] who studied these functions in communications theory.

A feature vector is generated based on the output of these preattentive filters and conveyed to the brain along the optic channel. The brain interprets the signals and creates an image. The interpretation process is partially genetic, and is dependent on training. Daugman [C-1] and many others have shown the validity of this model by actual measurements on the eye of various animals. While the process seems almost unbelievable in its complexity, upon reflection it seems an eminently sensible way of reducing the image data to an essential subset which can be processed in some reasonable time.

Machine Vision - A General Architecture

Systems for emulating animal vision system have a similar architecture, as shown in Figure 1. The sensors could be physical elements producing a characteristic of the image, or simulated elements whose outputs are derived by a computation on the input image. Sensors outputs are fed to a processing element which act directly to produce results (such as classification) or to an interpreter for subsequent processing. In the case of simulated filters, images are usually captured by some form of scanner which produces a pixel stream representing light intensity and/or color. The sequence of pixels becomes the synthetic image presented to the computational procedure. The characteristics of the sensors and their number depends on the application.

A PARTICULAR VIEW OF NEURAL COMPUTATION

All The World is a Vector

Neural computing, in all its paradigms, assumes some form of vector input and produces a vector output. The interpretation of the vectors and the processes of responding to the input vector vary widely, however the basic view remains unaltered.

In order to provide an image input to a neural network, it is necessary to reduce the image to a vector. This is usually done by some form of raster scan in which the pixels become the vector components. The generation of pixels depends on the sensor and on the problem. For example in scanning a satellite image of clouds, a one kilometer square is averaged to produce a pixel.

The interpretation of the output vector depends on the problem. In image classification, for example, the output would represent the estimate of which class the image is from. In target recognition, the output would be an estimate of which the class of targets the object is from.

The initial task of the system designer is to decide on the format, size and interpretation of the input and output vectors, and then on the appropriate neural paradigm needed to generate the transformation.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The neuron in most paradigms computes a distance function between its internal weights and the incoming vector. This is usually an-inner product or a vector-difference of lengths. The resulting number represents how close the input is to the neuron's weights. The output of the middle layer of a feedforward network is a set of numbers representing the closeness of the input vector with each of the neuron weights. This vector must be processed to produce the desired output.

When considering image systems it is necessary to consider the effects of the volumes occupied by the class of images and by the desired responses of the system.

Hyperspace and Hypervolumes

The world of images can be considered to occupy an n-dimensional space where each pixel is interpreted as a basis vector in the space. In a real sense then, an image is a vector and a class of similar images could be considered to occupy a volume in image space. For convenience, multidimensional spaces are referred to as hypervolumes to indicate their n-dimensional character. This distinction is important to remember since the intuitive extension of our concept of volumes does not prove valid in n-dimensional space. The sphere is the only volume that preserves its intuitive shape and metrics (volume, radius, circumference, etc.). The cube for example becomes a multipointed star.

Image classes typically occupy very convoluted volumes in image space, which demands a complex partitioning mechanism in order to separate and identify a particular class of images. Multilayer neural networks would seem an ideal mechanism to accomplish this, since, in theory, a multilayer network can create arbitrarily complex partitions. In practice, there are a multiple of practical difficulties. The most significant being that successful training demands a representative set of training examples which will expose to the neural network the complexity of the image volume, and define strictly the boundaries between distinct volumes. Since the shape of the image space is *impossible to define, the selection of representative images also becomes very difficult to* guarantee.

In addition, very large image spaces (say 128x128 pixels or higher) demand relatively large neural nets and there is no theoretical way of predicting the exact size or topology (number of layers and number of neurons per layer). Despite the theoretical capabilities of multilayer neural networks, the reality is that training by backpropagation (of an error) through many layers become ineffective, since the error, as it propagates backward, becomes less and less meaningful. Thus multilayer networks become extremely difficult to train.

The results of these and other factors usually means that the space is partitioned in such a manner that the exact partition between classes is only approximate and some intrinsic error always remains. In most cases, there is a need to reduce the dimensionality of the image space by extracting a feature vector which preserves the essential features of the image needed for the particular application.

We will look at two applications to illustrate these concerns: classification and target recognition.

CLASSIFICATION AND RECOGNITION

Introduction

The classification and the recognition problems have similar attributes, however the problems are essentially different. For our purposes we will assume that the classification problem will refer to a situation in which a number of classes of images exist in which members of the same class share some similar attributes. The problem becomes to view a new image and assign it to one of the classes. Recognition usually refers to the (more difficult) problem of viewing an object within a scene and assigning it to a class of objects. In the first case, the image is usually homogeneous, while in the second the object can be arbitrarily located in some form of background clutter.

In the case of object recognition, the object must be found before it can be recognized. The recognition algorithm must be insensitive to translation (both horizontal and vertical). It must in general also be insensitive to rotation and scale changes of the object. These constraints impose very difficult requirements on the recognition algorithm. In the case of classification, the scene may also be rotated, and translation may imposed because of the starting point of the picture. In general recognition is a more difficult problem than classification.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Image (Vector) Classification

Image classification is the task of placing an unknown image into a class of predefined image classes. The classification of clouds from satellite images, sea state from radar returns, or ground cover are typical examples.

The classification of cloud images using neural computing paradigms, for example, seems an ideal application. The classification requires trained meteorologists, the exact class differences while recognizable are difficult to quantify, and many data sets are available for training and testing the neural network. In addition there are many examples of classified images available, and even results from other approaches to the classification problem, to provide comparisons of performance. Based on these problem attributes, it would be assumed that the problem was an ideal candidate for neural technology.

The major difficulty with the application of neural networks to the classification of images is the massive data sets required to describe an image. (say 100x100 pixels or larger). This introduces major problems:

1. The computational load is immense, and adequate artificial neural network simulators running on reasonable computers become very slow in training and operation.

2. The error surface during training becomes many-dimensional and very convoluted, so that training may or may not converge in finite time.

3. The shape of the image volumes become impossible to predict and as a result the design of the neural topology to achieve an acceptable partition is subject to a error approach.

4. The selection of training examples is difficult, since the training set must be both representative (of the density in a complex image volume) and be chosen to achieve rotational and translational invariance of the images.

The first two problems have been traditionally attacked by preprocessing the raw image data to obtain a lower dimensional feature vector, which adequately represents the original image. The goal is to generate a feature vector which has lower dimensionality than the image data and which retains all the essential attributes of the image for subsequent processing. The latter two problems offer a most difficult challenge in characterizing the training and test sets.

Object (Target) Recognition

Object recognition is a term used to describe the task of picking an object or class of objects from an image. In this application, it is expected that the interpretation mechanisms will be presented with a feature vector and the output will be a decision on the existence and possibly the location of a member of a class of objects. In many cases, depending on the complexity of the system, estimates of the existence of an object can be made even when they are obscured by screens.

The major difficulty is that the background is essentially clutter from which the objects must be located and identified. This usually involves finding masses of distinguishing features and then creating a negative in black and white, followed by a search for the shape of each mass. The masses are isolated and features of each created for subsequent identification.

Image (Vector) Completion

Image completion is a simple extension of the recognition problem. In this application, the image is first classified and the classification is used to drive the display of a prototype of the class. A typical application is to display the complete image of a partially concealed object, such as a gun or a vehicle. The problem here is to define the class boundaries in such way that an incomplete vector will terminate in the hypervolume assigned to the class of objects.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

PRE-ATTENTIVE FILTERS AND VISION SYSTEMS

Introduction

In classical statistical analysis of large images, it is common to derive a feature vector which is assumed to describe the essential attributes of the image. It is assumed that images in a class will have features that are grouped in a lower dimensional hypervolume than the original image. The feature vector is then subjected to statistical analysis to separate classes, usually by some form of linear discriminate measure. A well chosen feature vector will maintain a one-to-one mapping between the image classes and the feature vector space, and substantially reduce the computational requirements for subsequent classification. Garand [C-1], for example, has proposed a set of thirteen features which are used to classify cloud images with an accuracy in the high 80% range.

In machine vision system the image is captured by a scanning technique and represented by pixels (grey scale or color), which in turn are used as inputs to computational elements which compute an element of the feature vector. The elements are called pre-attentive filters or sometimes lenses.

Pre-Attentive Filters - The Concept

Any image can be considered as a projection onto a set of bases vectors $\{L(x,y)\}$, where $\{x,y\}$ represent the Cartesian location of the image pixel in image space. The resulting image becomes:

$$I'(x,y) = \Sigma\{a_i L_i(x,y)\}$$

If the $\{L_i(x,y)\}$ is a complete orthogonal set (such as a Fourier series), then the set $\{a_i\}$ can be computed by a standard inner-product computation, and the representation has a set of well known characteristics. Orthogonal representations have been widely studied, perhaps because the calculation of the coefficients is tractable, and orthogonality is comprehensible by humans. The obvious question becomes 'How closely does the representation I'(x,y) correspond to the original image?' The answer must be qualified by several considerations, for example, "Is the goal is to create a set of coefficients that preserve the image in detail to the extent that it can be reproduced, or is the goal to extract a set of features particular to some application?"

$\{L_i(x,y)\}$ can be considered as a generalized set of filters whose individual characteristics will determine their applicability to a particular problem domain. Filters used for various image processing applications become subclasses of the generalized filter, each having a set of characteristics and parameters which distinguishes them, and defines their suitability for a particular application. Within an application, the number of filters required to achieve the necessary performance becomes the issue, since this will determine the computational complexity required to generate the features. Having generated the feature vector, the question becomes 'What processing is required to exhibit the required results?' Finally, the location of the pixels in an image need not necessarily be in Cartesian coordinates. A polar representation is used in some cases. The selection of an appropriate set of L functions becomes the major issue in most vision systems.

Feature vectors based on the apparent preprocessing performed by the human eye have been studied. These are called Gabor lenses, and image compression (and reproduction) with less than one bit per pixel has been reported. Gabor lenses are also insensitive to translation of the image. Fourier-Mellin lenses have also been demonstrated, which are insensitive to rotation. These lenses retain the essence of an image with a reasonably small feature vector.

The computation performed by these lenses correspond to that of a linear neuron. They compute an inner product of the neuron weights, and the input image. The set of such products is the feature vector. Each lens, however, requires an iterative experimental procedure to determine the individual lens parameters (the weights), and the number of lenses to achieve the desired compression or fidelity. In the experimental sciences, the lens parameters are adjusted to fit the experimental observations (say of the image preprocessing in a cat's eye). In the image classification problem, no such data exists and the parameters and number of lenses must be chosen by an iterative set of experiments, which hopefully converge to the desired performance.

The selection of statistical feature vectors tends, on the other hand, to be based on image attributes recognizable (and computable) by humans. The selection of a set of lenses to create a feature vector, depends on the requirements of the problem, i.e., image compression and reproduction,

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

image classification, etc. No well defined theory exists to guide the choice of features in any of these approaches. Experimental results are the final validation.

The weights of the first layer of a feed-forward neural network contains the same number of weights as the input space. In some sense these weights could be viewed as a synthetic image. In a trained neural net, each middle layer neuron represents a region in the image hypervolume. The selection of the number of neurons and their weights, by what ever training mechanism, must provide a minimal number of appropriately weighted neurons to yield the required classification accuracy. Other than trial and error, no procedure exists in classical neural training approaches to guarantee these results.

It is proposed here that the major difficulty is the lack of knowledge of the complexity of the hypervolumes occupied by each image class. Indeed, given that this assumption is valid, the problem is even more difficult because a description of the hypervolume is impossible to obtain. Any approach to representing the distribution of images in this hypervolume must be based on this assumption.

Figure 2 illustrates the situation in two dimensions. The image volumes are convoluted and potentially interlaced as shown. The selection of an unrepresentative training set could create a partitioning hyperplane as shown. Remembering that a neuron computes a distance measure from its internal weights, in this case it is clear that a single exemplar at the centroid will cause overlapping with the neighboring class as shown in Figure 3. A multilayer network, while potentially capable of drawing complex boundaries between such classes, must still be given the correct number of neurons and the number of hidden layers and an appropriate training set to achieve the correct partition. Clearly an image lens based on the centroid of the classes is completely inappropriate. This problem seemed to define the upper limits of classification accuracy (regardless of the length of training). The final apparently insurmountable problem seems to be that that the shape of the image volumes in image space cannot be determined.

Pre-Attentive Filters - The Theory

In general, some set of two-dimensional functions $L_i(x,y)$ defined on the same set of pixels as the image can be defined (in the familiar case, for example, the exponential functions of the Fourier series), such that a feature vector representing some estimate of the image is generated by the series:

$$F[x,y] = \Sigma(a_i L_i[x,y]$$

where the set $\{a_i\}$ represent the projections on $\{L_i\}$.

The series expansion is an attempt to build up the original function by the superposition of a set of simpler functions, which have some predefined set of desirable attributes (such as orthogonality).

The resultant F[x,y] is either identical to I[x,y] or is different is some way. F is now processed to regain I or to derive some attribute of I. Clearly if $L_i[x,y]$ is a complete orthogonal set, then F[x,y] is an exact representation of I[x,y], and the set $\{a_i\}$ can be computed as the (normalized) inner product.

$$a_i = \Sigma(L_i[x,y]I[x,y])/\Sigma L_i[x,y]$$

The inner products and the projections of a vector on a nonorthogonal set of axis are not the same, and they must be determined according to an optimization criterion. What ever the criterion it should be tractable, and meaningful in practice. Consider for example minimizing the squared-norm of the difference in the lengths of the image and the feature vector, i.e.,:

$$E = \|I[x,y] - F[x,y]\|^2$$

The difference can be computed by direct substitution at the pixel level:

$$\Sigma(I[x,y] - F[x,y])^2$$

Substituting the series expression for F(x,y) and differentiating with respect to a_i yields:

$$\delta E/\delta a_i = -2\Sigma(I[x,y]L_i[x,y]) + \Sigma 2(\Sigma a_k L_k[x,y])L_i[x,y]) = 0$$

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Satisfying this condition yields a set of simultaneous equations:

$$|\Sigma(I[x,y]L_i[x,y]) = \Sigma(a_kL_k[x,y]L_i[x,y])|$$

The left-hand side represents an inner product calculation of the projection of each lens on all the other lenses. This suggests that to minimize the mean square difference vector, we must find a set of coefficients $\{a_i\}$ such that the inner product of each vector L_i with the entire set combination of $\Sigma(a_kL_k[x,y])$ is the same as the inner product with the original image. We note that this is obviously true if:

$$F[x,y] = I[x,y]$$

By substituting for the inner summation, the result can be written as :

$$\Sigma(I[x,y]L_i[x,y]) = S_i = \Sigma F[x,y]L_i[x,y]$$

We note also that the left hand side is the inner product of the image and the basis vectors. The set of equations could therefore be represented as a matrix equation:

$$\mathbf{S = L_{ij}A}$$

Where S is a vector of length n, L_{ij} is a nxm matrix (where the terms are computed as the inner product L_iL_j) and A is a vector of length n. For example, for the two dimensional case:

$$S_1 = L_1L_1\, a_1 + L_1L_2a_2$$
$$S_2 = L_2L_1a_1 + L_2L_1a_2$$

By inverting the L_{ij} matrix we could solve for a_i and derive the exact representation of the image providing the basis set were complete. The computational task is well known providing the basis vectors are orthogonal. If they are not, the computational task is formidable for any reasonable sized matrix, and accounts for the general lack of interest in nonorthogonal representations.

In practice, the off-diagonal terms become an important indicator of the orthogonality of the pre-attentive filter. If the lengths of all filter vectors are normalized on a unit hypersphere, then the diagonal terms will be unity and the off-diagonal terms of the of the matrix, depending on their size, will show how close the vectors are to being orthogonal.

The important conclusion from this generalization is that all preattentive filters can be described by such an expansion, and their detailed character depends on the actual mathematical form of the L_i terms. Thus the choice of lenses depends on the detailed properties. We note also that each lense computes an inner product with the input image, thus each lens regardless of type has the same computational loading. The minimal computational load will thus depend only on the number of lenses required to achieve the desired feature vector.

Invariance Properties

Under most conditions encountered in real guidance and control problems, the image space must be considered unconstrained by orientation, and the image boundaries. In terms of the processing required in a vision system, this implies that the image (or object) can be translated both horizontally and vertically and arbitrarily rotated. In some cases, the image will be subjected to magnification or contraction. This requirement places a very limiting constraint on the generation of the feature vector which must, if required, preserve the set of features under the potential of all these variations.

An essential characteristics in the definition of a vision system is the limitations on translational and rotational invariances, and as a result the selection of the feature vector must reflect these requirements.

Fourier-Mellin Filters

Filters based on Fourier coefficients depend on the spectral information contained in the image. Fourier coefficients can be computed along the image vector (considered as a time series), as a two dimensional transform in x and y coordinates or as a polar transform. The major weakness of this

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

approach is the large number of coefficients required to generate the feature vector. To completely capture the image suitable for reproduction theoretically requires an infinite series. Several alternatives to reduce the number of coefficients are used in practice, including the Gabor-Logon and the Fourier-Mellin variations.

The Fourier-Mellin transform is a two step procedure which first computes the polar Fourier transform, and then the energy moments along the radius.

The approach begins with a representation of the image in polar coordinates I[r,θ]. A set of circular harmonic can be generated as:

$$F_m(r) = (1/2\pi)\int I(r,\theta)\ \exp[-im\theta]d\theta$$

where the circular harmonic frequency m is an integer. The $F_m(r)$ are referred to as a circular harmonic function (CHF). The give the energy at each frequency as a function of the radius.

The image can be reconstructed as a Fourier series by:

$$I(r,\theta) = \Sigma F_m(r)\exp(im\theta)$$

These coefficients could (and are) used in some applications as the feature vector. In some cases the energy distribution as a function of the radius of each harmonic can be used. This distribution can be modelled using moments and are computed in general as a Mellin transform:

$$M_{s,m} = \int r^{s-1}F_m(r)dr$$

These coefficients are referred to as Fourier-Mellin descriptors. In general, s can be a complex number. In practice, it is usually real. It is usually the case that a few moments will be sufficient to describe the image.

A F-M spatial filter is constructed by generating impulse response functions of the form:

$$F_r(x,y) = \{r^{s-2}\exp(im\theta)\}^*$$

where * indicates complex conjugate.

Scale and intensity invariance can be obtained by suitable normalizations of the F-M description. If the descriptors are computed for S a real number, then the scale and intensity of an image can be varied by multipliers α and k. In which case, the descriptors become:

$$|M_{s,m}|^2 = \alpha^{2s}k^2|M_{s,m}|^2$$

The scale and intensity invariance can be achieved by defining a normalized invariant feature as:

$$\Phi = |M_{s,m}|^2 / \Sigma|M_{s,m}|^2$$

All moments of the same order suffer the same multiples, and hence the feature Φ(s,m) remains invariant under translation, rotation, scal and illumination.

The advantages of this approach are

1. The representation is completely invariant .
2. The number of moments required is normally small

Gabor Filters

A Gabor filter is a variant on the Fourier approach. The Gabor filter consists of a two-dimensional Fourier transform weighted by a two-dimensional Gaussian function. The results is a filter which is translationally invariant, but is rotationally dependent.

The two-dimensional Gabor filter is represented as a series of two-dimension Gabor functions

$$I(x,y) = \Sigma\ a_iG(x,y)$$

The two-dimensional Gabor filter is the product of a 2-D sinusoid and a 2-D Gaussian weighting

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

function. This has been shown by Daugman [C-2] to achieve a minimal space-time uncertainty, and also to provide a model of animal vision systems.

The initial Fourier spectrum yields an orthogonal basis for examining the image, however, the Gaussian weighting function renders the final feature vector non-orthogonal.

Define

$$G = M(x,y)^{*} \cdot W(x,y)$$

Where M is the 2-D sinusoid and W is the Gaussian weight.

Let

$$M(x,y) = \exp\{-2\pi i(u_0 x + v_0 y)\}$$

Where u_0 and v_0 are spatial frequencies in cycles per radian.

This function can be centered at an arbitrary point x_m, y_m in the image by defining:

$$x_i = x - x_m$$
$$y_i = y - y_m$$

Thus

$$M(x,y) = \exp\{-2\pi i(u_0(x-x_m) + v_0(y-y_m))\}$$

which can be written:

$$M(x,y) = \exp\{-2\pi i(u_0 x + v_0 y) - i\phi\}$$

where

$$\phi = 2\pi(x_m u_0 + y_m v_0)$$

which is a phase angle.

The Gaussian weighting function is defined as:

$$W(x,y) = \exp(-1/2(x^2/a^2 + y_2/b^2))$$

where a and b are spatial variances.

This function can be arbitrarily centered and rotated by defining:

$$x_j = x - x_0$$
$$y_j = y - y_0$$
$$x_g = x_1\cos\theta - y_1\sin\theta$$
$$y_g = x_1\sin\theta - y_1\cos\theta$$

Thus

$$W(x,y) = \exp(-(x_g^2/a^2 + y_g^2/b^2)/2)$$

is the damping function centered at (x_0,y_0) at an orientation angle θ.

The Gabor functions are nonorthogonal. It is straight forward to work out the inner product, which is:

$$\langle G_i(x,y)\, G_j(x,y)\rangle = \exp[-\pi(u_i - u_j)^2/(a_i^2 + a_j^2) + (v_i - v_j)2/(b_i^2 + b_j^2)]$$

Daugman [C-2] has shown also that these functions achieve a maximum possible joint resolution in the conjoint 2-D visual and the 2-D frequency domains. He has shown that they achieve the theoretical lower bound on joint uncertainty in the two conjoint domains(x,y), the visual space, and (u,v) the spatial frequency domain. Defining uncertainty in each of the four variables by the normalized second moments, Δx, Δy, Δu, Δv about the principle axes he has shown that for:

$$(\Delta x)(\Delta y)(\Delta u)(\Delta v) \geq 1/16\pi^2$$

the lower bound exists for the 2-D Gabor functions.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Advantages: Compression of less than one bit per pixel has been reported.

Disadvantages: There are eight parameters that must be set either by experiment or simulation.

Object Recognition - The Work of Sheng

A group lead by Professor Yulong Sheng at Laval University [A] has published a wide variety of theoretical and experimental papers exploiting the Fourier-Mellin approach to generating feature vectors.

It is evident from the equations that the Fourier-Mellin transform does not yield translational symmetry because of the dependence of the center point chosen for the polar representation of the image. In most applications, some choice of center is necessary, often the center of gravity of the picture. or some such definition that can be fond by scanning.

Arsenault and Sheng [A]] have proposed some practical limitations on the number of harmonic components necessary to represent an image (a space shuttle) on a uniform background. A centroid of the shuttle was computed and chosen as the center of the filter. The image was reconstructed using increasingly higher order harmonic components. Their experiment showed that up to thirty seven components were needed to provide good detail (e.g., to show the tip of the tail).

They concluded form this experiment that a simple inverse relationship existed between the angular dimension of the object and the angular frequency of the CHC:

"An object detail subtending and angle of $2\pi/M_C$ at the center, where M_C (an integer) can be described by CHC orders up to M_C."

As a consequence of this, they proposed that for an image of NxN pixels, the maximum circular harmonic frequency is equal to the integer part of π_N.????

They observed also that if an object has n-fold rotational symmetry, the image has an angular periodicity of $2\pi/n$. Thus the CHC are different from zero only at the discrete angular frequencies of:

$$m = 0, \pm n, \pm 2n, \ldots$$

Two and even four-fold symmetry is not uncommon in some image classification problems.

Image Exemplars - A Generalization

When a filter array has been defined, each filter is an array of numbers corresponding to the dimensionality of the input image space. The filter could therefore be consider as a synthetic image and the result of the calculation is an inner product of the input image and the filter. Each filter in the bank contributes to the feature vector a number representing its closeness to the input image. The set of numbers must then be evaluated depending on the application.

In a sense each filter forming the feature vector could be regarded as a synthetic image in image space. The task is to find a set of such vectors to yield the feature useful for the task at hand. Since an image occupies a potentially convoluted volume, it seems reasonable to suspect that regardless of the mechanism for arranging the eights of the pre-attentive filters that the end result is a set of vectors which cover the image volumes for each class of image, in such a way as to obtain the generalization needed for the task. The filters are in some sense a set of exemplars of the of the volume occupied by the class. Based on this model there may be some hope in the future of synthesizing the appropriate exemplars as a function of the optimization requirements.

SUMMARY AND CONCLUSIONS

Summary

The principle thesis developed in the preceding has been that the architecture for machine vision systems will probably be based on some model of the animal vision system. This model suggests a two fold-partition of functionality: first the extraction from the image space of a set of features,

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

followed by some form of interpretive function responsible for creating the appropriate response.

Feature extraction may be done directly by some form of sensor suite or be preceded (as is common now) by some scanning mechanism, which in turn supplies an image representation to synthetic feature extractors. In any event, the result is the same; a feature vector must be interpreted by subsequent processing to derive the final result.

The subsequent processing may be directed toward such functions as target location and identification, classification of global image features, or the exact reproduction of the image in a reduced format from that produced by the scanners.

We have shown that the feature extraction can be represented by a general mathematical model, however, we have not been able to show how this could be applied to a particular requirements.

Conclusions

There is at this time no global approach to defining the desired task of the vision system and synthesizing the components necessary to optimize this task. A variety of feature generation mechanisms have been studied, and experimental results are available for different tasks, however no known optimization procedure exist at this time. On the other hand vision systems seem to encompass a wide variety of techniques in neural computing not normally associated with 'seeing.' Perhaps vision as understood by machines is a larger activity than that normally associated with seeing.

It seems clear however that machine vision systems will evolve according to particular needs, and the final integration of these into a human-like capability will probably be a result of advances both in the physiology and psychology of animal vision system, combined with the development of mathematical models and the creation of the appropriate processing capabilities. The future development of vision systems will occur in a fragmented way depending on specific requirements, as we increase our understanding of mechanisms for deriving the appropriate features for the task at hand.

The computational complexity of vision systems demands a high level of computer capability and it is probably safe to say that while an understanding of the process can be obtained by simulation in software, the eventual development of real-time systems will depend on hardware for both the sensors and interpretation. Neither of these possibilities are too remote. Gabor filters are now being developed by HNC and high performance analog neural chips are available from Intel. These chips (80170NW) each contain 64 neurons each containing 80 weights. The chip achieves two billion multipy-accumulate operations per second. The next few years will see special purpose vision system in wide availability and use.

ANNOTATED BIBLIOGRAPHY

A. Target Recognition

The following conference proceedings contains numerous papers directed at the problem of target recognition using neural networks. It is an obvious first reading for this particular application of neural computing.

1. "Neural Networks for Automatic Target Recognition," A Research Conference at the Wang Institute, Boston University, May 11-13, 1990.

The following papers by Sheng and associates outlines the theory and practical application of Fourier-Mellin filters to the problem of target identification:

2. Yulong Sheng, Henri H. Arsenault, "Experiments on Pattern Recognition using Invariant Fourier-Mellin Descriptors," J. Opt. Soc. Am., A/Vel. 3/No. 6, June 1986, pp. 771-776.

3. Yulong Sheng, Henri H. Arsenault, "Object Detection from a Real Scene using the Correlation Peak Coordinates of Multiple Circular Harmonic Filters," Applied Optics, Jan 15, 1989/Vol. 28/No. 2, pp.245-249.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

4. Yulong Sheng, et al, "Frequency-Domain Fourier-Mellin Descriptors for Invariant Pattern Recognition," Optical Engineering, May 1989/Vol. 27, No. 5, pp.345-357.

5. Yulong Sheng, "Fourier-Mellin Spatial Filters for Invariant Pattern Recognition," Optical Engineering, May 1989/Vol. 28/No. 5, pp. 494-500.

B. Image Compression

The following paper by Gabor is the original derivation of the Gabor Logon. It is written in older frame of reference and is somewhat difficult to read (depending on your background):

1. Gabor, D., "Theory of Communication," Journal I. E. E., London 1946, pp. 429-457.

In this seminal paper Daugman brings together the work of many previous researchers and demonstrate the image compression and segmentation capabilities of the Gabor pre-attentive filters. The paper contains a host of references to earlier work.

2. John G. Daugman, "Complete Discrete 2-D Gabor Transformations by Neural Networks for Image Analysis and Compression," IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. 36, No. 7, July 1988, pp. 1169-1179.

C. Classification

Lois Garand, "Automated Recognition of Oceonic Cloud Patterns and its application to Remote Sensing of Meteorological Parameters," Ph.d. Thesis, Department of Meteorology, University of Wisconsin-Madison, 1986.

B. Archie Bowen, and Jianli Liu, "Pattern Classification from Raster Data using Vector Lenses, Neural Networks and Expert Systems," Mapping and Modelling for Navigation, NATO AI Series F, Vol. F65, Edited by L. F. Pau, 1990.

D. Pre-Attentive Filters

An excellent paper on the general area of pre-attentive filters is contained in:

1. John G. Daugman, "Six Formal Properties of Two-Dimensional Anisotropic Visual Filters: Structural Principles and Frequency/Orientation Selectivity," IEEE Trans. Systems, Man, and Cybernetics, Vol. SMC-13, No. 13, September/October 1983.

2. John G. Daugman, "Uncertainty Relation for Resolution in Space, Spatial Frequency and Orientation Optimized by Two-dimensional Visual Cortical Filters," J. Opt. Soc. Am. A/Vol.2, No. 7/July 1985, pp. 1160-1169.

3. M. R. Turner, "Texture Discrimination by Gabor Functions," Biological Cybernetics, Springer-Verlag, Vol. 55, 1986, pp 71-82.

4. Eric Suand, "Dimensionality-Reduction Using Connectionist Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. II, No. 3, March 1989, pp. 304-314.

E. Animal Vision Systems

The following papers present experimental evidence of the functionality of animal vision systems:

1. Jones, J. P., and L. A. Palmer, "An Evaluation of the Two-Dimensional Gabor Filter Model of Simple Receptive Fields in Cat Striate Cortex," Jour. of Neurophysiology, Vol. 58/No. 6, Dec 19878, pp.1233-1258.

2. John G. Daugman, "Two-Dimensional Spectral Analysis of Cortical Receptive Field Profiles," Vision Research, Vol. 20, pp 847-856, Permagon Press Ltd., 1980.

3. John G. Daugman, "Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimization by Two-Dimensional Visual Cortical Filters," J. Optical Soc. Am., Vol. 2, No. 7, July 1985, pp. 1160-1169.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

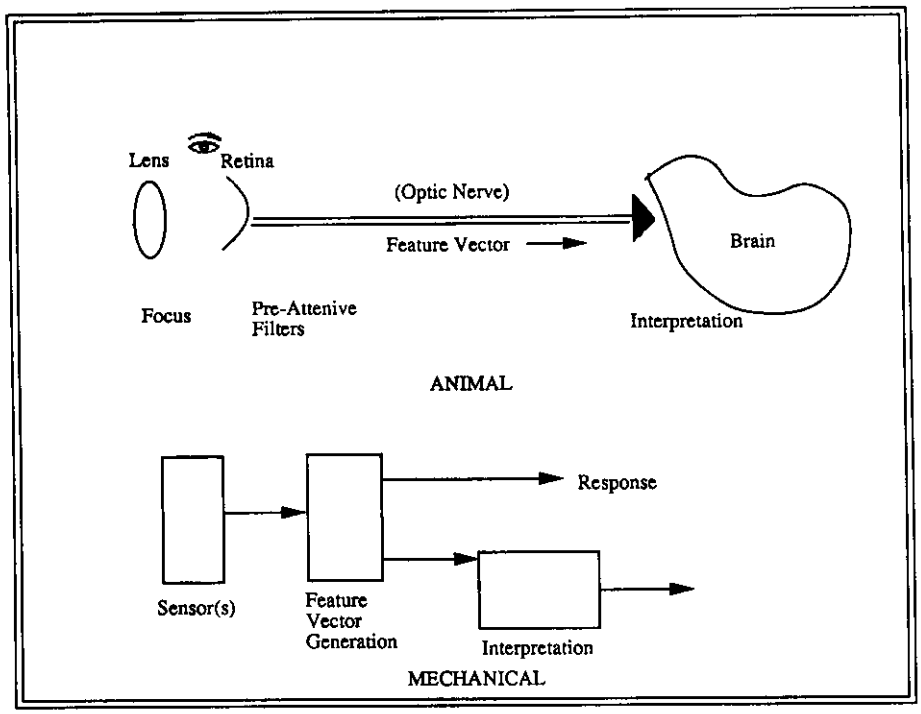


Figure 1: Vision Systems Models

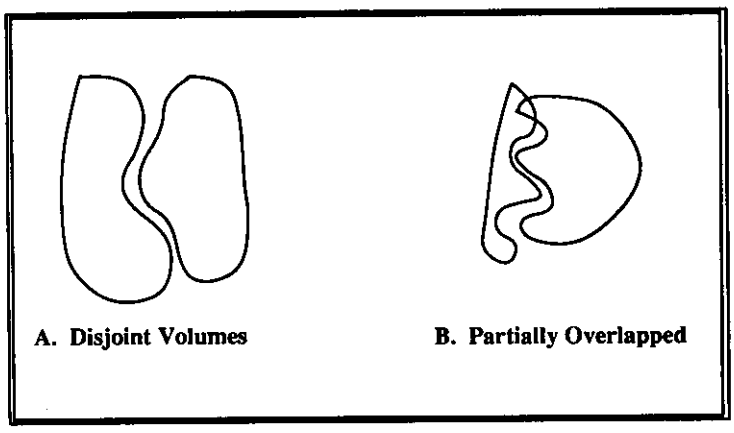


Figure 2. Image Volumes in 2-D

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

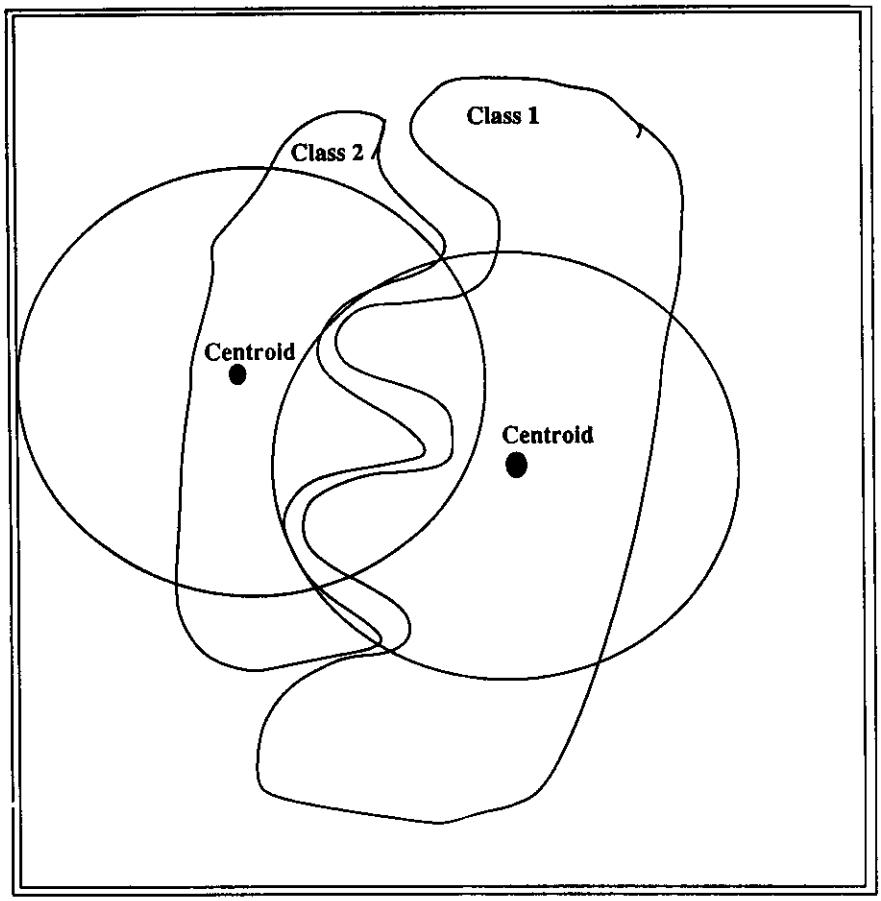


Figure 3: Image Volume Centroids

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

Neural Networks for Military Robots

Dr. W.A.Wright
Sowerby Research Centre
FPC 267 British Aerospace
Bristol BS12 7QW.

The paper, gives a short review of mobile robotic research, and through the use of three case studies which describe, in brief, current research undertaken at three establishments, indicates the role that neural networks are playing in this process and hence the impact that they may have on the military environment.

The three case studies are chosen to illustrate the advantage, in terms of speed, compactness, *and* adaptability, *of the use of these systems in what are defined as "the three essential functional areas for mobile robot control":*

- *localisation (where am I?),*
- *path planning (where do I want to be?),*
- *obstacle avoidance (is there anything in the way?).*

The first case study describes an ultra-sonic obstacle avoidance system that has been developed by the German company IBP Pietzsch for the ESPRIT II project ANNIE. The second is a description of an investigation, carried out at The Sowerby Research Centre also for the ANNIE project, into the use of a neural system for the localisation a known mobile robot by the appropriate "fusing" of data obtained from several off-board sensors. The last study describes a VLSI implementation of a localisation and path planning system that has been designed and constructed by the University of Oxford's Robotics Group.

Although it is not intended, by presenting these case studies, to portray them as the extent of the state of the art in this field it is, however, hoped that they will give a clear idea of how and why neural networks are being used in this area, and illustrate the potential advantages to be gained from their use in the field of military robotics.

Introduction

Over the past few years there has been a keen interest in the development of the military robot. This has been reflected not only by the large amount of work on mobile robotics that has been undertaken at various establishment through-out the world (see appendix A) but also by the funding that has been made available by both the Department of Defence in the USA and the British and other European Defence Ministries for research projects aimed at investigating and developing such systems. The most notable of these projects are possibly the DARPA ALV (Simpson 1987) initiative, the French ALV initiative ROVA (Savage 1991) and the British *Mobile Advanced Robotics Defence Initiative* MARDI (Bateman 1991). These projects have concentrated or are concentrating upon the production of an all-terrain autonomous mobile vehicle capable of navigating through an uncertain environment, on a reconnaissance mission mapping out the terrain or seeking out a particular target for instance. Other civil projects, the most notable being the Mars Rover (Wolfe and Chun 1987), can in some circumstances be seen as derivatives of these[1]. The use of mobile robots therefore in the military arena is not a thing of science fiction. The autonomous mobile robot is: a real, tracked, wheeled, multi-legged, or even flying vehicle.

In general, however, *robotic* systems developed and actually used in the 1980's come in a very different guise. The robots that have already found their way into factory production lines are not mobile vehicles but the static jointed arms or the more extensive assembly automated units. These are used in the manufacturing industry for the automated production of anything from PCBs through to cars or washing machines. In comparison to the static systems the functionality of the industrial mobile robotic systems are much less developed. In general most industrial mobile systems are either controlled remotely via an operator or operate in very restricted environments such as on the factory floor following a buried metal strip. The truly autonomous mobile robot, which is of prime interest to the military, is still very much of a novelty.

The major problem involved in producing a truly autonomous mobile robot is that although in many cases the processing required is understood hardware limitations prevent it from being carried out with a speed that is great enough on equipment that is small enough to be practical. As devices have become faster and faster this imbalance between processing ability, size, and processing power is being redressed. This paper intends, through a short review of mobile robotic research, and the use of three case studies which describe, in brief, current research undertaken at three establishments, to indicate the role that *neural networks* are playing in this process and hence the impact that they may have on the military environment.

[1] A list outlining the main ALV projects and institutions involved in these is given in appendix A.

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

It is not intended in this paper to produce a complete overview of the use of neural networks for mobile robots since this would be too great a task. Nor is it intended, by presenting these few case studies, to portray them as the extent of the state of the art in this field. This would be very unfair to a large number of very able workers. It is intended, however, through these case studies to *demonstrate* how this technology is being used and the potential advantages that can be obtained with the technology. Each case study reviews a piece of on-going research, attempts to highlight the relationship to mobile robotics in general, and the contribution made by neural networks in particular, summarises experimental results, and discusses their implications.

Each case study describes the use of neural networks in what, for the purposes of this paper, are defined as the three principal functional areas of any mobile robot:

- localisation (where am I?),
- path planning (where do I want to be?),
- obstacle avoidance (is there anything in the way?).

The thinking behind these functions and the constraints they impose, together with a brief résumé of the work now being undertaken in the area of mobile robotics with particular regard to the use of neural networks, are given in the next section.

The first two case studies stem from the ANNIE project (*The Application of Neural Networks for Industry in Europe*) of which British Aerospace is a full partner. This is an ESPRIT[2] II project, which is supported by the European Commission, and aims to investigate the use of neural networks in areas relevant to European industry. The project is divided into three application areas:

- image processing,
- optimisation,
- control.

Both the ANNIE case studies describe work that is being conducted within the control application area which has concentrated on the investigation of the use of neural networks in areas of particular relevance to mobile robotics.

[2] ESPRIT (European Strategic Programme for Research in Information Technology) is a European Commission funding body for collaborative research in the area of, as the name suggests, information technology. This includes many varied areas, from office systems to industrial robots.

The first case study describes work carried out by IBP Pietzsch who are a small German company specialising in the production of inertial and robotic platforms. Here, neural networks have been used to provide an *obstacle avoidance* function by studying the signatures obtained from a bank of ultra-sonic sensors placed around the robot. Although slightly artificial this study provides a graphic illustration of the use of a neural network for *sensor/motor association*, an area where the use of neural networks is becoming more prevalent.

The second case study describes the work that has been carried out recently by British Aerospace for the ANNIE project at the company's corporate research laboratories, the Sowerby Research Centre. Here a neural network has been integrated into a vision based surveillance system which, by matching the data processed by the surveillance system with data derived from a mobile robot's own sensors, is able to identify and so *localise* the robot. The work demonstrates the use of neural networks for *data fusion* and advantage to be gained from *hybrid* system which comprises several neural networks.

The last case study describes the work now being undertaken a the University of Oxford's Robotics Group under Dr Tarassenko. The group has succeeded in constructing a small working autonomous robot based on the analogue *Pulsed Stream* CMOS chips that have been designed at Oxford in conjunction with Edinburgh University's Department of Electrical Engineering. The work represents one of the first demonstrations of the integration of neural network hardware into the control architecture of a mobile robot and illustrates the advantages, in terms of *speed* and *compatibility*, that can be gained from these systems. The Oxford project is particularly concerned with the production and demonstration of an integral localisation and path planning system for a mobile robot.

Finally, the possibilities that lie in store in the area of mobile robotics are briefly reviewed in the final section. Although it is always hard to predict new developments with any certainty it is hoped that at least some idea of what the future might hold is given here.

Background

What is an Autonomous Mobile Robot?

As stated in the introduction there are now a vast variety of robotic systems. However, this paper will concentrate upon the use of neural networks in the design of mobile robotics and their impact in the military environment. The first requirement in such a discussion is to define what is meant by the phrase *autonomous mobile robot*. For the purposes of this paper it will be taken that an "autonomous mobile robot" is:

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

> any self contained system which is able to move through an "environment" as part of a process of achieving certain goals or objectives, and further is able to react to changes or unforeseen events in that environment, in order to pursue the achievement of those objectives, in "real time".

"Environment" here can mean anything from the laboratory which is structured and usually well understood through to open country which will be unstructured and possibly at best only partially known.

Here "real time" means:

> to be able to react to a stimuli at sufficient speed such that any action taken as a result of that stimuli occurs in time for that action to be relevant.

What is regarded here as "real time", therefore, changes depending upon the environment and the performance required of the robot moving through that environment. For instance real time constraints required for short term obstacle avoidance will differ from those required for long term path planning. A vehicle that requires several minutes to calculate a new trajectory around an obstacle where that obstacle is only seconds away can *not* be said to be acting in "real time", where as a vehicle that takes minutes to calculate a path that will take hours to negotiate may certainly be regarded as processing in "real time". The real time requirements for a mobile robot can therefore range from a few milli-seconds to possibly several minutes, depending upon the circumstances. The time constraints of the various embedded control loops also have a major input on the interpretation of the phrase "real time". This is a key element of the real time requirement for mobile robotic systems, and one for which appropriate processing architectures must be designed.

Functional Requirements for a Mobile Robot

The nature of the type of processing required in real time for any autonomous robot moving through a changing and uncertain environment are summarised for the purpose of this paper under the three headings:

Path planning: given the position of the robot in the environment, path planning is required to allow the robot to reach its desired destination, whilst allowing for the relevant factors in the environment such as any hazards and difficult terrain. Often these environment factors may change, due to unforeseen events, or obstacles etc. In general, therefore, it is desirable that the map of the environment and the path planning system is adaptable to accommodate these.

Localisation: given a map of the environment the position of the robot in that environment needs to be determined. In practice this can be obtained in a variety of ways. Dead reckoning, using the vehicle's inertial navigation, or odometry, can in some cases be sufficient. However, other methods are available such as the use of beacons or GPS satellite localisation. Other methods, neural implementations of which are described in the case studies, use observed sensor information and compare that with a taught or preprogrammed world model.

Obstacle avoidance: a mobile robot must have the ability to respond to unexpected obstacles in "immediate" path. Often such systems involve the use of computer vision techniques, the use of active sensors (e.g. ultra-sonics, laser range finders, etc), or a combination of both (Thorpe et al. 1987)

These three criteria, obviously, give a somewhat restricted view of the functionality of a mobile robot. The fact that any robotic system may have other subsidiary goals, such as searching, or tracking and following a particular object, has been ignored.

It is clear that the relative real time constraints for each of these functions will differ from one to the next. In the simplest case, where obstacle avoidance is purely reactive and has no input to the path planning, then this function is required to have the shortest response time. However, in practice the response times for the other functions increase as the complexity of the interrelations between the differing functions is increased.

As has already been mentioned, the high level of computation involved in creating a real time system with this degree of functionality in a relatively small space available on an autonomous system is one of the major limitations of the current systems and has troubled many research programmes such as the DARPA ALV (Simpson 1987) for example. More recent ALV programmes such as the ESPRIT II programme PANORARMA[3] (Vacherand et al. 1990) and the Universitat der Bundeswehr ALV (Dickmanns 1990) have successfully overcome this problem by using dedicated image processing hardware coupled with a distributed parallel processing system. In the case of PANORARMA this consists of Transputers together with a variety of other processors (a SUN 4 and several 68000s) (Vacherand et al. 1990).

Other Initiatives

In some limited cases the above requirements have been overcome, either by restricting the functionality of the robot or by ensuring that the robot has only

[3] Perception and Navigation Organisation for Autonomous Mobile Applications

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

to function in a well ordered and limited environment. For the industrial market limited "autonomous" systems are now available (e.g. the cleaning robot produced by Robosoft in Paris). These are generally robots with a very limited functionality designed to clean floors or transport materials. The functionality of these systems is usually limited to simple obstacle avoidance. This is achieved through the analysis of returns obtained from an active sensor or sensors, usually ultra-sonic, placed on the robot. These allow the proximity of objects to be determined and so theoretically any obstacles in the path of the robot may be detected. Any path planning that is performed in these limited systems is usually pre–programmed into the robot before operation and is therefore not adaptive.

In the military field the examples of working autonomous vehicles are not as common. Perhaps the most dramatic, as has been highlighted by the resent Gulf war, is the Cruise Missile. This uses a localisation system called TERCOM to up-date its position and so allow mid-course correction to the missile's flight path. The TERCOM system works by matching the ground terrain with a map of the ground relief held digitally in the memory of the missile. Such a system, therefore, exhibits two of the main functions of an autonomous vehicle: *localisation* and *path planning*. Obstacle avoidance particularly with regard to the terminal phase of the missile's mission is, not surprisingly, omitted.

Probably the most familiar example of an autonomous robotic system in the military and civil fields is the autonomous land vehicle or ALV. There have been and are many research projects to investigate and build ALV. A brief list, which gives some idea of the range and scope of these projects, is given in appendix A. The environment that the typical military ALV has to operate in can be very extreme. Unlike the controlled and sterile environments found on most industrial shop floors a military ALV used in anger would be expected to be able to function not only in an outdoor environment where diurnal, climatic, and seasonal conditions can have a great effect, but also under very hostile conditions that are found near and on the battle field. It is not surprising that such an environment is likely to be very unstructured and may change dramatically. Many ALVs projects attempt to use vision to guide the vehicle and for the obstacle avoidance function (Bateman 1991, Savage 1991, Buxton and Roberts 1990, Vacherand et al. 1990, Wolfe and Chun 1987, Klein et al. 1987, Simpson 1987, Mitchell and Keirsey 1984). Localisation can also be achieved by visually identifying beacons or way markers (Vacherand et al. 1990). The attraction of vision that is important in the military environment is that it is passive. However, active systems such as laser rangers have also been used (Thorpe et al. 1987, Klein et al. 1987, Buxton and Roberts 1990).

Although the functional road following ALV is now nearing reality the amount of computing power that is required to drive these robotic systems can be a limiting factor in achievable performance. This can be illustrated by the processing power required for one of the earlier ALV systems at Carnegi-Mellon University, the NavLab (Thorpe et al. 1987). This system was built into a Chevrolet van and used a vision and laser range finder system to guide the vehicle down a metaled road whilst avoiding any obstacles found in its path. The processing power required for this system consisted of the Warp systolic array and 4 sun computers. At the time this power allowed the van to travel unassisted at a speed of $\sim$ 2 miles per hour. Obviously, since the NavLab was first built computational power has improved. ALVs such as that built by Professor Dickmanns at the Universitat der Bunderswehr are able to travel at $\sim$ 50Km/h on metaled well constructed roads (Dickmanns 1990, Dickmanns and Graefe 1988). Furthermore, future improvements to this system are expected to allow the vehicle to travel on unmetaled tracks hopefully over hilly terrain. This system uses a large array of Transputers coupled with specifically designed image processing hardware. Other notable systems are the: French ROVA (Savage 1991) "Autonomous Road Vehicle", the UK MARDI (Bateman 1991) systems the eight wheeled Mars Rover (Spiessbach et al. 1987, Wilcox et al. 1987) and the six legged ASV (Adaptive Suspension Vehicle) (Spiessbach et al. 1987, Klein et al. 1987). Further details of the the large ALV projects are given in appendix A.

Neural Networks for Obstacle Avoidance and Control

It was the work on the NavLab that led to the first real use of neural network technology for the control of an autonomous vehicle. ALVINN (*Autonomous Land Vehicle in a Neural Network*) (Pomerleau 1988, Touretzky and Pomerlau 1989) demonstrated the possible advantages to be obtained by the inclusion of neural network processing for the control of the vehicle. The idea behind ALVINN was simple: use a neural network to find the road in a visual and laser ranger images (see figure 1).

An MLP (Rumelhart et al. 1986) was given pixelated image data from both a camera and a laser ranger on board the van. During training the images given were those that would be obtained if the van were leaving or off the road upon which it was supposed to drive. The MLP was then trained to provide the correct control signal (direction of motion) to bring the van back onto the road. Once trained the configured network, when implemented on the NavLab, resulted in an improvement by a factor of two over the processing speed achieved previously using conventional techniques.

Given that the ALVINN network used whole pixelated images as input it is not surprising that the size of

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

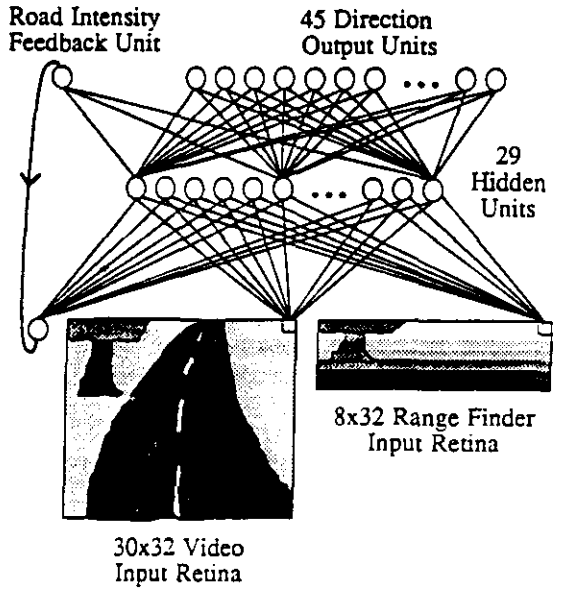


Figure 1: Schematic view of the ALVINN architecture.

the network is very large even by the standards used today. Specifically the input consisted of 960 inputs from a 30 × 32 camera image and 256 from an 8 × 32 image obtained from the laser ranger. The final configuration of the network consisted of:

- 1216 input units,
- 29 hidden units,
- 46 output units.

The output units encoded a linear representation of the turning radius the vehicle should take, with the tightest radius to the left being indicated by the left-most unit the, tightest radius to the right the right-most unit and straight on by the central unit.

Obviously to train a network of this size required immense amounts of both data and computational time. To this end, since it was difficult to gain time on the NavLab, data for training was *simulated* using actual data gathered from the vehicle as a template. Although this meant that real data was not used to train the networks it had the advantage that data could be generated that simulated the vehicle leaving or off the road with out having to place the vehicle in such a predicament.

Alternative approaches to reducing the amount of computation, applied in research elsewhere, have involved the use of processed visual data. Here, rather than input whole pixelated images, the image may be processed first using computer vision methods which are able to extract the salient features in the image: regions and their statistical features for instance. This processed data may then be fed into

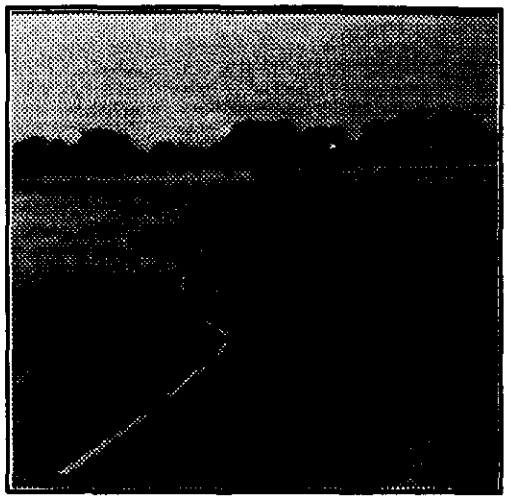


Figure 2: Road image.

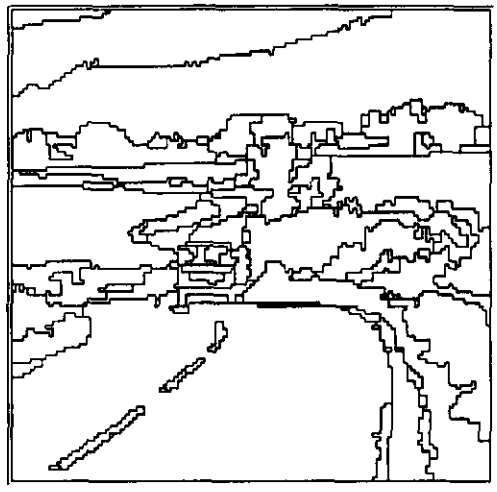


Figure 3: Segmentation of road image.

a much reduced network which will, therefore, have much smaller overheads in terms of the data required and the time required to train the network (Hutchinson 1990, Carpenter and Grossberg 1987, Jamison and Schalkoff 1988). An example of this can be found in the work of Wright (Wright 1989). Here, region features obtained from a segmented image (see figures 2 & 3) are input to a network which is subsequently trained to identify and label the road-like regions in the image (see figure 4). Having identified the road and obtained its position relative to the robot this information can be used to direct the vehicle. Such systems are now being prepared as a guidance mechanism for the MARDI ALV (Bateman 1991).

Other techniques use more structured networks which have a much reduced connectivity (Fukushima and Miyake 1982) which facilitates

This Document Provided by the ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

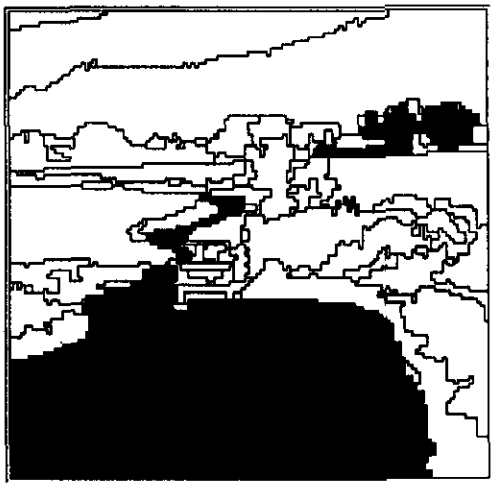


Figure 4: Segmentation of the road image with the regions labelled by the neural network as *road* displayed in black.

training on complete images.

The inclusion of neural networks to carry out the object/obstacle detection and subsequent motion control has been further developed and demonstrated at Fujitsu (Watanabe et al. 1989) and MIT/University of Boston (Baloch and Waxman 1990). Both these systems, which are described further in the third case study, use a hierarchy of networks to process the data.

Neural Networks for Path Planning & Localisation

The initial use of neural networks to perform a path planning function are exemplified by the work of Jorgensen. His work addresses the problem of determining a navigational path in a number of *different* room environments (Jorgensen 1987). Here a sonar map of each room was obtained by recording, after extensive pre-processing, eight 180° sonar scans obtained from different positions in each room. These recordings where stored in a modified Hopfield network (Hopfield 1982) i.e. the neurons could adopt a continuous value between 0 and 1. A rectangular grid of 1024 square cells was used to represent each room and a unique neuron from the Hopfield network was identified with an individual cell of the grid. The level of activity of that neuron indicated the sonar activity at that point. The idea of dividing the robot's environment into a grid is not a new one: for example the idea of *Certainty Grids* had been used earlier by Thorpe (Thorpe 1984) and Moravec (Moravec 1986) at Carnegie–Melon University, and this is the basis of the common "free space" approach, which can be used for obstacle avoidance is described in the first case study.

During the recall phase, the robot was given a single view of the room and the sonar return from that point used to prompt the Hopfield network's associative memory to complete the interior of the room. Having obtained a map of the room the path could then be computed.

This method is limited by the storage capacity of the Hopfield network (Amit et al. 1985). The use of a network with 1024 neurons meant that 37 room patterns could be stored with little problem, although in practice only 10 rooms were stored. The system was implemented on the Oak Ridge National Laboratory's mobile robot HERMIES (Hostile Environment Robotic Machine Intelligence Experimental Series) with the sonar system placed around the body of the vehicle.

The method, however, was seriously limited by the considerable storage required for the synaptic weights (there are n^2 synapses for a fully connected n–neuron network). This meant that the computations required for this associative recall required nearly 3 hours on the robot's on-board PC AT. Replacing the PC host with a 4 node N Cube gave a sizable speed up but the resultant speed and the limited recall of the Hopfield network limited this approach (Jorgensen 1987). The idea of grid localisation using a neural network has been adopted else where (Tarassenko et al. 1991), and this work forms the central element of one of the case studies presented here.

Neural Controllers

Although the generality of the subject of the application of neural networks to *control* systems falls somewhat outside the scope of this paper, their use is important and considered worthy of mention. The use of neural networks for the control of a vehicle's motion has been taken up by many workers in the field. Possibly one of the most well known is that of Widrow with "The truck backer up" (Widrow 1990). Other work has used various strategies: e.g. networks have been used to provide a trainable inverse model of a system based on the input/output observations of the plant, Kawato (Kawato et al. 1987), Chen (Chen and Pao 1989). The inverse model is then used to generate control signals.

Other methods use two networks, one to model the control response of the system and the other to produce control decisions. A great deal of the recent developments in this area within Europe have been reported in the proceedings of the IEE conference *Control 91*. A large proportion of this work has concentrated upon exploitation of the non–linear and adaptive nature of a neural network to provide the adaptive feed-back controller that is central to some non–linear predictor adaptive control systems, see figure 5. Such systems have great relevance to the driverless, or pilotless vehicle. Here, without a human controller, the vehicle will have to be able to adapt to both slow changes in the characteristics of the vehicles performance, e.g. the lightening of the vehicle

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

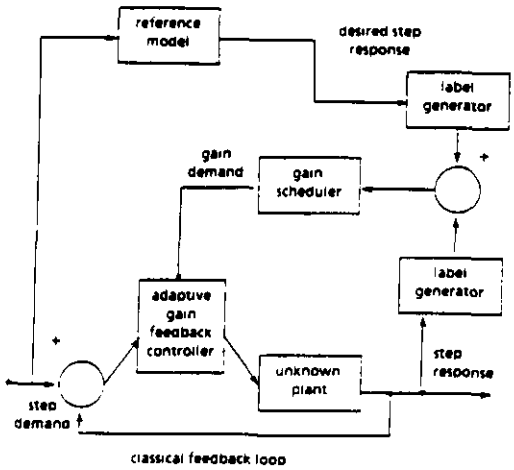


Figure 5: Predictor Adaptive Gain Control Architecture

as the fuel load decreases, and more importantly sudden changes e.g. sudden changes in terrain, weight changes caused by the delivery of of munitions, or as has also been suggested damage to control surfaces on aircraft (White and Sofge 1991).

A good example of the use of neural networks in adaptive control can be found in the papers by Brown et al and Ince et al. Here a recurrent layered network is used to model the non-linear response of the vehicle to the control signals it is given (Brown et al. 1991, Ince et al. 1991). The controller network runs in parallel to the predictor model and adapts as the vehicle's response changes by back propagating an error signal generated by differencing the output of the reference model and the actual response obtained from the vehicle that the network is modelling. The network therefore acts as an adaptive gain feed-back controller (see figure 5) for the vehicle which, as is demonstrated in Brown et al and Ince et al, can be integrated directly into a conventional predictor controller.

Neural Hardware

The brief review given above has tried to give an idea of the breadth of work on the application of neural networks in the areas in sensing, control, path planning, and obstacle avoidance. The more recent work in this area has started to demonstrate the advantages to be gained from the use of these systems. However, it is the contention of the author that the true worth of using a neural network can not be realised unless the network that has been designed can be implemented on appropriate hardware and integrated into a complete processing system. This view has particular merit in the subject area that concerns this paper. Here any working system has to be realised in hardware that will provide the appropriate real time performance. Furthermore, this hardware must be small enough and flexible enough to fit into the control system of a mobile robot. These constraints can be particularly harsh in military environment[4].

It may be argued that the potentially high speed, compact nature of a neural network, once implemented on the appropriate hardware technology, is perhaps the greatest advantage of these systems over and above that of more conventional processing techniques. Obviously, this is not the only view of the worth of neural networks, but it is a view that is of great importance in the field of mobile robotics. Although neural systems generally do not easily map onto conventional sequential or parallel hardware all the systems that have been mentioned so far in this paper have used some form of on board *non-neural* processor. With the advent of dedicated hardware (LeCun et al. 1990, Murray et al. 1990, Holler et al. 1989) a further reduction in size and increase in performance can now be anticipated. Perhaps the first example of the use of such hardware is the work carried out by the Robotics Group at Oxford University (Tarassenko et al. 1991), a description of which forms one of the case studies which are now described.

Case Study 1: Obstacle Avoidance Using an Ultra Sonic Array

The use of data from ultra-sonic arrays, or for that matter other dense range dependent data, for obstacle avoidance is quite widespread (Buxton and Roberts 1990, Jorgensen 1987). The techniques developed to provide an obstacle avoidance function using this type of data divide into two.

Configuration space: this is a derivative of the *certainty grid* (Elfes 1987) idea that was explained earlier. Here a dense map of the environment is obtained via either an active or passive sensor. This map is then used to compute "free space corridors", which allow for the size of the vehicle, around obstacles present in the environment. There are many difficulties with this method: to generate the configuration space requires a large amount of processing, the method is not body centred and therefore the view of the environment may not be consistent with the view seen from the vehicle once it has moved to a different position, and without continuous updating the method cannot cope with moving obstacles. The method is characterised by an explicit recalculation of the robot's path around the obstacle.

Potential field methods: this method (Khatib 1986) usually relies upon mon-

[4] I intend to leave the difficult question of verification of such neural systems until the conclusion

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

itoring the signature of an array of sensors capable of generating range dependent data in a dense pattern around the vehicle. Simply, this *method uses the range data to determining the* position of obstacles relative to the robot. These obstacles are then considered to have a repulsive potential which repels the robot and so prevents the vehicle from hitting the obstacle. Unlike the previous method this technique is body centred and since the method uses data that is continually updated is able to deal with moving obstacles. Here, *the robot moves "relatively" with an* implicit re-calculation of the path.

The work described here is the result of an investigation carried out by the German company IBP Pietzsch for the ANNIE project into the use of neural network architectures for reactive obstacle avoidance. Ultrasonic signatures are processed to produce a control signal necessary to ensure that the vehicle avoids obstacles placed in its path. The method that is developed here is somewhat similar to the potential field method described briefly above. Although the results of this investigation are obtained via simulation and do not use real data it is hoped that they give a graphic description of the potential use of neural networks for sensor/motor integration. A more detailed description follows.

Network Implementation

Briefly the simulation used is composed of:

- a mobile robot that is equipped with 9 ultra-sonic sensors, as is shown in figure 6. These 9 sensors are arranged in groups: 4 pointing forward, 2 on each side of the robot pointing to the left and to the right, and one sensor pointing to the rear,
- The environment in which the vehicle moves consists of a room containing obstacles of differing shape and complexity (see figure 7).

The robot is allowed to move through the environment using 8 possible motions:

1. stop,
2. fast forward,
3. slow forward,
4. turn right by 45°,
5. turn left by 45°,
6. turn right by 90°,
7. turn left by 90°,
8. slow backwards.

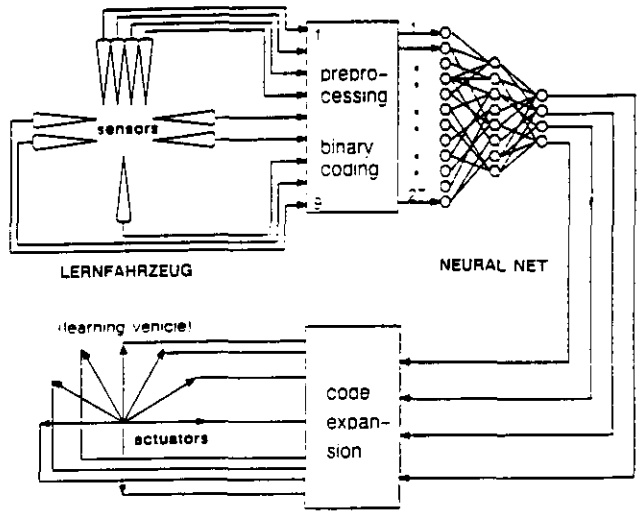


Figure 6: Control Architecture for Simulated Vehicle

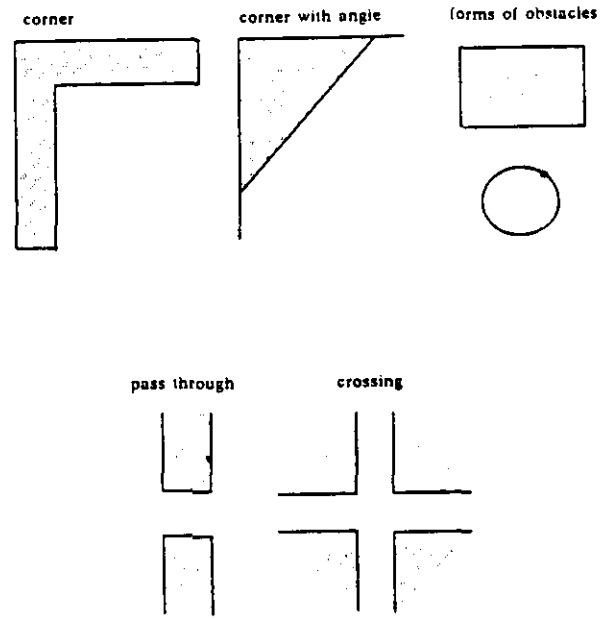


Figure 7: The 5 primitives used to construct obstacles for simulated environment

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

As the control architecture diagram (figure 6) suggests the output from the ultra-sonic sensors were extensively preprocessed. The preprocessing range gated the sensor output, before it was put into the neural network, into 15 range values that were spaced logarithmically. To ensure that this range data was presented to the network in a robust manner the range values from each of the sensors were encoded in a 4 bit coding[5] designed such that the codes for neighbouring range gates were separated by a small Hamming distance. Thus similar codes would be obtained for range values that just fell either side of a range boundary.

The 4 bit coding from each of the 9 sensors gave a 36 bit binary input to the network that was used, which was a 3 layer MLP. The output layer of this MLP consisted of 3 units. These encoded the 8 possible control instructions that the robot should receive. Again, as in the input coding it was ensured that this coding was robust to small fluctuations and so similar motions were given codings separated by small Hamming distances.

To train the network the vehicle was placed repeatedly in close proximity to 10 typical rectilinear obstacles such as: corners, corridors, walls, and walls with openings (see figure 8). To ensure that the vehicle is able to meet all the situations that it may find itself in, the position and orientation of the vehicle was also varied. This ensured that the configuration generated on the network during training was as general as possible. In presenting the vehicle to the various obstacles the sensor signals from the 9 sensors were generated and this data together with a motor response signal given by an operator was given to the network to allow it to train.

As with all MLP simulations the precise construction of the network is not clear at the outset and empirical data has to be gathered to determine the number of hidden units and to set the back propagation (Rumelhart et al. 1986) parameters. The final configuration obtained from these experiments was an MLP with:

- 36 input,
- 8 hidden (the only unknown variable),
- 3 output.

For this straight forward problem satisfactory convergence was obtained after the repeated presentation of 10 obstacles as shown in figure 8. The slow nature of the MLP error back-propagation required over 20,000 presentation of the 10 obstacles.

The slow learning rate obtained has since been greatly improved by the use of direct analogue input from the sensors themselves. Here a slightly different architecture has been used.

[5] Obviously this assumes that a perfect return signature is obtainable from these sensors which is usually not possible.

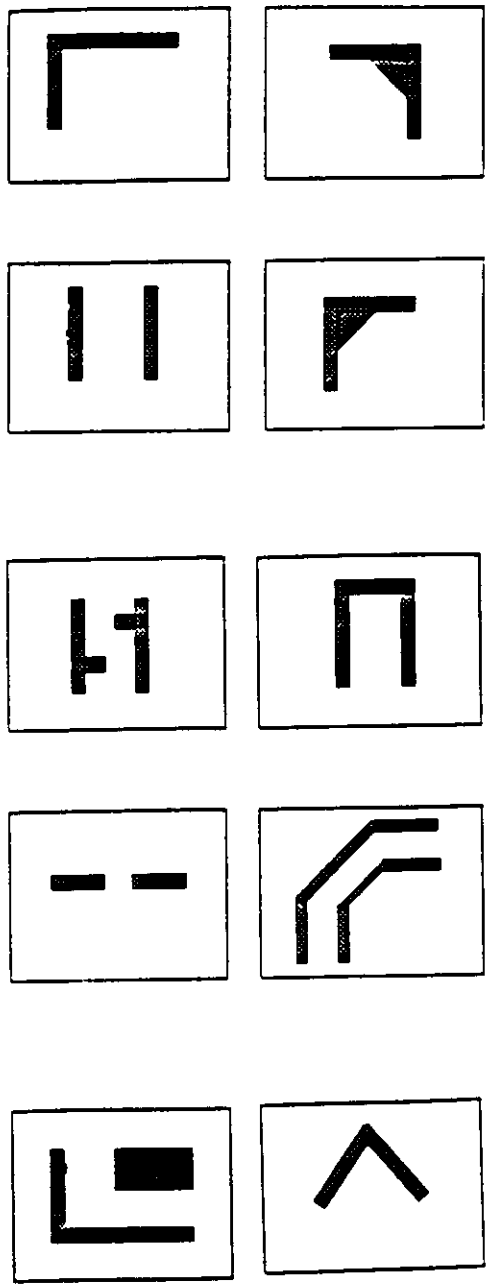


Figure 8: 10 obstacles used to train the network

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

- 10 inputs, 9 returns from the ultra–sonnic sensors which are inverted, together with the current speed of the vehicle,
- 3 hidden units,
- 2 output units, indicating the change in the vehicles speed and angle of turn.

Although the performance of this network is not radically different from the previous design the use of analogue inputs allows the network to be much smaller. The small size of the network allows the training to be accomplished much more easily.

Discussion

Once trained it was found the the network was able to negotiate perfectly the obstacles it was given to train upon. Testing the network on a set of rectilinear obstacles upon which it had not been trained indicated that the network could generalise to obstacles with which it was not familiar. The network was able to negotiate the new obstacles only failing to avoid these in a small percentage (1%) of the cases. Furthermore, this performance could be increased by retraining the network on those cases which it found difficult. Perhaps surprisingly it was found that the performance of the neural controller depended heavily upon the identity of the operator who was used to give the direction of motion of the vehicle for each training situation. This highlights an important point regarding the adaptive nature of a neural network, in that the final configuration of the network can be heavily dependent upon not only the nature of the training data used to configure it but also the way in which that data is presented.

The dependency of the final configuration of a neural network after training with respect to these factors obviously has great bearing upon the variability of such systems. This variability can be reduced if the data used to train the network and the way that data is presented is tightly specified. This point and others related to the verification of these systems are discussed further in the last section.

Although limited in its scope it is the intention of this case study to demonstrate how a neural network can be used to perform a sensor/motor association. The use of a neural network to produce a reactive motor response to a given stimulus has many advantages over the more conventional approaches that have been described. Apart from the speed and the compact nature of these devices once implemented in VLSI silicon the highly parallel nature of these systems (which allows data from many processors to be processed simultaneously) coupled with their adaptability (which allows data to be processed without the requirement for direct calibration since this can naturally be configured into the network during training) makes neural systems very useful for reactive control. The use of neural networks to perform this association has been carried out successfully in a number of other areas related to robot control (Waxman et al. 1988, Peterson 1991).

Case Study 2: Localisation from Off Board Sensors

The work presented here represents part of that carried out by British Aerospace's corporate research centre The Sowerby Research Centre for the ESPRIT II project ANNIE. The investigation is concerned with the localisation of a robotic vehicle. However, rather than performing this localisation using sensors placed on the robot the investigation is concerned with the somewhat different problem of localising the robot using a surveillance system separate from it. It is assumed, not unreasonably, that the surveillance system is able to provide both the bearing and range of the objects it detects but is *not* able to consequently identify the object. Localisation of a known vehicle is, therefore, not possible if there are other targets present without the use of prior knowledge such as the robot's approximate position or the identity of the objects.

In general, military systems overcome this problem by using say IFF techniques or allowing the vehicle in the field to determine its position against a known frame of reference, using GPS for instance, and communicating this back to the surveillance system. However, the use of either of these systems is not always desirable or possible. An alternative is to allow the vehicle to simply communicate to the surveillance system the present trajectory and then through a process of "data fusion" determine which surveillance track best matches the trajectory and so identify and localise the vehicle. This last alternative has the advantage in that it does not rely upon an external system such as a satellite, nor would it be easy to jam or suffer from external interference. To perform this "data fusion" however, which requires the data received from a vehicle to be correlated with all the objects detected by the surveillance system, may require some very intensive computing. Furthermore, the correlation between the signals may not be obvious and could well be non-linear.

This investigation looks at the possibility of using a neural network to perform this correlation in the hope that the high bandwidth and non-linear adaptability of neural networks will be of advantage, and demonstrates this in a real laboratory environment. To carry this out the investigation exploited a distributed real time surveillance system that had already been built for the ESPRIT I project SKIDS[6] and constructed in a $10m \times 10m$ room in one of the laboratories at The Sowerby Research Centre. This environment together with the mobile robot that was used for the investigation are described further in the following section.

[6] Signal and Knowledge Integration with Decisional control for multi-sensory Systems

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

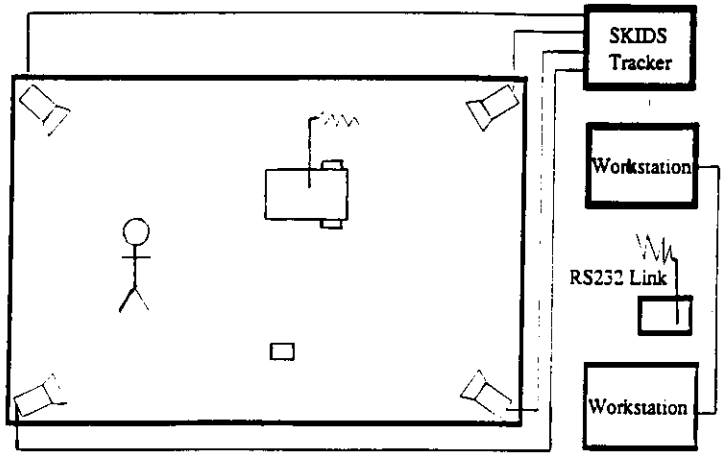


Figure 9: Schematic View of SKIDS Environment

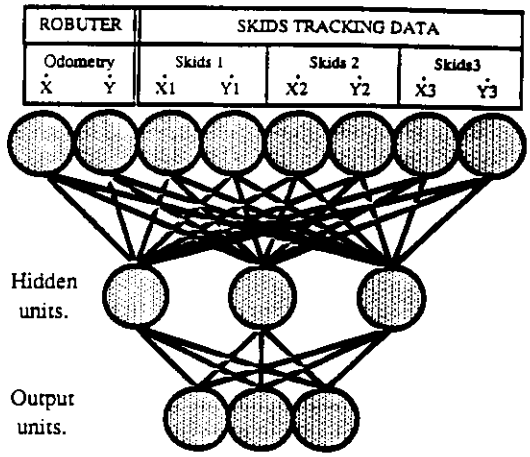


Figure 10: Large Multi-input network

The Environment

The SKIDS surveillance, or tracking, system monitors the room through the use of four monochrome CCD cameras, one mounted at each of the four corners of the room (see figure 9). The images of the room form reference frames which are stored by the SKIDS machine and constantly updated. An *event* within the room is detected by differencing the current camera image and the reference frame. The resulting differenced image is then thresholded and grouped into regions corresponding to the moving objects in the room. From the least enclosing rectangle, which is computed around each region, the position of the event can be determined by projecting the bottom of the rectangle onto the floor. The image processing required to perform the segmentation is computationally very demanding. A parallel processor and a specialised image processing engine are therefore employed within the SKIDS system to provide sufficient computational power. A Datacube pipeline image processor is used to acquire and scale images from the CCD cameras and a Transputer array provides the parallel processing support required for the remaining image processing tasks. In this way the SKIDS machine supports the real time detection, positioning, and tracking of all events within the room (the system typically operates at a sampling rate of about 6Hz).

Apart from objects such as humans and other vehicles the environment also contains a mobile robot. It is intended that the robot should be under the control of the SKIDS machine. The function of the robot is to carry a sensor suite to "remote" parts of the environment, for example to perform a localised inspection task. However, the localisation of the robot from the SKIDS track data alone is not possible without first identifying which event corresponds to the vehicle.

The mobile robot used for the purposes of this investigation is the Robosoft Robuter. The Robuter has two drive wheels mounted at the back with two castors at the front. Optical shaft encoders are attached to the two drive wheels in order to monitor and constantly feedback the motion of the robot wheels. The Robuter can be controlled remotely from a Sun4 workstation via an RS232 radio link which sends movement and measurement commands to the on-board operating system. This operating system is based around a 68020 microprocessor and supports movement control and sampling of the odometry obtained from shaft encoders on the wheels.

This study describes how the "data fusion" between the event data produced by the SKIDS machine and the robot trajectory as given by its odometry can be carried out by a neural network to perform the positional and orientational independent identification of the SKIDS event that corresponds to the robot which can then be localised.

Network Implementation

Two successful network implementations have been produced. Both are based upon the MLP and use error-back propagation (Rumelhart et al. 1986). The first network consists of a large input layer. This, as can be seen from figure 10, comprises inputs from each SKIDS track together with an equivalent signature obtained from the robot odometry. This investigation tried several different signatures based upon spatial or angular decompositions of the SKIDS event and robot velocities. The best performance was obtained from the following two reference frame independent signatures:

- object speed and angular velocity,
- object speed and acceleration.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

The fact that these signatures are found to be the best is not so surprising. The velocity of the vehicle is a relative measurement which can be obtained directly from the vehicle odometry with out the problems induced by systematic errors that would effect positional measurements. The use of velocity, therefore, is positional independent. Furthermore, orientation independence can be obtained if scalar, rather than vector, measures such as speed are used.

The output of the network used n units where n represented the number of SKIDS tracks that the network was designed for. For the case shown in the figure 10 $n = 3$. This allowed a 1 from n coding to be used to encode the output. Here a high value at the n^{th} output indicates a robot signature match with the n^{th} SKIDS input. It has been shown by MacKay (MacKay 1987) among others that this output coding, provided the network has been trained in the correct manner, allows the values given at the output to be interpreted as a confidence of the n^{th} interpretation. Since in certain circumstances one or more SKIDS events are not distinguishable from the real event this coding allowed the network to give a result which reflects the level of confusion.

To prevent any bias being introduced during training the position of the robot track in the training data was randomised. After some experimentation the most suitable network configuration for identifying the robot from 3 SKIDS events was found to be:

- 8 input units,
- 8 hidden units,
- 3 output units.

Upon testing of the network a $\sim$ 90 % success rate on data different from that used to train it was found. Furthermore, although as the percentage success rate suggests in some cases the network was unable to identify to which track the robot odometry belonged, this was usually because the network was unable to label a track as coming from the robot with enough confidence, as reflected by the value given at the output of the network, for unambiguous recognition. Since the majority of such cases resulted from situations where the network indicated that there were two robot candidates, one of which was the correct solution, the level of the miss-classification was much smaller than suggested by the above result.

The disadvantage in using the large multi–input network that is described here is that the system does not have any inherent ability to scale. To change the system from differentiating between not three but four or five SKIDS tracks requires the network to be extended and completely retrained. Obviously this suggests that although the large network may give desirable results its lack of flexibility probably precludes its use in a real system.

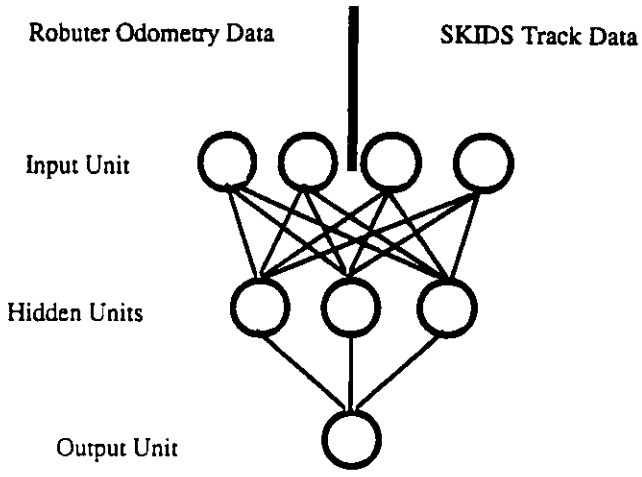


Figure 11: Single SKIDS Event Input Network

An alternative to the use of a single large network is to use a hybrid system of several small networks each trained to determine if a *single* SKIDS event matches the robot's odometry. This has the advantage that the individual networks that comprise such a system can be trained separately. Furthermore, if this training is carried out appropriately then it is only necessary to train a single network and allow the other networks in the hybrid system to be "carbon copies" of the first. Taking this idea a small MLP was trained using error back-propagation (Rumelhart et al. 1986) with the same velocity signature data that was found to be effective with the large multi-input network.

These small networks were simple in construction as is illustrated in figure 11. The input comprised 4 units which allowed the odometry signature of the robot and a *single* SKIDS track to be input to the network. The output, which consisted of just 1 unit, signified whether the SKIDS track given to the network belonged to the robot or not. The typical performance of the network with 3 hidden units was found to be marginly lower (85-90%) that obtained from the large multi-input network.

The hybrid design has many advantages. As has already been mentioned such a system scales in a much more sensible way than the large multi-input network (as the network increases in size the more small networks are used). If the networks were implemented in parallel, the computational burden imposed by this hybrid system increases approximately linearly with the number of networks and therefore SKIDS events. Furthermore, since it is possible that only one small network may have to be trained this greatly reduces the amount of training and therefore data required to configure the whole system.

A significant disadvantage of this system, however, is that when the individual small networks are trained, unlike the large multi-input networks, they are not aware of the presence of other events that may have

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

been detected. Not surprisingly therefore the output from the hybrid system may not be unique. If the input data is confused then several of the networks may match the robot odometry to the particular SKIDS event that they were given. This explains why slightly lower results for the hybrid system in comparison with the multi-input system were obtained. This problem can be overcome by introducing a "winner takes all" mechanism (Lippmann and Huang 1987) on the outputs of the networks.

An alternative to the "winner takes all" mechanism is to use the temporal continuity of the events generated by the tracker. This exploits the fact that there is a significant probability that the identity of an event will remain the same from one time frame to another. Obviously this probability is affected by the amount of noise in the system, and presence and number of other events in the room with which the event could become confused. This temporal continuity can be exploited by allowing lateral inhibition (Carpenter and Grossberg 1987, Kohonen 1984) between the outputs of the hybrid system. Here the weighted links between the outputs adapt with time such that an output that has had a high value for several time steps is enhanced whilst the others are diminished. This serves to dampen fluctuations in the output of the hybrid system such that in the event that an output is ambiguous the network still gives a definite answer.

Discussion

What has been demonstrated here is the use of a neural network, or networks, to perform the correlation central for the "data fusion" required for the identification of a known vehicle detected by a surveillance system. The localisation that results from this process is relative to the co-ordinate frame of the surveillance system which may be moving or static. Further, it has been demonstrated that this surveillance system can be distributed and so dispersed though-out the region of interest which gives a increase in the base line of the system allowing better positioning to be determined.

Although this system, like other identification methods, requires communication between the tracker and vehicle or vehicles, since the system is distributed, line of sight communication can be carried out to a variety of points, reducing the risk of interference or revealing the position of the vehicle.

This system also presents the alternative possibility where the surveillance system is distributed across the vehicles themselves to form a *distributed robotic system*. Such a system would consist of a large number of very simple mobile vehicles which are able to communicate with each other. The essential element of this distributed system is that, like a colony of ants, although the individual elements have a low degree of complexity the emergent behaviour of the whole system (if configured correctly) may be extremely complex. To allow the elements (vehicles) of this system to move together and therefore act as whole it is necessary for each vehicle know the relative position of the others. The positions of differing objects relative to a particular vehicle can be found via a simple passive or active tracking system. However, as has been shown, it is necessary to identify them first before a particular vehicle may be localised. This, as in the case study, can be carried out by matching the signature of the tracked objects with their odometry as communicated. The relative location of each identified vehicle can then be determined in relative to the whole group.

In the military field a robotic system such as this has several desirable qualities. The system is constructed of many simple, and hopefully, therefore cheap, disposable elements. Since the system does not depend upon any single element the system should be able to withstand a high level of attrition without a catastrophic effect upon the whole system's performance. A variety of possible applications come to mind from recognition and terrain mapping to the autonomous convoying of logistic support around a battle field.

Case Study 3: Integrated Localisation and Path Planning

The use of neural networks for the control of a mobile robot has already been graphically demonstrated by Waxman and his co-workers (Baloch and Waxman 1990, Waxman et al. 1988) together with, for example, the work carried out at the Fujitsu laboratories (Watanabe et al. 1989). In both cases a hierarchal architecture of neural networks have been designed to perform the differing functions required of the respective systems. In both these systems, however, a large proportion, if not all, of the neural processing is carried out by a static workstation communicating to the robot via a radio link.

The system built by Waxman and his co-workers for instance, MAVIN (Mobile Adaptive VIsual Navigation) (Baloch and Waxman 1990), uses a large hierarchy of networks to perform the processing required for the robot's cameras saccade and gaze control (simple ADALINEs (Widrow and Hoff 1960) are used here) through to object classification (ART I (Carpenter and Grossberg 1987) networks are used extensively here). All the networks used in this demonstration were simulated on a SUN 3/60 which communicated with the robot via a radio link. The image processing required was carried out on an ASPEX PIPE 1/800 video rate computer.

The network implementation for the Fujitsu robots is some what different from MAVIN. Here the networks that controlled the Fujitsu robots were trained and adapted off the robot on a workstation; the trained

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

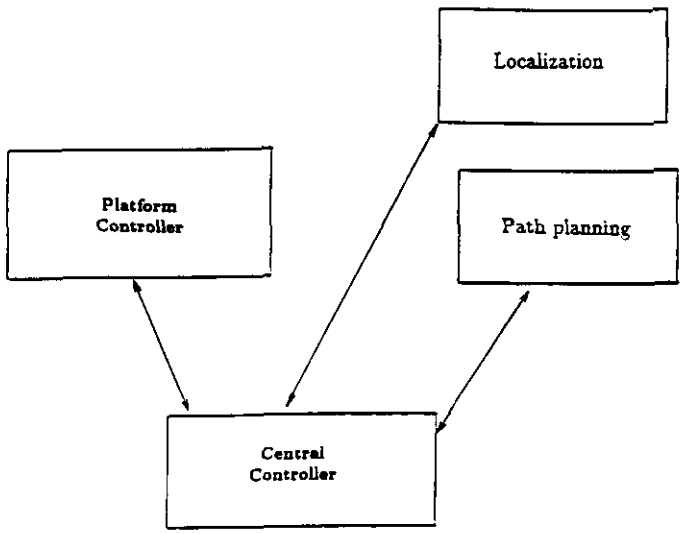


Figure 12: Control Architecture for the Oxford Robot

networks are then down loaded onto the robots' onboard processor. Although this allows the robot to be self contained the networks cannot be adapted on the robot itself.

It is the intention of this case study to highlight the potential further and substantial advances to be gained through the use of neural networks implemented on dedicated VLSI hardware. The work described is that still being undertaken by the Robotics Group at the University of Oxford under Dr. Lionel Tarassenko and in part[7] supported by RSRE[8] Malvern. The thrust of this work is to build a low-cost, real time mobile navigation system based upon a set of VLSI neural network navigational modules. These modules are based upon the two functional requirements that have been described earlier. This case study gives an overview of the path planning and localisation modules together with a description of how these two modules can be integrated together. Obviously the localisation module directly impinges upon the path planning module; a schematic diagram of the robot control architecture is given in figure 12. Both the path planning and localisation systems operate on a *certainty grid* idea which has been briefly described earlier.

Localisation

The localisation system on this robot relies on the certainty grid idea described above. Here a 28-point grid was used to map the robot's environment, see figure 13. In a similar way to that used by Jorgensen (Jorgensen 1987) the environmental characteristics were learned by recording the 360° signature obtained from a time-of-flight optical range finder. This

[7] The resistive grid path planning system

[8] Royal Signal and Radar Establishment

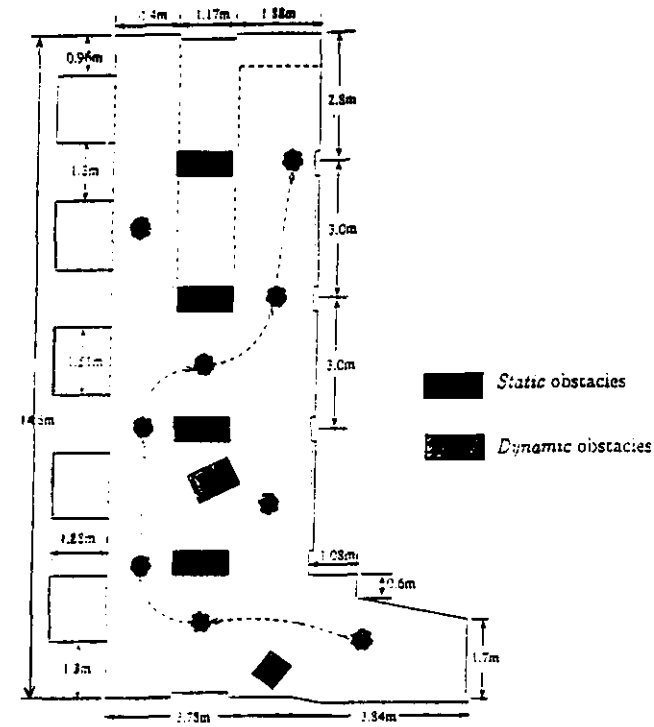


Figure 13: Diagrammatic view of the Oxford Laboratory Robot Environment

was a phase sensitive near infra-red device that was developed by the research group with this purpose in mind. This device is capable of resolving phase shifts of 0.1° over a 50 dB range. This, as can be seen from figure 14, allows a very detailed range map to be produced. The original work carried out by Oxford in this area concentrated on the use of an ultra-sonic sensor. This required extensive preprocessing before the signatures could be input to the network. The high resolution infra-red scanner mitigates this problem.

Given a set of learned signatures the grid system can be used to compute the robot's approximate position. This may be determined by comparing the current signature x with one of the k learned patterns u_i which correspond to the signature of the range finder at each of the k grid points. By finding the closest match between x and one of the u_i's the position of the nearest grid point to the present position of the robot can be obtained. If a Euclidean metric is used to determine the difference between x and all u_i's then the closest match may be obtained for that u_i were:

$$\|x - u_i\|^2 = \|x\|^2 - 2u_i^T x + \|u_i\|^2 \qquad (1)$$

is a minimum. Given that x is constant with respect to i using equation 1 a linear discriminant function $g(x)$ can be written were:

$$g_i(x) = u_i^T x + \omega_{i0}, \qquad (2)$$

and $\omega_{i0} = -1/2\|u\|_i^2$. The discriminant function thus uses the *cross correlation* of the input with the stored

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

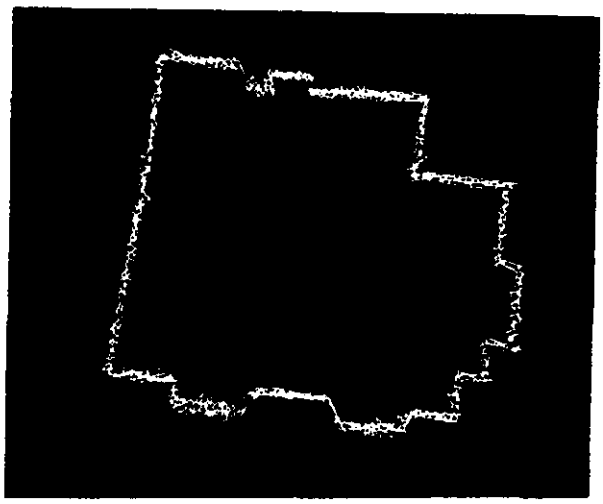


Figure 14: Range map of the Oxford environment

patterns, the maximum value of which gives the pattern u with which the signature x most closely correlates.

If equation 2 is rewritten by identifying $u_i = \{T_{ij}\}$ and $x = \{V_j\}$ as:

$$g_i(x) = \Sigma_{j=1}^{n} T_{ij} V_j + \omega_{i0}, \qquad (3)$$

where n is the number of range points obtained in each scan, the patterns recorded at the grid points can be identified with neural weights Tij. The cross correlation central to the discriminant function can be written as the vector matrix multiplication, $\Sigma_j^n T_{ij} V_j$, that is central to a neural network.

The advantage of the formulation given in equation 3 is that it provides a natural representation that when implemented on a dedicated neural device allows the simultaneous comparison of all range points with the k learned patterns u_i. The maximum of the discriminant function $g(x)$ can then be picked out using by using a "winner takes all" function on the network.

The advantage of such a system of course depends upon it implementation. As has previously been mentioned, for any implementation to be of advantage both its speed and size are important characteristics. The *localisation algorithm* that has been described here can be built quite simply into a small "winner takes all" network. Since both the input vectors V_j and the network weights T_{ij} are analogue this allows the implementation to be mapped easily into the pulse-stream VLSI analogue neural devices that have been designed by the Department of Electrical Engineering at the University of Edinburgh in conjunction with the Robotics Group (Murray et al. 1988, Murray et al. 1990) which provide real time capability. The speed of the localisation system is simply limited by the traverse time of the infra-red scanner which is approximately a second. Further, the compact size and analogue nature of the pulse stream device allows the processing to take place compactly *on* the sensor where as a more conventional implementation say with Transputers would require many more devices with a larger resultant demand for power.

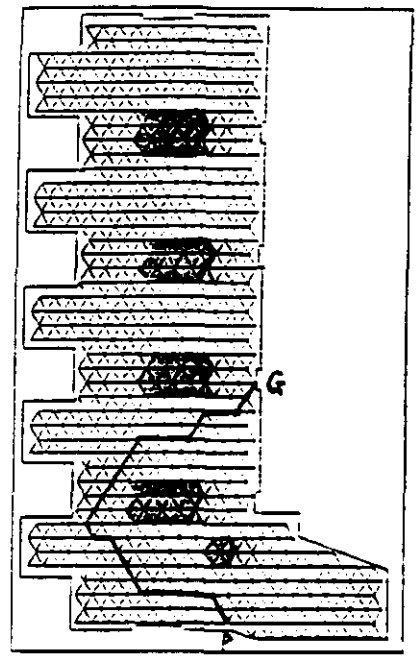


Figure 15: Resistive grid map of the robot's environment; high resistances (black areas) indicate obstacles. The optimal path between P and G is indicated by the black line joining these points.

Path Planning

The path planning module in the Oxford robot adopts a resistive grid approach to this problem. The use of resistive grids was suggested in a related field by Horn (Horn 1974) in the mid seventies. The idea has also been exploited by Mead and his co-workers and forms the central element of the silicon retina (Mead and Mahowald 1988). This approach maps the robot's environment as a resistive grid, see figure 15. Here the vertices of the grid are variable resistors: obstacles and difficult terrain are indicated by infinite or high resistances. This provides a map of the terrain in terms of high and low resistances, the valleys and peaks indicating the easy and difficult (accessible) regions of the environment. An optimal path can be obtained through the environment by simply applying a potential difference between the the robot position (P) and its desired destination (G) and following the path of maximum current. Since the current cannot flow through regions with an infinite resistance (obstacles) and will be reduced in regions of high resistance (difficult terrain) following such a current path will *guarantee* an obstacle free path.

Although this method has been tried before (Mitchell and Keirsey 1984) it is the intention of this

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

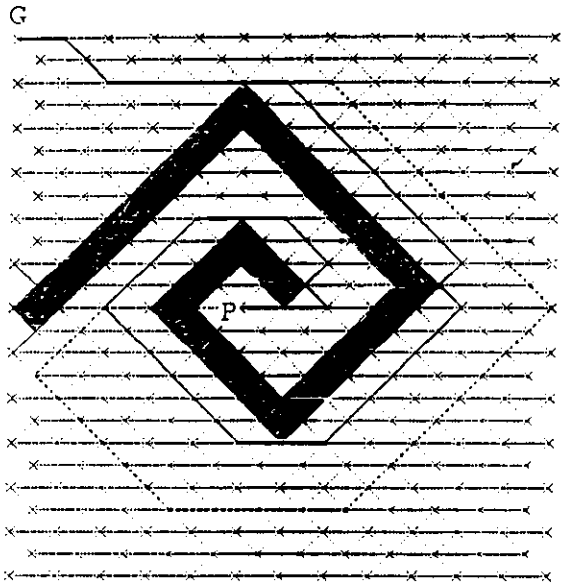


Figure 16: Path from middle of maze (P) to top left corner (G). The dotted line represents that obtained by repeated calculation of the path direction. The solid line is the complete path obtained by a single calculation of the path a point P

implementation to map the resistive grid directly into a VLSI device using an array of MOS switches. Here the grid vertices can adopt one of two states: an infinite resistance if the switch is open, and zero resistance with the switch closed. The map of the environment, therefore, consists of a zero resistance surface with regions of infinite resistance representing the obstacles. Allowing the resistance grid to have a 1:1 mapping with the localisation certainty grid enables the current position of the robot in the resistive map to be easily updated as the vehicle moves through the environment.

The optimal path through the environment is found by applying a potential difference between the robot's position and the desired destination and then determining the path of maximum current. This is indicated by the node on the *hexagonal* grid that has the largest potential difference between itself and the robot's node.

Having moved to the new node the process can then be recalculated and an updated path found in real time. It has been shown that by continualy recalculating the direction of motion after each step a better (i.e. shorter) path can be obtained than by simply calculating the complete path across the grid in one go (Tarassenko and Blake 1991). This is illustrated in figure 16. Here the paths out of the maze from point P to point G, have been calculated using:

- repeated computation of the path direction, dotted line;
- single computation of the path from point P, solid line.

Since this calculation is carried out on chip (this is essentially a hardware computation of Kirchhoff's equation) the calculation can take place in the time it takes the MOS grid to settle once the voltage is applied. Furthermore, since the grid map can be altered by simply reconfiguring the MOS switches from data down-loaded from RAM this implementation provides a real time reconfigurable map that can be updated as soon as new obstacles are detected or the position of the robot is determined.

Control Architecture

The control architecture for this robot reflects the structure of the path planning and localisation systems and has been designed in a modular fashion (see figure 12). Communication between the differing modules takes place asynchronously via a conventional *central controller* which routes the appropriate control signals to and from the modules. The *central controller* is also responsible for goal specification and issuing commands to the robot *platform controller* which, for the purposes of this design, is again conventional. Since the intention of this control architecture is to allow the bulk of the control processing to take place locally within the localisation and path planning modules, the *central controller* is very simple in construction.

Discussion

At the time of writing a small mobile robot has been constructed and the localisation system implemented on dedicated VLSI neural devices. A separate implementation of the path planning system together with the localisation system has also been undertaken. Since the path planning system has not yet been implemented on an appropriate device the integrated localisation/path planning has been carried out on a SUN 4 which communicated with the robot via a radio link. To allow the path planner to operate on the SUN 4 in near real time dynamic reconfigurablity was not used.

This system has been demonstrated on a small battery powered Turtle that has been modified to carry the infra-red scanner. Both the scanner and the robot controller, which is based upon a 68000 processor, are powered by the robots battery. This implementation allowed the robot to move through a static laboratory environment (i.e. no moving obstacles) with the position of the robot and its direction of motion being updated in real time at an approximate speed of 0.4 $ms^{-}1$. This performance is limited by the traverse speed of the scanner and the band width of the radio link.

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Although no obstacle avoidance function has yet been integrated in the control architecture, this is planned for the (near) future. It is expected that this function will be implemented in a similar fashion to the work described in the first case study, using a sensor/motor association network. However, in this case it is proposed that a number of fixed optical sensors are used rather than ultra-sonic.

Although the networks described here have not all been fully implemented in special hardware, this case study has illustrated the advantages to be gained from the use of dedicated hardware in terms of speed, ease of integration, and particularly size. With the future advent of larger, faster and more complex neural devices it could be argued the the full potential of these systems has still to be realised. It could further be argued that it is a only matter of time before devices similar to those described here are produced and used in real production systems.

Conclusion

The case studies presented in this paper have tried to outline potential areas where mobile robotics will benefit from the use of neural networks. To do this studies have been chosen which, rather than describing work already completed and available in the scientific press, portray some of the typical research that is currently being undertaken. Since much of this research is still in progress some of the results are inevitably not complete.

The first two studies demonstrate how adaptable non-linear systems can be used for the processing required for functions from obstacle avoidance through to their possible use for the "data fusion" required to localise a vehicle detected by a distributed surveillance system. Both studies are relevant to a number of the fundamental functional requirements for any mobile robotic system. The third case study illustrates in part how the use of these systems can be implemented in dedicated silicon. As both digital and analogue neural VLSI devices are developed, it is expected that neural networks will provide cheaper, faster, and more compact alternatives to conventional hardware. (Holler et al. 1989, LeCun et al. 1990) This is likely to be of direct relevance to military requirements where systems necessarily need to be adaptable, and space on any vehicle is likely to be at premium.

In the case studies large volumes of data were required to train the networks appropriately. This all too often presents a problem in that although databases exist these are usually too small to provide sufficient data. Two solutions have been suggested to this problem. The first is to use real on-line data by integrating the networks directly into the system in which it is supposed to operate. This has the obvious advantage of enabling an accurate estimate of the neural network's performance, on real data, to be obtained.

The second alternative is to adopt the solution demonstrated in the first case study: the data is simulated. If the simulation is designed with care this can quite often provide a good alternative. Furthermore, strict controls can also be placed upon the data allowing the performance to be tested easily. However, by its very nature a simulation can never truly represent *real* data with all its anomalies and inaccuracies. An analysis of how a network trained on simulated data would behave once placed in the real world would therefore be uncertain and difficult to verify without, as was undertaken with ALVINN (Pomerleau 1988), eventually testing the networks on real data

A further alternative is to use real data but to gather this into a large data base, such as a library of images for instance. This alternative has the advantage of providing a repeatable set of real data which can, when required for experimental reasons, be properly controlled. However, the work involved in gathering such a database can be very large and particular consideration has to be taken to ensure no bias is introduced into it during the production stage. This quite often makes the production of such a data base surprisingly expensive and therefore not desirable.

Another major problem for the future use of neural networks for the sensor processing and control on a mobile robot is the *verification* of these systems. In both civil and military applications, for any safety critical operations, it is necessary for the behaviour of the systems used not only to be understood but to be designed with the appropriate safe guards to prevent undesirable responses. An appropriate certification procedure would also be required.

Until recently, with the exception of the large body of work that exists for some of the unsupervised neural networks, there has been very little effort in this area. However, with the use of more *mathematically* structured neural networks, such as the radial basis function networks (Broomhead and Lowe 1988), verification has started to become a possibility. Furthermore, with the more recent interest of the control research community, the problem of certifying such systems has started to be addressed (Simper 1991). Although there are no procedures laid down as yet to ensure the verification of the design, configuration (training), and testing of a neural network it has been suggested that principles similar to those use for the verification of a mathematical process be used. A thorough understanding of the problem that the system is to be designed to solve is required, something which is generally necessary when trying to design a network solution for a problem in any case. This can be difficult since many problems to which neural network are being applied are highly non-linear and therefore may not be easily tractable. With an appropriate understanding it is suggested (Simper 1991) that sufficient safe guards could be put in place (e.g. an expert system harness) to check against undesirable inputs being

This Document Provided by the ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

presented to the network or outputs from the network having an undesirable effect.

It is accepted that the robotic systems that have been described in the case studies are somewhat simple compared to the all terrain robotic systems that are required for the military environment. This reflects the fact that the use of neural networks in the area of mobile robotics is still limited. However, the case studies that have been presented have demonstrated that neural networks offer potential solutions to some of the problems that are generic to the whole field of mobile robotics, and, if implemented in dedicated VLSI silicon, will hopefully have a direct bearing on the future construction of such vehicles where *fast*, *compact*, and *adaptable*, systems are required. As these devices become available the true nature of the advantages to be obtained from the use of neural networks should become apparent over the next few years.

Acknowledgements

The author would like to thank Dr.L.Tarassenko of Oxford University, Dr.D.Lowe of RSRE, Mr R. Opitz of IBP Pietzsch, the ANNIE project management, and Professor C. Harris of Southampton University for their co-operation and kind permission to use their material that has made this article possible. Furthermore the author would also like to thank Dr P. Greenway without whose indefatigable effort in reading the manuscript this article would not be possible, and Mr G. Lamont for the preporation of some of the figures.

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
abbottaerospace.com

Appendix A: Mobile Robot Initiatives

Commercial Research & Development

- ESPRIT II Panorama (including BAe, SAGEM (France), Rauma-Repola (Finland), Tamrock (Finland), University of Helsinki, Universidad Politecnica de Madrid, Easams (Frimley), Southampton University, Central Energy Atomique (Grenoble & Saclay, France), SEPA (FIAT, Italy), EID (Portugal), LNETI (Portugal), CRIF (Belgium)).

  Target vehicles are 4x4 Mercedes Jeep, Rauma-Repola Forwarder (FMG 933C Lokomo), Tamrock Driller. Five year project ending March 1994.

- ESPRIT I Voila (including: GEC, Plessy ELSAG, MS2i, RMR, Oxford Univestity, Sheffield University, INRIA, University of Genoa):- production of a vision guided mobile robot.

- MARDI, (including BAe, UK MOD, Royal Armaments Research and Development Establishment (RARDE), Southampton University, Bristol University, Lucas):- production of an all terrain military robot.

- GEC/Oxford University 'Turtle' project.

- Advanced Robotics Research Centre, Salford:-, UK national center for robotics.

- PROMETHEUS. Companies involved include Jaguar, Lucas, Pilkington, BMW, Porsche, Volkswagen, SAAB and Volvo, with PSA (a French consortium consisting of Peugeot, Citroen and Talbot). Academic involvement is with the University of Southampton and University of Oxford.

- IVHS - Intelligent Vehicle Highway System. This is a US Department of Transport project, which commenced in 1989 and is heavily funded with 181 approved projects so far.

- Daimler-Benz AG, Stuttgart, Germany - automated guidance system.

- Mazda, Japan - three autonomous vehicle testbeds.

- Nissan Motor Company, Yokosuka, Japan.

  Fuzzy logic steering control of an autonomous vehicle

- Volkswagen, Germany

  Self Parking vehicles research

- Toyota, Japan

- SENTRY - Denning Mobile Robotics Inc., Woburn, Mass., USA.

- EUREKA - AMR (Advanced Mobile Robot).

- EUREKA - MITHRA.

- Autonomous Land Vehicle, Martin Marietta Corp., Denver, USA.

  DARPA funded project, vehicle intended mainly for military purposes.

- Fujitsu Lab, Ltd.,Kawasaki, Japan.

  Image processing for autonomous vehicles.

- Mech. Eng. Lab, AIST, MITI, Ibaraki, Japan.

  Steering control for an autonomous vehicle.

- Tokyo Research Lab, IBM Japan, Japan. - visual navigation of autonomous vehicles.

- Shinko Electric Co. Ltd, Hyogo, Japan. - ultrasonics guided autonomous vehicles.

- Naval Ocean Systems Centre, San Diego, USA. - ground surveillance robot.

- Sandia National Labs, Albuquerque, N. Mexico, USA. - fleet of vehicles for remote control and autonomous operation.

- Savannah River Lab., Aiken, South Carolina, USA. - autonomous vehicles for nuclear applications.

- Jet Propulsion Lab, Pasadena, California, USA. - primarily work for the Mars Rover vehicle.

- Tokyo Institute of Technology, Japan - primarily walking vehicles but with spin-off applications including control technology.

- FMC Corporation, Central Engineering Laboratories, Artificial Engineering Centre, Santa Clara, California, USA. - multi-goal, real-time global path planning for an autonomous land vehicle.

- Army Engineer Topographic Labs, Fort Belvoir, Virginia, USA. - robotic reconnaissance vehicle with terrain analysis.

Academic Research

- Carnegie-Mellon University, Pittsburgh, USA. Chuck Thorpe, - Navlab/Alvan and Terregator.

- Oxford University, Engineering Department, Prof. Mike Brady.

- MIT, USA, R. Brooks & A. Waxman.

- LAAS, France, Raja Chatila - HILARE.

- Heriot-Watt University, Edinburgh, Intelligent Automation Lab., Chantler, M.J. et al.

- Southampton University, Department of Aeronautics and Astronautics, Prof. Chris Harris.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

- Tech. Univ. Munich, Germany, Lehrstuhl fur Mikrowellentech - imaging radar for autonomous vehicles.
- Massachusetts University, Amherst, Dept. Computer and Information Science - Autonomous Vehicle Navigation Project.
- Oakland University, USA, Centre for Robotics and Advanced Automation - Autonomous vehicle project.
- Univ. der Bundeswehr Munchen, Neubiberg, Germany, Inst. fur Messtech, Prof. Dickman.
- Ohio State University, Columbus, USA.
- University of Maryland, Center for Automation Research, Maryland, USA. - computer vision systems for Martin Marietta autonomous vehicle.

References

Amit, D.J., Gutfreund, H., and Sompolinsky, H., (1985). *Spin-glass models of neural networks.* Physical Review, **A32.**

Baloch, A.A. and Waxman, A.M., (1990). *A neural system for behavioral conditioning of mobile robots.* In **Proceedings of the International Conference on Neural Networks**, San Diego.

Bateman, P.J., (1991). *The UK mobile advanced robotics defence initiative (MARDI).* In **Proceedings of the NATO DRG Seminar on Battle Field Robotics.**

Broomhead, D.S. and Lowe, D., (1988). *Multivariable function interpolation and adaptive networks.* Complex Systems, **2**, 269–303.

Brown, M., Fraser, R., Harris, C., and Moore, C.G., (1991). *Intelligent self-organising controllers for autonomous guided vehicles comparative aspects of fuzzy logic and neural nets.* In **the proceedings of Control 91**, Volume 1, pages 134–137. IEE.

Buxton, B.F. and Roberts, M.T., (1990). *VOILA vision research pilot project.* In **ESPRIT Information Processing Systems: results and prograss of selected projects.** Commission of the European Communities.

Carpenter, G.A. and Grossberg, S., (1987). *A massively parallel architecture for self-organizing neural pattern recognition machine.* Computer Vision, Graphics, and Image Processing, **38**, 54–115.

Chen, V.C. and Pao, Y.H., (1989). In **Proceedingds IEEE conference on Robotics and Automation**, Volume 1, pages 1448–1453.

Dickmanns, E.D., (1990). *Dynamic vision for inteligent motion control.* In **Proceedings of IEEE International Workshop on Inteligent Motion Control.**

Dickmanns, E.D. and Graefe, V., (1988). Machine Vision and Applications.

Elfes, A., (1987). *Sonar–based real–world mapping and navigation.* IEEE Journal of Robotics and Automation, **RA–3**, 249–265.

Fukushima, K. and Miyake, S, (1982). *Neocognition: A new algorithm for pattern recognition tolerant of deformations and shifts in position.* Pattern Recognition.

Holler, M., Tam, S., Castro, H., and Benson, R., (1989). *An electrically trainable artificial neural network with 10,240 floating gate synapses.* In **Proceedings of the International Joint Conference on Neural Networks.**

Hopfield, J.J., (1982). *Neural networks and physical systems with emergent collective computational abilities.* Proceedings of the Academy of Science USA, **79**, 2554–2558.

Horn, B.K.P., (1974). *Determining lightness from an image.* Computational Graphics and Image Processing, **3**, 277–299.

Hutchinson, R.A., (1990). *Development of an MLP feature location technique using preprocessed images.* In **Proceedings of International Neural Network Conference.**

Ince, D.L., Bialasiewicz, J.T., and Wall, E.T., (1991). *Neural network-based model reference adaptive control system.* In **the proceedings of the Second Workshop on Neural Networks WNN-AIND 91**, Volume 1, pages 543–551, Alabama.

Jamison, T.A. and Schalkoff, R.J., (1988). *Image labeling: A neural network approach.* Image and Vision Computing, 203–214.

Jorgensen, C.C., (June 1987). *Neural network representation of sensor graphs in autonomous robot path planning.* In **the proceedings of IEEE First International Conference on Neural Networks**, pages IV–507–IV515, San Diego.

Kawato, M., Furukawa, K., and Susuti, R., (1987). Biological Cybernetics, **57**, 422–428.

Khatib, O., (1986). *Real–time obstacle avoidance for manipulators and mobile robots.* International Journal of Robotics Research.

This Document Provided by the ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Klein, C.A., Kau, C-C., Ribble, E.A., and Patterson, M.R., (1987). *Vision processing and foothold selection for the ASV walking machine.* In **Proceedings of the SPIE conference on Mobile Robots II.**

Kohonen, T., (1984). **Self-organisation and associative memory.** Springer-Verlag.

LeCun, Y., Jackel, L.D., Graf, H.P., Boser, B., Denker, J.S., Guyon, I., Henderson, D., Howard, R.E., Hubbard, W., and Solla, S.A., (1990). *Optical charicter recognition and neural-net chips.* In **Proceedings of International Neural Network Conference.**

Lippmann, R.P. and Huang, W.Y., (November 1987). *A framework for speech recognition using a neural network.* In **Proceedings of the IEEE Conference on Neural Information Processing Systems, Natural and Synthetic**, Denver.

MacKay, D.J., (1987). **A method of increasing the contextual input to adaptive pattern recognition systems.** Technical Report RIPRREP/1000/14/87, Research Initiative in Pattern Recognition.

Mead, C.A. and Mahowald, M.A., (1988). *A silicon model of early visual processing.* Neural Networks, 1, 91–97.

Mitchell, J.S.B. and Keirsey, D.M., (1984). *Planning strategic paths through variable terrain data.* In **Proceedings of SPIE Applications of Artificial Intelligence.**

Moravec, H.P., (1986). **Certainty grids for mobile robots.** Technical report, Carnegie-Mellon University.

Murray, A.F., Hamilton, A., and Tarassenko, L., (November 1988). *Fully-programmable analogue VLSI devices for the implementation of neural networks.* In **IEEE Conference on Neural Information Processing Systems: Natural and Synthetic**, pages 671–677, Denver.

Murray, A.F., Hamilton, H., Reekie, H.M., Churcher, S., Baxter, D.J., Butler, Z.F., and Tarassenko, L., (1990). *Innovations in pulse-stream neural VLSI arithmetic and communications.* In **the proceedings of IEEE Workshop on Microelectronics for Neural Networks**, Dortmund.

Peterson, J., (1991). *Obstacle avoidance using neural networks and hierarchical dynamic programming.* In **the proceedings of the Second Workshop on Neural Networks WNN-AIND 91,** pages 553–560, Alabama.

Pomerleau, D.A., (December 1988). *ALVINN: An autonomous land vehicle in a neural network.* In **the proceedings of the IEEE Conference on Neural Information Processing Systems: Natural and Synthetic**, pages 305–313, Denver.

Rumelhart, D.E., Hinton, G.E., and Williams, R.J., (October 1986). *Learning representations by back propagation of errors.* Nature, **323**, 533–536.

Savage, J.T., (1991). *ROVA–an autonomous road vehicle.* In **Proceedings of the NATO DRG Seminar on Battle Fields Robotics.**

Simper, A.M., (1991). *Quality assurance and validation of neural networks.* In **Sira communications on Neural Networks in Process Control.** Sira.

Simpson, (1987). *Image understanding and vision research at darpa.* In **Proceedings of SPIE conference on Intelligent Robots and Computer Vision.**

Spiessbach, A., Clark, B., Larimer, S., Tobey, B., Lindauer, B., Koenig, R., and Lisec, T., (1987). *Issues and options for a Mars rover.* In **Proceedings of the SPIE conference on Mobile Robots II.**

Tarassenko, L. and Blake, A., (1991). *Analogue computation of collision-free paths.* In **Proceedings of 1991 IEEE International Conference on Robotics and Automation.**

Tarassenko, L., Brownlow, M., Marshall, G., Tombs, J., and Murray, A., (1991). *Real-time autonomous robot navigation using VLSI neural networks.* In **Advances in Neural Information Processing Systems**, pages 422–428. Morgan Kaufmann.

Thorpe, C., (1984). *Path relaxation: Path planning for a mobile robot.* In **Proceedings of the AAAI,** pages 318–321. AAAI.

Thorpe, C., Herbert, M., Kanade, T., and Shafer, S., (1987). **Vision and navigation for the Carnegie Mellon NAVLAB.** Volume 2 of *Annual Review of Computer Science.* Annual Reviews Inc.

Touretzky, D.S. and Pomerlau, D.A., (August 1989). *What's hidden in the hidden layers?* Byte, 227–233.

Vacherand, F., Crochon, E., Faure-Reguillon, F., Rogaert, M., Do, S., Campos, M. De., and Tavora,

This Document Provided by the ABBOTT AEROSPACE TECHNICAL LIBRARY ABBOTTAEROSPACE.COM

J., (1990). *PANORARMA local navigation for a class of outdoor robots.* In **ESPRIT Computer Integrated Manufacture: results and prograss of selected projects.** Commission of the European Communities.

Watanabe, N., Nagata, S., and Asakawa, K., (1989). *Mobile robot control by neural networks and their associated learning algorithm.* SPIE proceedings, **963**, 570.

Waxman, A.M., Wong, W.L., Goldenberg, R., Bayle, S., and Baloch, A., (1988). *Robotic eye-head-neck motions and visual-navigation reflex learning using adaptive linear neurons.* Neural Networks, **1**, 365.

White, D. and Sofge, D., (1991). *NSF workshop on aerospace applications of neurocontrol.* In the **proceedings of the Second Workshop on Neural Networks WNN-AIND 91**, pages 523–527, Alabama.

Widrow, B., (1990). *The truck backer up.* In **Proceedings of Parallel Processing in Neural Systems and Computers.**

Widrow, B. and Hoff, M.E., (1960). IRE Wescon.

Wilcox, B.H., Gennery, D.B., Mishkin, A.H., Cooper, B.K., Lawton, T.B., Lay, N.K., and Katzmann, S.P., (1987). *A vision system for a Mars rover.* In **Proceedings of the SPIE conference on Mobile Robots II.**

Wolfe, W.J. and Chun, W.H., (Eds.), (1987). **Proceedings of Mobile Robots II**, chapter Session 4. SIPE.

Wright, W.A., (October 1989). *Image labelling with a neural network.* In **Fifth Alvey Vision Conference**, Reading.

This work has been carried out with the support of the Procurement Executive, Ministry of Defence.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-179

MULTISENSOR DATA FUSION AS APPLIED TO GUIDANCE AND CONTROL

by
Patrick K. Simpson
General Dynamics Electronics Division
P.O. Box 85310, MZ 7202-K
San Diego, CA 92138

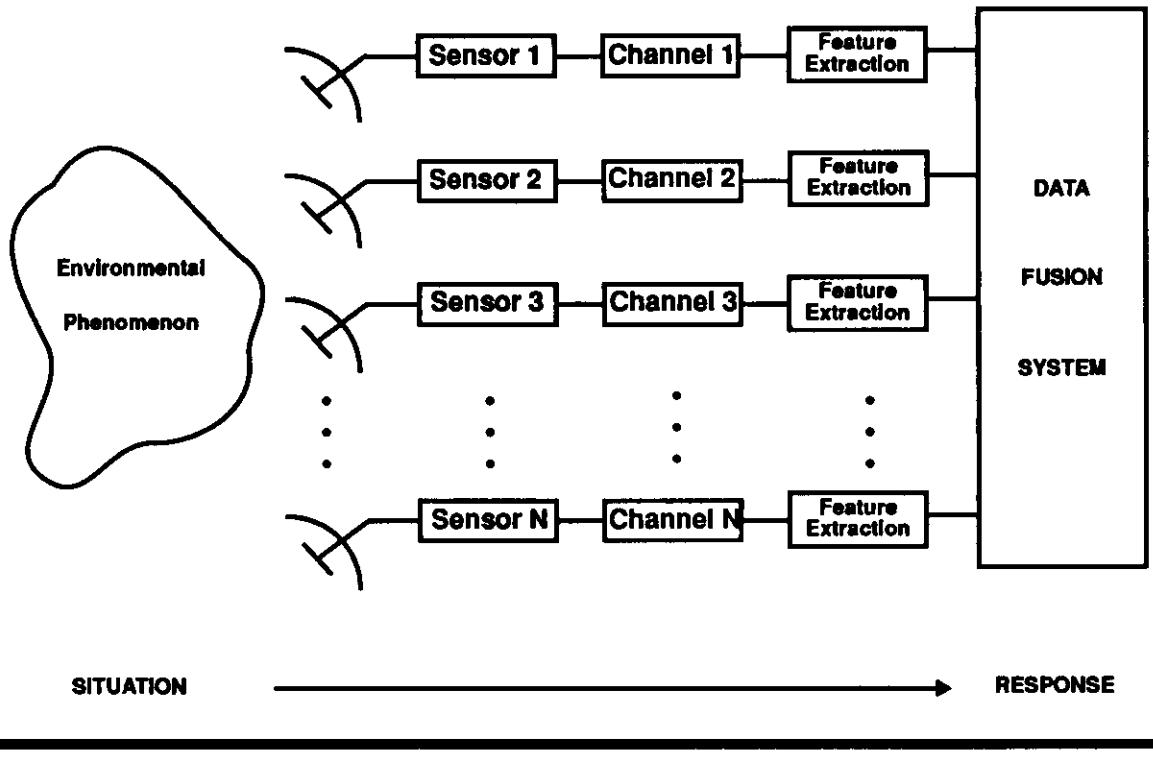
SUMMARY

Multisensor data fusion (MDF) is the synergistic application of data from several sources, typically sensors, toward a specific task. In the area of guidance and control data fusion plays a very important role. By combining the information from several sensors it is possible to improve the performance of guidance and control systems. Neural networks are ideally suited to applications where only a few decisions are required from a massive amount of data. In this sense, neural networks should play a crucial role in future data fusion systems. This paper will describe several methods of applying neural networks to data fusion, including: self-organizing hierarchical neural systems, multi-layer error correction learning networks, and single layer pattern completion systems. Application case studies will be examined to determine how researchers have applied neural networks to data fusion. In addition, a discussion of feature representation and feature weighting will be provided.

1. INTRODUCTION

Multisensor data fusion remains one of the most challenging research areas in the compu-

Figure 1: Multi-sensor Data Fusion System



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

tational sciences. Multisensor data fusion is the process of combining the data from several distributed sensors (potentially thousands) and making a decision. The sensors can vary widely in reliability, the type of data being received can vary, and the resulting decision might be required in real-time. The applicability of neural networks to this environment represents a natural synergism between the inherent capabilities of neural networks and the computational demands of the multisensor data fusion (MDF) problem. This paper reviews current MDF techniques, describes three neural network approaches to MDF, and presents some potential MDF applications in guidance and control.

2. OVERVIEW OF DATA FUSION

Data fusion is the synergistic combination of data from several sources into a coherent decision. When the data is supplied solely from sensors the result is a multisensor data fusion system. Figure 1 illustrates a general multisensor data fusion (MDF) system. In general, a MDF system can be viewed as a situation-response system. Some phenomenon occurs in the environment that is observed by a set of N sensors. Each sensor collects information and transmits it across a channel where features are abstracted from the sensor data. The entire set of features recorded during a given interval of time represents the situation. The set of features produced from each sensor is subject to different levels of noise, different time-delays for information propagation, and different relative importance weightings between sensors. The situation data is eventually fed to a data fusion system where a response must be provided from the data. It is immediately evident why data fusion is so difficult.

2.1. Advantages of Multisensor Data Fusion

The primary motivation for data fusion is the realization that single sensor information is often not enough. The synergistic collection of information from a wide variety of sensors is required to produce reliable responses. Although this is the primary reason for data fusion, there are others. Luo & Kay (1989) have identified four advantages of MDF systems:

Redundancy: By receiving sensor information from several similar sensors it is possible to attain improved accuracy. In systems that utilize redundant sensors the fusion is performed at a low level.

Complementary: By receiving sensor information from different sensors, it is possible to create a more robust representation of the phenomenon being sensed. In systems that utilize complementary sensors the fusion is performed at a high level.

Timeliness: By distributing the sensing task to several sensors it is possible to produce faster decisions. Single sensor systems often need to repeatedly sample prior to emitting an accurate decision. Multisensor systems take advantage of the redundancy to achieve the desired accuracy.

Cost: Depending on the system, it is possible to provide a multisensor system at less cost than a single sensor system.

2.2. Multisensor Data Fusion Paradigms

A MDF system requires several capabilities. It must be able to incorporate and arbitrate data from a large number of sources. It should allow the relative weighting of sources to be done easily. And, it should provide timely responses. Luo and Kay (1989) have outlined four primary paradigms that meet all of these requirements:

Hierarchical Phase-Template Systems: A general paradigm for robotic systems based upon four temporal phase of sensor-to-object distance (far- away, near-to, touching, and manipulation).

Logical Sensors: Abstracting each sensor from a physical device to a logical entity allows

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

collections of sensors to represented very elegantly. This approach is useful in applications that require a world model to operate in harmony with the sensor system (eg. robotics).

Object Oriented Programming: Each sensor in the MDF system is represented as a data object. An object contains both data and functions and it communicates to other objects via messages. Like the Logical Sensors approach, this has a very appealing general structure that is amenable to several symbolic-based data fusion tasks.

Neural Networks: Create patterns from the various sensors (via preprocessing) and process the multiple patterns using a neural network. This technique will remain the focus of the remainder of this paper.

2.3. Examples of Multisensor Data Fusion Systems

The most incredible MDF systems are mammals, especially humans. The ability to fuse auditory, visual, olfactory, and tactile information is unparalleled. Recent work by Singer and others (Barinaga, 1990) has exposed some clues about how humans are able to perform sensor fusion. Information in disparate regions of the brain has been found to phase-lock and operate synchronously. This research is revealing a new approach to neural systems where information is stored in oscillations of different frequencies. Relative to mammals most MDF systems pale, but the full capability of a human is not necessary to provide improved performance for most applications. Recent examples of highly capable MDF systems include robots, surveillance systems, and target tracking systems.

2.3. Multisensor Data Fusion Surveys

This paper will review the neural network aspect of MDF with an emphasis on guidance and control. There are several resources that discuss other aspects of MDF. Maren & Pereira (1989) have conducted an extensive survey of multisensor information fusion that analyzes sensor selection, levels of abstraction, architectures, and methodologies for fusion. Luo and Kay (1989) have also conducted an extensive review of multisensor integration and fusion with an emphasis on robotics applications. Mitchie & Aggarwal (1986) have performed a survey of multisensor integration with an emphasis on image processing applications and Garvey (1987) has analyzed the Artificial Intelligence approaches to multisensor information fusion.

3. NEURAL NETWORK DATA FUSION

There are three primary methods for neural network data fusion: (1) pattern completion, (2) pattern matching, and (3) hierarchical systems. Each neural network fusion technique has its own merits and an affinity for different application areas. In the following sections each of these techniques will be examined with specific applications cited with each technique.

3.1. Pattern Completion Neural Fusion

The pattern completion technique for neural network MDF is illustrated in Figure 2. All of the sensor data types are concatenated together into a large vector with the desired response. As an example, Anderson, et al. (1990) used this representation for the classification of radar emitters. In this instance, the data types where pulse repetition interval, operating frequency, and so on, and the corresponding output was the name of the radar system.

Pattern completion neural fusion fits within a situation-response framework very well. Applications that might use this fusion technique might include target recognition, signal classification, and control applications. Target recognition might utilize infrared, optical, radar and acoustic data to describe the situation and correlates this information with the classification of the target as a response. Signal classification can utilize Fourier spectra, duration

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

of signal, and total signal power as the situation and produce a classification of the signal as the response. Control applications can collect sensor data from the platform being controlled and merge this with infrared information to create the situation and the response would be the next action to take.

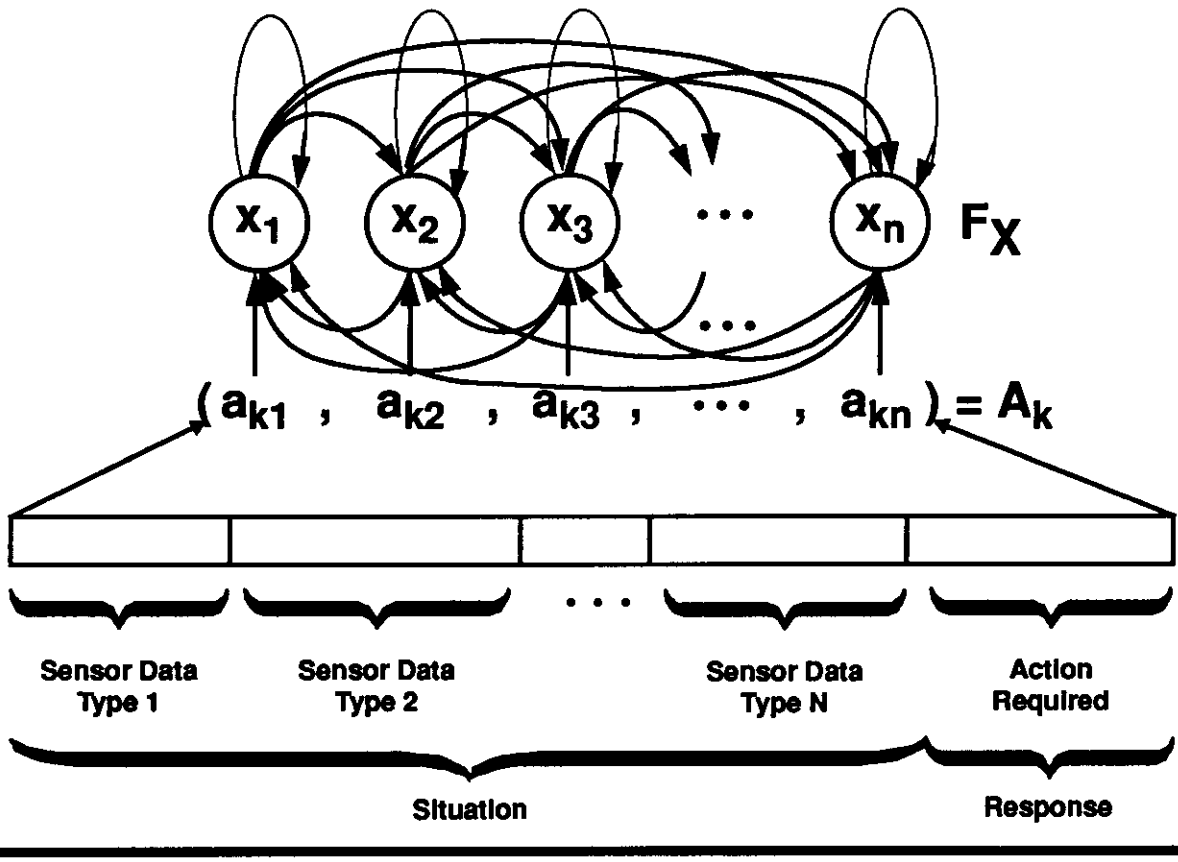
Pattern completion neural fusion primarily relies on autoassociative feedback neural networks (Simpson, 1990a & 1991). Neural networks that can be used for pattern completion include the Brain-State-in-a-Box (Anderson, et al., 1977) and the Hopfield associative memory (Hopfield, 1982). Because of the feedback nature of these systems, stability is usually achieved at the expense of nonlinear saturation points for each processing element's response. In other words, feedback neural systems tend to require a binary representation of the data to work most effectively.

The restriction to a binary representation requires some clever preprocessing that provides the requisite information. Several techniques have been developed for effectively representing information in a binary vector, including complete enumeration, thermometer codes, and closeness codes (Collins, 1990). When using pattern completion neural fusion it is vitally important to develop a robust code that can be used to represent the problem, or the full potential of the neural network will not be achieved. The code that is developed must accurately represent both the value of the sensor data and the relative importance of that sensor data.

3.2. Pattern Matching Neural Fusion

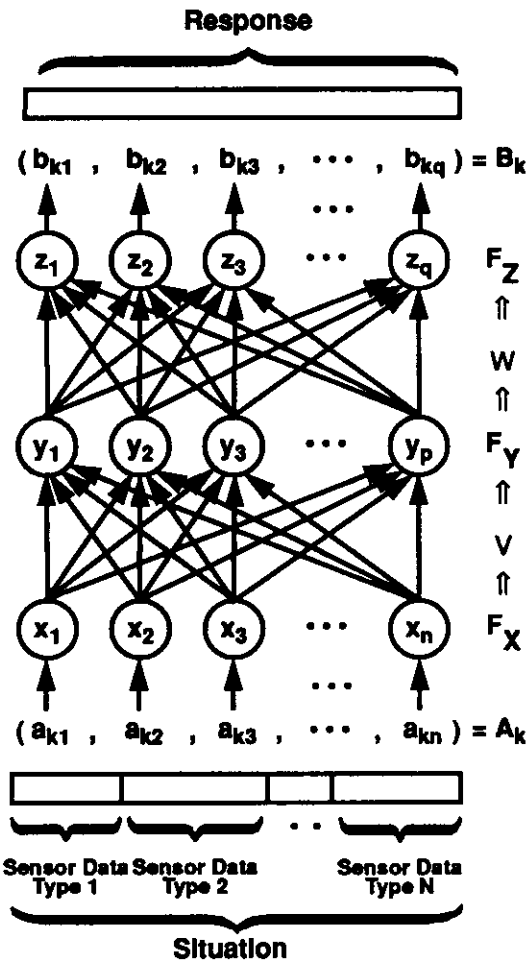
One of the most common forms of neural data fusion is the pattern matching approach. As shown in Figure 3, the situation is passed to

Figure 2: Pattern Completion Data Fusion



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Figure 3: Pattern Matching Data Fusion



the network as the input and the response is produced from the network as an output. There are several neural networks that can be used for pattern matching neural fusion, including the Boltzmann Machine (Ackley, Hinton & Sejnowski, 1985), the Cauchy Machine (Szu, 1986), the Probabilistic Neural Network (Specht, 1990), the Adaline/Madaline (Widrow & Winter, 1988), the Functional Link Net (Pao, 1989), and backpropagation (Werbos, 1974; Parker, 1982; Rumelhart, Hinton & Williams, 1986). With the exception of the Adaline/Madaline and the Functional Link Net, each of these pattern matching neural networks have more than two layers. Although it is not necessary to have a multi-layer neural network for pattern matching neural fusion, the interrelationships between the various sensor data types tend to be nonlinear and multi-layer neural networks tend to be the most common form of nonlinear pattern matching networks (others include higher-order neural networks such as the Functional Link Net). In addition to the pattern matching neural networks, it is also possible to include the pattern classification networks such as Adaptive Resonance Theory (Carpenter & Grossberg, 1987a & 1987b), Learning Vector Quantization (Kohonen, 1990), and the Fuzzy Adaptive Min-Max Unsupervised Classifier (Simpson, 1990b).

3.2.1. Automatic Target Recognition

Rewrite to eliminate system type, numbers and specific methods --

Ruck, et al. (1990) have used the multilayer neural network pattern matching MDF approach for the discrimination of various objects in images. The data used in the experiments was forward looking infrared (FLIR) and absolute range. After the image was segmented into blobs, features where abstracted from the data from each of the two sensors. The FLIR data was broken into a feature set that included number of pixels in the blob, background standard deviation, and complexity (ratio of border pixels to total pixels). The absolute range feature set included height of blob, complexity of blob (computed the same as the FLIR complexity), and pixel standard deviation across the blob.

The features where then concatenated together to form a large input vector to a backpropagation network. The MDF system was first tested using only range data to classify the blobs. This demonstrated showed that the backpropagation network was able to handle the fusion problem effectively and the performance improved when multisensor data was used. Backpropagation is not the only neural classifier that could have been used. Other neural network pattern classifiers could have resulted in

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

an equally acceptable solution. In the next example of pattern matching neural fusion the output is not a classification of the response, rather it is a set of values, hence a pattern classification system would not be applicable here.

3.2.2. Space Object Status Monitoring

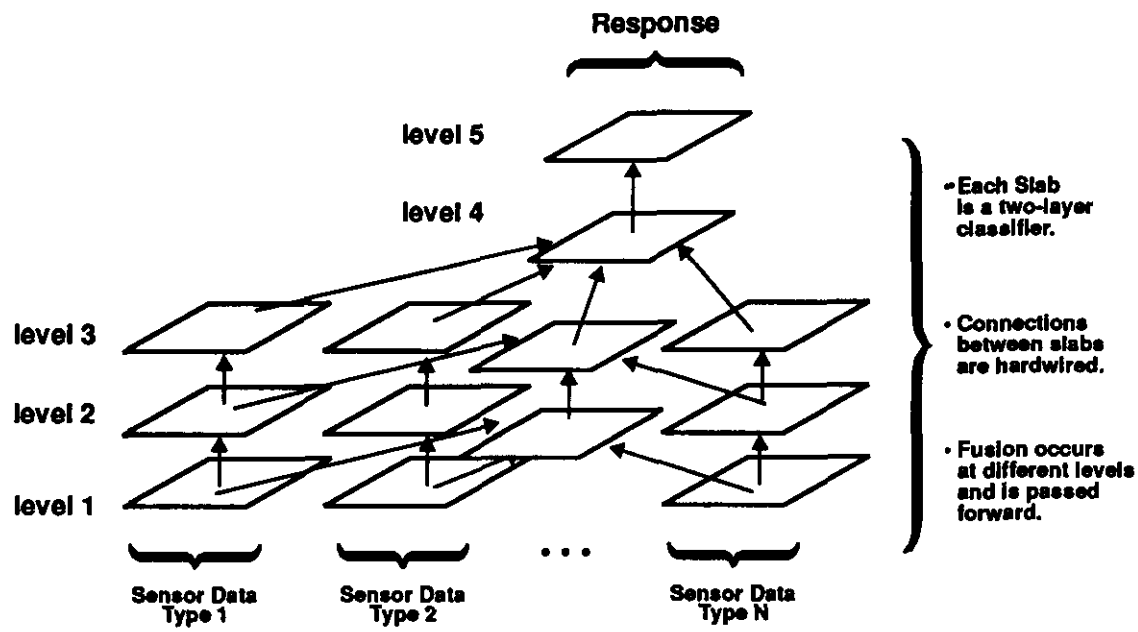
Eggers & Khuon (1990) have used a backpropagation network for the monitoring of space objects. The sensor data consisted of two radars, one operating in the L-band and the other operating in the X-band. Each set of sensor data was preprocessed using a fourth-order autoregressive model that produced a four-dimensional feature vector. These two vectors were concatenated together to form the input to the backpropagation network. The output from the network was a four dimensional vector describing the current state of the object (stable, pitch, roll, and yaw). The performance of the system showed reliable output responses.

3.3. Hierarchical Neural Fusion

Hierarchical neural networks are used in fusion systems that require low level sensor information to be abstracted into higher level features prior to the fusion. In the previous two instances it was assumed that the feature extraction process was sufficient enough to create a representation that could be used by a neural network. Sometimes it is not possible to extract enough information from non-neural network techniques, especially in image processing applications where scale, rotation, and translation invariance are key elements that need to be addressed prior to fusion.

Figure 4 shows a typical hierarchical network. This network has five levels. There are several input planes (level 1) that receive data from the sensors. In successive levels the features are gradually extracted from the data and fused together. At level 4 there is a final fusion of the information into a representation that is classified by level 5. Each level of the hierarchical network is a two-layer neural network classifier. The connections within each level (slab) are modifiable and are used to classify the features into a more abstracted representation. The connections between levels are hard-

Figure 4: Hierarchical Network Data Fusion



THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

wired to extract certain types of feature composites. Typical adaptation algorithms for these modifiable connections include Hebbian learning (Fukushima, 1988), competitive learning (Hecht-Nielsen, 1990), and adaptive resonance (Rajapakse, Jakubowicz, & Acharya, 1990).

The first system to employ this form of hierarchical composition was the Neocognitron (Fukushima, 1988) which was applied to handwritten character recognition. Other applications of the neocognitron include situation analysis (Jakubowicz, 1990) and automatic target recognition (Gilmore & Czuchry, 1990).

3.3.1. Target Recognition

An ART-1 based hierarchical system has been applied to the recognition of target-like images (Rajapakse & Acharya, 1990). The input sensors were simulated to represent two different types of data. The features present at each sensor are demonstrated to be insufficient to classify the image when used alone, but the combination of sensors was successful at the same task. The system is currently being extended to work with biomedical images.

4. APPLICATIONS OF NEURAL DATA FUSION TO GUIDANCE AND CONTROL

There are several areas where neural fusion can be applied to guidance and control. The following three sections outline some candidate application areas and provide some guidelines for applying the neural fusion techniques described above.

4.1. Guidance Systems

Guidance systems require a few decisions to made from a massive amount of data. Neural networks are ideally suited for these types of applications. Because of the parallel nature of neural networks, additional sensors will not necessarily slow the system. In addition, neural networks are able to automatically weight the relative importance of each type of sensor automatically.

4.2. Control Systems: Inverted Pendulum

Sometimes there are several different types of sensor data available, but the use of the data is not clear. The inverted pendulum (broom balancing) is an example of such as system. Sensors placed on the cart and on the joint of the inverted pendulum can be used to produce data that is used to determine which direction to move the cart so the pendulum will remain upright. It is possible to add an image sensor that can also determine the position of the pendulum relative to the cart. Fusing the information is not straightforward using conventional techniques, but a pattern matching neural fusion approach using a supervised learning neural network like the backpropagation network presents a feasible approach.

Other platforms that might utilize data fusion for control include robotics, automobiles and aircraft. Robots can utilize MDF for navigation purposes. Information from high-frequency active sonar and from cameras can be fused to control a mobile robot. Cars with look-ahead cameras can provide data that can be fused with sensors on the suspension system to produce commands back to the suspension system that will adjust the tension to fit the needs of the road. And, aircraft can fuse engine sensor data to control air intake and fuel flow to optimize for fuel efficiency, speed, or stealth purposes.

4.3. Surveillance Systems: Border Surveillance

Although it is not a strict guidance or control application, the use of neural fusion for surveillance is extremely promising and worthy of mention. One of the most difficult elements in a surveillance system is the fusion of data from the massive number of available sensors.

As an example, border surveillance sensors might include acoustic, seismic, radar and intelligence. Effectively fusing this data to clas-

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

sify the activity is difficult using conventional techniques. But, it is possible to train a neural network to perform this task by supplying the network with examples of the various sensor readings and the associated activity using a pattern matching neural fusion approach. Correlations that might not have been intuitively obvious are often discovered by pattern matching neural networks, an extremely useful attribute in this application.

Other areas where data fusion can be used include home security systems that fuse motion and infrared data to determine if an intruder is in the area. The output of the system can be used to control lights and sirens in the localized area of intrusion while automatically notifying law enforcement.

5. CONCLUSIONS

Neural fusion techniques are becoming more prominent because of their ability to easily handle massive amounts of data from a wide variety of sources. The use of data fusion provides a mechanism for improving the reliability of guidance and control systems at the expense of greater system complexity and more computational requirements. The use of neural networks in a MDF environment represents a natural fit of the strengths of neural networks with the weaknesses in data fusion.

REFERENCES

Anderson, J., Penz, A., Collins, D. & Gately, M. (1990). Radar signal categorization using a neural network, Proceedings of the IEEE, August.

Anderson, J., Silverstein, J., Ritz, S., & Jones, R. (1977). Distinctive features, categorical perception, and probability learning: Some applications of a neural model, Psychological Review, Vol. 84, pp. 413-451.

Barinaga, M. (1990). The mind revealed?, Science, Vol. 249, pp. 856-858.

Carpenter, G. & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine, Computer Vision, Graphics and Image Understanding, Vol. 37, pp. 54-115.

Carpenter, G. & Grossberg, S. (1987b). ART2: Self- organization of stable category recognition codes for analog input patterns, Applied Optics, Vol. 26, pp. 4919-4930.

Collins, D. (1990). Applications of neural networks to multisensor fusion, UCLA Extension Short Course Notes, October 15-19.

Fukushima, K. (1988). Neocognitron: A hierarchical neural network capable of visual pattern recognition, Neural Networks, Vol. 1, pp. 119-130.

Garvey, T. (1987). A survey of AI approaches to the integration of information, in Proceedings of the SPIE, Vol. 782, Infrared Sensors and Sensor Fusion, R. Buser & F. Warren, Eds., Orlando, FL, pp. 68-82.

Gilmore, J. & Czuchry, A. (1990). An application of the neocognitron in target recognition, Proceedings of the International Neural Network Conference: Volume 1, (pp. 15-18). Paris, France, July 9-13, Kluwer Academic Publishers: Dordrecht, Netherlands.

Hecht-Nielsen, R. (1990). Neurocomputing, Addison- Wesley: Reading, MA.

Hopfield, J. (1982). Neural networks and physical systems with emergent collective computational abilities, Proceedings of the National Academy of Sciences, Vol. 79, pp. 2554-2558.

Jakubowicz, O. (1990). Multilayer multi-feature map architecture for situation analysis, Proceedings of the IEEE/INNS International Joint Conference on Neural Networks: Volume II, (pp. 19-30). IEEE: Piscataway, NJ.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

Kagel, J., Platt, C., Donaven, T. & Samstad, E. (1990). Multispectral image fusion using *neural networks*, Proceedings of the SPIE: Volume 1294 - Applications of Artificial Neural Networks, S. Rogers, Ed., Orlando, FL, April 18-20. SPIE: Bellingham, WA.

Kam, M., Naim, A., Labonski, P. & Guez, A. (1989). Adaptive sensor fusion with nets of binary threshold elements, In Proceedings of the IEEE/ INNS International Joint Conference on Neural Networks (IJCNN 89): Volume II, (pp. 57-64). Piscataway, NJ: IEEE.

Kohonen, T. (1990). Self-Organization and Associative Memory: Third Edition, Springer-Verlag: Berlin.

Luo, R. & Kay, M. (1989). Multisensor integration and fusion in intelligent systems, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, pp. 901-931.

Maren, A. (1990). Neural networks for data compression and data fusion, In Maren, A., Harston, C. & Pap, B., The Handbook of Neural Computing Applications, (pp. 401-407). San Diego: Academic Press.

Maren, A. & Pereira, L. (1989). Multisensor information fusion: A comprehensive literature review, TCNEA AHCEL Team Technical Report 89-01, University of Tennessee Space Institute.

Mitchie, A. & Aggarwal, J. (1986). Multiple sensor integration/fusion through image processing: A review, Optical Engineering, Vol. 25, pp. 380-386.

Pao, Y-H. (1989). Adaptive Pattern Recognition and Neural Networks, Addison-Wesley: Reading, MA.

Parker, D. (1982). Learning logic, Stanford University, Dept. of Electrical Engineering, Invention Report 581- 64, October.

Rajapakse, J. & Acharya, R. (1990). Multi-sensor data fusion within hierarchical neural networks, In Proceedings of the IEEE/ INNS International Joint Conference on Neural Networks (IJCNN 90 - SD): Volume II, (pp. 17-22). Piscataway, NJ: IEEE.

Rajapakse, J., Jakubowicz, O. & Acharya, R. (1990). A real-time ART-1 based vision system for distortion invariant recognition and auto-association, In Proceedings of the IEEE/ INNS International Joint Conference on Neural Networks (IJCNN 90 - DC): Volume II, (pp. 298-301). Washington, DC, January 15-19. Lawrence Erlbaum Associates: Hillsdale, NJ.

Rhody, H., Sher, D. & Modestino, J. (1989). Intelligent signal processor techniques for multi-sensor surveillance systems, Final Technical Report, RADC-TR- 89-292, Rome Air Development Center, New York.

Ruck, D., Rogers, S., Kabrisky, M. & Mills, J. (1990). Multisensor fusion classification with a multilayer perceptron, In Proceedings of the IEEE/ INNS International Joint Conference on Neural Networks (IJCNN 90): Volume II, (pp. 863-868). IEEE: Piscataway, NJ.

Rumelhart, D., Hinton, G. & Williams, R. (1986). Learning representations by back-propagating errors, Nature, Vol. 323, pp. 533-536.

Simpson, P. (1990a). Artificial Neural Systems: Foundations, Paradigms, Applications and Implementations, Pergamon Press: Elmsford, NY.

Simpson, P. (1990b). The fuzzy adaptive min-max *unsupervised classifier, in review.*

Simpson, P. (1991). Foundations of neural networks, this NATO Lecture Series.

Smotroff, I., Howells, T. & Lehar, S. (1990). Meteorological classification of satellite imagery using neural network data fusion, In Proceedings of the IEEE/ INNS International Joint Conference on Neural Networks (IJCNN 90): Volume II, (pp. 23-28).

This Document Provided by the Abbott Aerospace
TECHNICAL LIBRARY
abbottaerospace.com

Piscataway, NJ: IEEE.

Specht, D. (1990). Probabilistic neural networks, Neural Networks, Vol. 3, pp. 109-118.

Szu, H. (1986). Fast simulated annealing, In J. Denker (Ed.), AIP Conference Proceedings 151: Neural Networks for Computing, (pp. 420-425). American Institute of Physics: New York.

Werbos, P. (1974). Beyond Regression, Ph.D. Dissertation, Harvard University, Cambridge, MA.

Widrow, B. & Winter, R. (1988). Neural nets for adaptive filtering and adaptive pattern recognition, IEEE Computer Magazine, March, pp. 25-39.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM
LS-194

ADVANCE NEURAL NETWORK ARCHITECTURES FOR GUIDANCE AND CONTROL

Todd Gutschow

HNC, Inc.
5501 Oberlin Drive
San Diego, CA 92121
619-546-8877

Robert Hecht-Nielsen

HNC,Inc.
and
Dept. of Electrical & Computer Engineering
University of California, SanDiego
La Jolla, CA 92093

Summary

Several advanced neural network architectures are expected to be of significant value in guidance and control. This paper reviews three advanced neural network architectures (the graded learning network, the recurrent backpropagation network, and the hierarchical matched filter network) and briefly discusses how they might be applied to problems in guidance and control.

1 Introduction

Many interesting problems in guidance and control can be reduced to the problem of implementing a time dependent mapping (i.e., a *spatiotemporal mapping*) between an n-dimensional input vector and an m-dimensional output vector. Such mapping problems are difficult to solve using conventional techniques such as linear control theory, statistical pattern recognition, or dynamic programming, due to the inherent complexity of spatiotemporal patterns, particularly when insensitivity to various warping transforms is demanded. Recent advances in neural network technology may provide significant new capabilities for addressing many of these problems.

This paper presents three neural network architectures that solve spatiotemporal mapping problems: the recurrent backpropagation network, the graded learning network, and the hierarchical matched filter network. It is expected that these networks will become increasingly important in solving complex spatiotemporal mapping problems. Thus, the focus of this paper is to familiarize the reader with the structure and operation of each network, and to point out similarities and differences.

These three networks were selected because they represent each of the three neural network learning paradigms: supervised learning (recurrent backpropagation network), reinforcement learning (grading learning network), and self-organization (hierarchical matched filtering network). This will allow us to compare the different types of learning and to gain insight into the suitability of each learning paradigm for various types of problems.

The graded learning and recurrent backpropagation networks are very similar in their approach to approximating spatiotemporal mappings. Their primary differences are in the training procedures that are used. The common architecture shared by these two networks is described in Section 3. This architecture consists of a single functional layer of fully connected processing units. Both of these network architectures address problems involving the approximation of arbitrary fixed spatiotemporal mappings.

In contrast, the hierarchical matched filtering network is designed specifically for spatiotemporal pattern classification problems. Its network architecture is fundamentally different than that of the other two networks. This architecture is described in Section 6.

2 Spatiotemporal Mappings

Intuitively, we can describe a spatiotemporal mapping as a mapping from a temporal sequence of n-dimensional input vectors to a temporal sequence of m-dimensional output vectors. Such an intuitive description can be made mathematically precise. We define $\mathrm{H}^{n,k,p}[C]$ to be the vector Sobolev space of all L^p generalized n-dimensional real vector functions of time with L^p generalized derivatives up to order k on a compact set $C \subset R$ (for a gentle introduction to generalized functions see [14], for a

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

terse definition see [1]). With this definition, a *spatiotemporal mapping* is defined as a mapping

$$\mathbf{x} : A \subset \mathrm{H}^{n,k,p}[C] \longrightarrow B \subset \mathrm{H}^{m,k',p'}[D].$$

Examples of spatiotemporal mappings include a speech classifier that maps a time-varying speech power spectrum to a word class number, and a control system that maps a plant disturbance to a system control function. More specifically, a speech classifier takes a time-varying speech power spectrum (a *spatiotemporal pattern*)

$$\mathbf{x} : C \subset R \longrightarrow R^n$$

and maps it to a class number function which, at each time step, gives the class number of the word that has most recently been completed (i.e., the class number function is an integer-valued function of time).

In a control system we typically have a plant with mathematical form

$$f(\mathbf{x}(t), \mathbf{u}(t), \mathbf{d}(t)) = \dot{\mathbf{x}}(t),$$

where $\mathbf{x}(t)$ is the state vector of the plant (typically composed of sensor readings) at time t, $\dot{\mathbf{x}}$ is the time rate of change of the state, $\mathbf{u}(t)$ is the vector of control signals at time t, and $\mathbf{d}(t)$ is the vector of plant disturbances (deviations from perfect closed-system mathematical operation) at time t. The goal of the control system is typically to achieve some kind of particular plant state (such as a specific final sheet thickness in a steel rolling mill). Thus, a *control system* is a mapping from an outside disturbance function $\mathbf{d}$ to a control function $\mathbf{u}$ that can achieve the desired control goal. This view of control theory assumes that the plant has a fixed dynamical structure so that the controller's job is to produce a control vector that deals with the effects of outside disturbances on the plant.

In general, the primary issue in spatiotemporal pattern recognition is to build classifiers that are insensitive to certain spatiotemporal warping transformations (such as pitch change and time warping in speech recognition). The primary issue in control is to build causal recursive controllers (i.e., controllers that operate in discrete time to map the set $\{\mathbf{x}(0), \mathbf{x}(1), \mathbf{u}(1), \mathbf{x}(2), \mathbf{u}(2), \ldots, \mathbf{x}(t-1), \mathbf{u}(t-1)\}$ into $\mathbf{u}(t)$) that perform well with respect to some particular set of goals. The graded learning network and the recurrent backpropagation network are useful for such control applications. The hierarchical matched filter network is useful for pattern recognition problems where there is a desire to be insensitive to time warps (a class of spatiotemporal warping transformations that map a spatiotemporal pattern $\mathbf{x}(t)$ into a pattern $\mathbf{x}(\theta(t))$, where θ is a strictly monotonically increasing smooth scalar function of time).

3 A Fully Connected Network Topology

A simple yet very powerful network topology is that of a single fully connected layer of processing units (see [10] for a discussion of the capabilities of this topology), such as shown in Figure 1, which consists of a single functional layer of N units. To simplify the discussion, an additional layer of fanout units is included. This layer distributes both the fed back output signals of the N functional units, and the n components $x_1(t-1)$, $x_2(t-1), \ldots, x_n(t-1)$ of the input vector $\mathbf{x}(t-1)$ (the input vector used during the network's operation at time t is latched into the fanout units at time $t-1$ along with the fed back processing element output signals from time increment $t-1$). Each of the N processing elements of the functional layer also receives a bias input, which we shall label $x_0(t-1)$ where $x_0(t) \equiv 1.0$ for all values of t. The number of fanout units is equal to $1+n+N$.

The outputs of the network at time t are the outputs $y'_1(t), y'_2(t), \ldots, y'_m(t)$ of the first m processing elements of the functional layer of the network. The output signals of the remaining units are $y'_{(m+1)}(t), y'_{(m+2)}(t), \ldots, y'_N(t)$.

To simplify the notation, we define

$$z_j(t) = \begin{cases} x_j(t) & \text{if } 0 \le j \le n \\ y'_{(j-n)}(t) & \text{if } (n+1) \le j \le L \end{cases} \qquad (1)$$

where $j = 0, 1, 2, \ldots, L$ and $L = N + n$. For convenience we shall assume that time always begins at $t = 0$.

On time step t, processing element i calculates its output signal $y'_i(t)$ by means of the formula

$$y'_i(t) = s_i(I_i(t)) \qquad i = 1, 2, \ldots, N \qquad (2)$$

$$I_i(t) = \sum_{j=0}^{L} w_{ij}\, z_j(t-1) \qquad (3)$$

where each of the functions $s_i(u)$ is bounded and has a continuous derivative. A typical functional

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

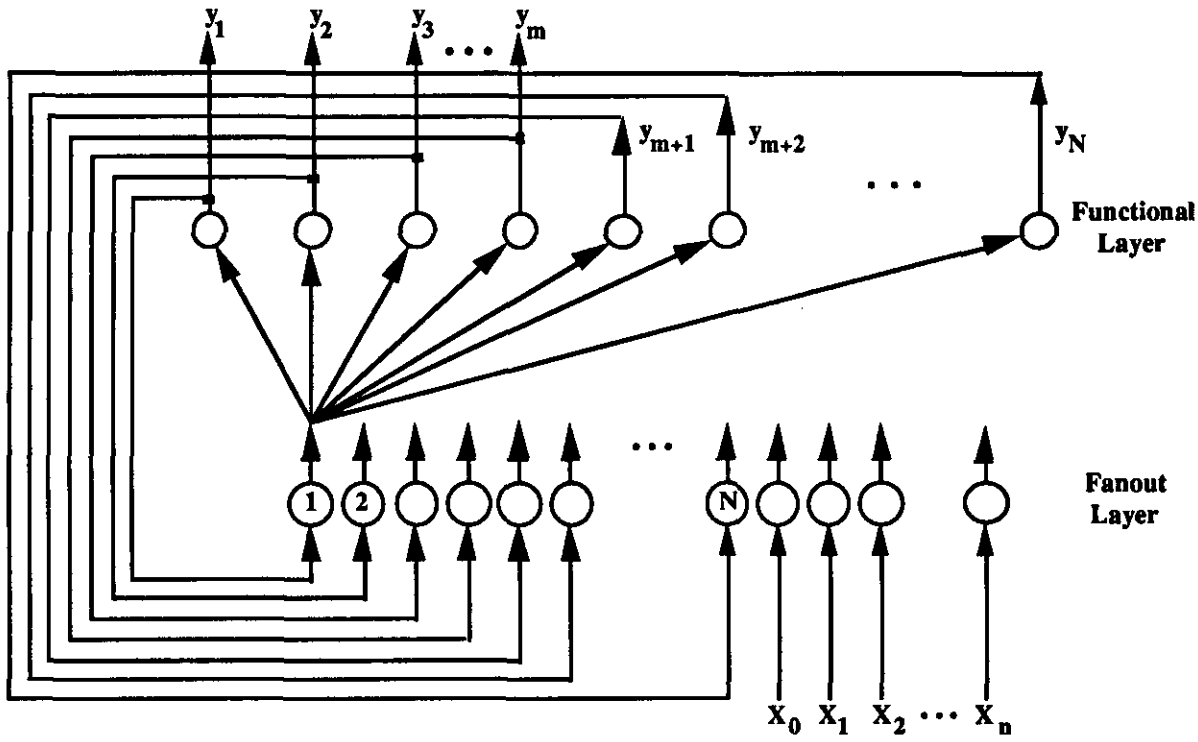


Figure 1: Single layer of fully connected processing units.

form for $s_i(u)$ is the bipolar logistic function given by

$$s_i(u) = \frac{1 - e^{-u}}{1 + e^{-u}}. \quad (4)$$

This function is bounded between -1 and +1 and has a slope of 1 at zero.

To solve a spatiotemporal mapping problem with the network topology shown in Figure 1, the connection weights must be learned from a set of examples of the mapping. The next two sections describe learning methods that yield good connection weights for this network topology.

4 Recurrent Backpropagation Network

The recurrent backpropagation network learns to approximate a mapping between a sequence of n dimensional input vectors and a sequence of m dimensional output vectors. The mapping is learned using a form of supervised learning to adapt the weights. Supervised learning requires that the some of outputs of the network be known for some or all of the input vectors in the sequence. In general, such information is more difficult to acquire than a measurement of performance. Thus, recurrent backpropagation is more restrictive than the graded learning network in terms of the types of problems that it can address. However, when supervised learning can be used it will in general produce a network that is superior to graded learning both in terms of required training time and approximation accuracy. Thus, when supervised learning can be used it should be.

4.1 Recurrent Backpropagation Error Function

Unlike the graded learning network, recurrent backpropagation has a fixed error function that it tries to minimize during training. This error function is a spatiotemporal generalization of the mean squared error function used in backpropagation. To understand this error function, we must first define the exact problem that recurrent backpropagation attempts to solve.

Let the input to and output from the system at time t be $\mathbf{x}(t-1)$ and $\mathbf{y}'(t)$, respectively. We shall

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

assume that the system starts operation at $t = 1$. Initial values for the internal states of the system at time $t = 0$ are uniquely defined by the initial values of the output signals (in other words, by the vector $\mathbf{y}'(0)$). The system runs forward in time until some arbitrary stopping time t_{stop} is reached. During each of these 'runs' of the system the input sequence $\{\mathbf{x}(0), \mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(t_{\text{stop}} - 1)\}$ is provided to the system. From the initial state of the system, $\mathbf{y}'(0)$, and the sequence of $\mathbf{x}(t)$ inputs, the system produces outputs $\{\mathbf{y}'(1), \mathbf{y}'(2), \ldots, \mathbf{y}'(t_{\text{stop}})\}$. Clearly then, the overall purpose of the system on each run is to map the set

$$\mathbf{x} \equiv \{\mathbf{y}'(0), \{\mathbf{x}(0), \mathbf{x}(1), \mathbf{x}(2), \ldots, \mathbf{x}(t_{\text{stop}} - 1)\}\}$$

into the set

$$\mathbf{y}' \equiv \{\mathbf{y}'(1), \mathbf{y}'(2), \ldots, \mathbf{y}'(t_{\text{stop}})\}.$$

Thus, we can view the operation of such a spatiotemporal system as performing a mapping from a set consisting of the initial system state and a set of input values provided over the run, to a set consisting of the output states produced by the system over the run. The confusing thing about this picture is that in many practical instances (such as most control systems) the $\mathbf{x}(t)$ inputs are functionally dependent upon earlier $\mathbf{y}'(t)$ outputs. The key observation is that *this doesn't matter.* The only effect this has is to limit the range of possibilities for the $\mathbf{x}(t)$ sequences that the system will see. We are only concerned with what the system does when a particular sequence of $\mathbf{x}(t)$ inputs is presented (given a certain initial state of the system). We don't care how these inputs arose.

The error calculation procedure for the recurrent backpropagation network is similar to that used with the backpropagation network, but with one important difference: not all correct output signals are known. In the case of recurrent backpropagation, we assume that with each training run example $\mathbf{x}$ we are also given information concerning some of the correct values of outputs of the network at various points during the run. Specifically, we assume that at each time t, $1 \leq t \leq t_{\text{stop}}$ during a training run we are given a set $U(t)$ of integers lying in the range from 1 to m, inclusive, such that the correct output value $y_k(t)$ for unit k at time t is given for each $k \in U(t)$. It is perfectly acceptable to have $U(t)$ be the empty set at some times t during the training run. However, for there to be useful training, $U(t)$ must be non-empty for at least one time t during training.

Given the sets $U(t)$, and the correct $y_k(t)$ values for each $k \in U(t)$, we define the mean squared error $F(\mathbf{w})$ of the recurrent backpropagation network to be

$$F(\mathbf{w}) = \mathrm{E}\left[\frac{1}{K}\sum_{t=1}^{t_{\text{stop}}}\sum_{k\in U(t)}[y_k(t) - y'_k(t)]^2\right] \quad (5)$$

$$K = \left(\frac{1}{\sum_{t=1}^{t_{\text{stop}}} \#U(t)}\right) \quad (6)$$

where $\mathbf{w}$ is the weight vector of the network, $\#U(t)$ = the number of elements in $U(t)$ ($\#U(t) = 0$ if $U(t)$ is empty), and E[ ] is the expectation or averaging operator (the averaging is done over an unboundedly large number of input examples chosen randomly with respect to ρ). Note that the entire sum is divided by K, the total number of error terms used. Thus, we are measuring the average squared error per output for which the correct output is given. This quantity is then averaged over the entire input space by the expectation operator. Again, as with backpropagation, the mean squared error depends only on the weights. Naturally, for this dependency to hold, it must be assumed that the weights are fixed throughout the evaluation of the network's performance.

4.2 Recurrent Backpropagation Network Learning Law

The recurrent backpropagation network learning law is based on the standard gradient descent method

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \alpha\nabla_{\mathbf{w}}F(\mathbf{w}). \quad (7)$$

The gradient calculation requires the partial derivatives of $F(\mathbf{w})$ with respect to the components of $\mathbf{w}$. The complete derivation of these partial derivatives can be found in [10]. The result is a set of recursion formulas

$$r_{kij}(t) = s'_k(I_k(t)) \left([\delta_{ik}\, z_j(t-1)] + \sum_{p=1}^{m}[w_{kl}\, r_{kij}(t-1)]\right). \quad (8)$$

where

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

$$r_{kij}(t) = \frac{\partial y'_k(t)}{\partial w_{ij}},$$
$$r_{kij}(0) = 0.$$

At the end of each run (after all of the $z_l(t-1)$ values are known), the recursion formulas in Equation 8 can be solved. Naturally, in order to adequately approximate the expectation operator, we must average over a large number of runs where the initial values and input sequence examples are chosen randomly in accordance with a fixed probability density function ρ. The need to batch the results from a number of runs before modifying the weights makes this learning law very slow.

Two variations of this learning law have been developed. The first of these updates **w** after each time step and is known as the jump-every-time-step variation. The second updates **w** at the end of each run. Both of these variations can improve the training time of the network.

Another variant of the recurrent backpropagation learning law is the *teacher-forced* learning law introduced by Ronald Williams and David Zipser [18] (who also introduced Equation 8). This variant is like the jump-every-time-step version, except for two changes. First, all of the correct output values $y_k(t)$ that we are given for training are used in the recursion equation (Equation 8) in place of the corresponding $y'_k(t)$ values. Second, after each weight jump the $r_{kij}(t)$ value used to compute the jump is set to zero. Williams and Zipser report that, at least for some problems, the teacher forced learning law seems to converge to a useful solution faster than the original learning law or the two other variants.

It is worth noting that the above derivation assumes that the inputs to the network do not depend upon the weight values. For many practical problems, such as in control, this assumption will be false because of the fact that the input is derived from the output (for example, by a plant that takes control signal outputs from the network, which definitely depend on the weights, and produces sensor inputs to the network — which therefore also depend upon the weights). Thus, in using the recurrent backpropagation network, this limitation must always be kept in mind. However, this isn't to say that the method is unusable in these cases. Often, the dependence of the input on the weights is small, in which case the method may still work.

5 The Graded Learning Network

The graded learning network (GLN) is a mapping neural network which uses a form of reinforcement learning in which a performance measure or grade is periodically presented to the network to guide learning [4]. It combines the well known optimization characteristics of simulated annealing [13,7] with the speed advantages of a gradient search method. The result is a powerful new method of optimization for a broad class of problems, including guidance and control.

Unlike supervised learning networks such as backpropagation, GLN does not require the desired output to be furnished for each training trial. Only a measure of overall network performance over a series of training trials is required. This is very significant for problems in guidance and control, since these problems are often characterized by a lack of knowledge of the desired output for a given training trial.

5.1 GLN Advantages

While GLN is not the only form of reinforcement learning network, it does have two distinct advantages over other such networks:

1. The GLN learning law does not specify the form of the grading function.
2. The GLN learning law is not coupled to the network topology.

The first of these advantages implies that the grade must be furnished by an entity external to the network. This external entity is typically some type of monitoring module which can assess the overall performance of the system. Such a performance measure can be very complex and often involves significant time delays between the network response and the measurement. In general, the grade measurement can be based upon any factors that are consistently and repeatably a function of the input-output behavior of the network.

The second GLN advantage allows it to be applied to arbitrary network topologies. In particular, GLN can be used with topologies that involve feedback connections between and within layers of processing elements. Thus, GLN can be applied to problems that have complex dynamical response with unknown or uncertain time delays.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

5.2 Description of Graded Learning Operation

The learning law for the graded learning network is executed whenever a grade, G, is presented. Upon such a presentation, the network adjusts its weights using a training process that is, roughly speaking, a biased form of Cauchy simulated annealing [15]. The bias is based on an estimate of the gradient of the grading function. Thus, each weight adjustment is a combination of a grading function gradient estimate and a Cauchy random jump. A temperature parameter determines the average size of this random jump.

In the following discussion of GLN learning, it will be convenient to define the network weight vector, $\mathbf{w}$, as the vector containing the weights of all the units in the functional layer, including the bias weights. The dimensionality of $\mathbf{w}$ is $q = (1 + n + N)(N)$.

In addition to $\mathbf{w}$, GLN maintains three other vectors of the same dimensionality for use during training. The first of these vectors, $\mathbf{a}$, is an estimate of the gradient of the grading function with respect to the network weight vector. The second vector, $\mathbf{b}$, contains the network weight vector that thus far has yielded the best (lowest) grade value. The final vector, $\mathbf{c}$, contains the random jump values.

When a grade, G, is presented to the network, it is first checked to determine if it is better or worse than the current best grade, $G_{\text{best}}(t)$. Subsequent processing depends on the outcome of this check:

case 1: $G < G_{\text{best}}(t)$

$$\begin{aligned} G_{\text{best}}^{new} &= \alpha G + (1-\alpha) G_{\text{best}}^{old} \\ T^{new} &= \beta T \\ \mathbf{a}^{new} &= \gamma \mathbf{a}^{old} + \delta \mathbf{c}^{old}. \end{aligned}$$

case 2: $G \geq G_{\text{best}}(t)$

$$\begin{aligned} G_{\text{best}}^{new} &= \alpha G_{\text{best}}^{old} + (1-\alpha) G \\ T^{new} &= \epsilon T \\ \mathbf{a}^{new} &= \theta \mathbf{a}^{old} + \phi \mathbf{c}^{old}. \end{aligned}$$

where α, β, γ, δ, ϵ, θ, and ϕ are parameters. Typical values for these parameters are given in Table 1.

Following these changes, the $\mathbf{c}$ and $\mathbf{w}$ vectors are updated as follows:

$$\mathbf{c}^{new} = \mathbf{a}^{new} + T\mathbf{r} \tag{9}$$

Table 1: Typical GLN Training Parameters

Parameter	Typical value
α	0.99
β	1.01
γ	0.85
δ	0.25
ϵ	0.995
θ	0.85
ϕ	-0.15

where $\mathbf{r}$ is a q-dimensional Cauchy random variable (see [15]). Finally, the new weight vector is calculated:

$$\mathbf{w}^{new} = \mathbf{b} + \mathbf{c}^{new}. \tag{10}$$

After $\mathbf{w}$ is updated, the network is run once again, with this new weight vector, to generate a new grade. The process of weight updating can be continued indefinitely (e.g., if the plant or its environment are expected to change significantly over time), or it can be turned off when a satisfactory level of performance is obtained.

6 Hierarchical Matched Filter Neural Network

The hierarchical matched filter network is designed to perform spatiotemporal pattern classification using a generalized multidimensional matched filter. Traditionally, matched filtering has been used in application areas such as communications, radar, and sonar, for detecting a specific waveform in a time series signal. The generalized multidimensional matched filter is optimized for spatiotemporal pattern classification. Banks of these matched filters can be used as high-performance classifiers for spatiotemporal patterns. Unfortunately, the direct implementation of such matched filter banks for large problems (such as large-vocabulary continuous speech recognition), while attractive, is not practical. However, it may be possible to develop a method for exploiting the inherent statistical redundancy of typical spatiotemporal pattern sets to allow more efficient implementations of such matched filter banks. In particular, we propose a hierarchical neural network approach to this implementation problem.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

6.1 Matched Filtering

One well-known method of pattern recognition is *template matching* or *nearest neighbor classification* [5,6], in which unknown patterns are simply compared with known examples (using an appropriate distance measurement procedure) to find the closest matching examples. Given a sufficiently rich set of example patterns, such classifiers can be shown to be near-optimal. However, for practical problems, classifiers with a sufficiently large number of example patterns are often impractical.

Given two spatiotemporal patterns, $\mathbf{u}(t)$ and $\mathbf{v}(t)$, we want to create a matched filter distance measurement that is invariant, or at least insensitive, to the distortion of patterns by some preselected class $\mathcal{C}$ of spatiotemporal warping transformations. For example, if we wished to be insensitive to small time warps, we might define the class $\mathcal{C}$ to consist of transformations of the form $\mathbf{u}(t) \longrightarrow \mathbf{u}(\theta(t))$ where $0.5 \leq d\theta/dt \leq 2.0$. Of course, $\mathcal{C}$ might consist of much more complicated transformations.

One choice for the distance measurement $H_{\mathbf{v}}(\mathbf{u}, t)$, that is invariant with respect to a class $\mathcal{C}$ of spatiotemporal warping transformations, and which only operates locally in time, is

$$H_{\mathbf{v}}(\mathbf{u}, t) \equiv \inf_{T \in \mathcal{C}} \int_{-\infty}^{\infty} \mu(\tau - t) \mid \mathbf{u}(\tau) - T\mathbf{v}(\tau) \mid d\tau, \tag{11}$$

where μ is a non–negative smooth function with $\mu(\tau) \geq 0$ for $\tau \in (-a, 0)$ (where a is a non–negative constant) and $\mu(\tau) = 0$ otherwise, and where $\mathcal{C}$ is a defined set of spatiotemporal warping transformations. The function μ is called a *time windowing function*. It serves the purpose of focusing the attention of the distance measurement on the time interval $[t - a, t]$. H can be interpreted as the distance between the spatiotemporal pattern $\mathbf{u}$ over the time interval $[t - a, t]$, and the best matching warped portion (of duration $[t - a, t]$) of $\mathbf{v}$. $H_{\mathbf{v}}(\mathbf{u}, t)$ which is called the *generalized multidimensional matched filter* (or simply *matched filter*, since we shall not use the traditional version in the sequel) for input spatiotemporal pattern $\mathbf{u}$, tuned to spatiotemporal pattern $\mathbf{v}$, over spatiotemporal warp class $\mathcal{C}$.

6.2 The Nearest Matched Filter Classifier

One way of building a pattern classifier for spatiotemporal patterns is to gather many examples of patterns belonging to each of the M classes into which each unknown input pattern is to be placed. An unknown spatiotemporal pattern can then be compared with these examples at each time t, by means of matched filters based upon the example patterns, to determine (via a classification decision policy) whether a pattern belonging to any one of the M classes has just finished arriving or not. This is the *nearest matched filter classifier.*

To make the notation concrete, let us define such a *training set* of patterns to be the set $P = \{(\mathbf{v}_1, \beta_1), (\mathbf{v}_2, \beta_2), ..., (\mathbf{v}_N, \beta_N)\}$, where $\beta_k \in \{1, 2, ..., M\}$ is the number of the class to which example pattern $\mathbf{v}_k$ belongs. The input signal, $\mathbf{u}$, is fed to all of these matched filters in parallel (the matched filters use weighting functions that are *balanced* so that their responses are comparable). The output of the classifier at time t is a class number β determined by putting the outputs of all N matched filters into a decision policy function. For example, if we wanted to use a simple 1-nearest neighbor policy, we would emit at each time t the class number β_i associated with the reference pattern $\mathbf{v}_i$ having the smallest matched filter output $H_{\mathbf{v}_i}(\mathbf{u}, t)$—unless the value of the smallest matched filter output exceeded a fixed threshold, in which case we would provide a class number output of 0, meaning that the input signal does not currently match any example pattern well. Clearly, the pattern class output typically will not be smooth (it will jump abruptly from one class number to another as the winning classifier of the competition process changes). The generalized multidimensional matched filter and the nearest matched filter classifier (along with a neural network implementation of the classifier for time warps) were introduced in 1982 [12]. For further information and discussion of these concepts, see [10].

The nearest matched filter classifier can be defined for a variety of spatiotemporal warping transformations. However, common choices might be time warping or pitch change transformations. Time warping would be useful, for example, for speech recognition, where the changes in how words are pronounced are typically of a time–warp nature. Pitch change transformations (such as those that occur when we speed up or slow down a

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

phonograph record) would be useful for recognizing vehicles by their sounds, since much of the sound of a vehicle is from its engine, transmission, and wheels, which produce sounds at pitches that are directly dependent on road speed and gear selection. In every case, the use of an appropriate class of transformations will ensure that each reference pattern can serve as a model for a wide class of similar, but transformed, patterns. This effective pattern reuse greatly reduces the number of reference patterns that must be used.

Finally, the theoretical classification performance of the nearest matched filter classifier has been established for the case where $\mathcal{C}$ is the set of time translations [11]. In this case, assuming that the training set is sufficiently comprehensive (and employs a 1-nearest neighbor classification decision policy), the classifier error rate will satisfy the Cover and Hart inequality [3]

$$R^* \leq R \leq R^* \left(2 - \frac{M}{M-1} R^*\right), \qquad (12)$$

where R^* is the error rate of the Bayes classifier.

The nearest matched filter classifier has one problem, and two advantages. The problem is that we may need an enormous training set; this requirement may make the direct implementation of such a classifier impossibly large and computationally burdensome (since all N of the $H_{\mathbf{v}_k}(\mathbf{u}, t)$ integrals must be computed in parallel). The advantages are that the classifier is capable of near-Bayesian performance (at least for some classes of spatiotemporal warping transformations), and that the individual matched filters are insensitive to noise. This latter advantage is particularly important if all of the matched filters are using the same weighting function (as opposed to weighting functions that merely have the same time integral), since Equation 12 shows that all of the matched filters will then react approximately the same to additive noise. Thus, since the decision process is typically largely a relative comparison of the matched filter outputs, the classifier output will be somewhat insensitive to additive noise. The combination of guaranteed high classification accuracy (given our ability and willingness to implement a sufficient training set) and additive noise insensitivity make the nearest matched filter classifier an interesting candidate for solving spatiotemporal classification problems.

Finally, because the windowing function limits the consideration of the incoming spatiotemporal pattern to the time interval $[t - a, t]$, the nearest matched filter classifier can carry out only the first local-in-time stage of spatiotemporal pattern recognition. For many problems, local-in-time classification is not sufficient. Often, to do a good job of classification, we must exploit context information that we can obtain only by considering longer periods of time. One way to do this would be to devise a classification decision policy function that could exploit a priori syntax and context information. Because such a postprocessing operation is often essential if adequate performance is to be achieved, the nearest matched filter classifier should really be thought of as just a front end for a complete classifier. We now consider the problem of implementing a nearest matched filter classifier in a hierarchical neural network structure.

6.3 Nearest Matched Filter Classifier Implementation

A neural network that approximately implements the nearest matched filter classifier for the class of time warp spatiotemporal warping transformations (see [10]) has the disadvantage that it requires one sub-network for each example pattern in the training set. Thus, the size of the network grows linearly with the size of the training set.

For many problems, such as continuous speech recognition, the patterns in the training set will be highly redundant. In other words, these patterns will have many sub-patterns (phonemes, for example) in common—usually at several different time duration levels. Thus, from a statistical perspective, a direct implementation of such a nearest matched filter classifier will be highly inefficient, since each matched filter will contain units that are tuned to essentially the same short-term patterns that a multitude of other units are also tuned to. Consolidating these units would decrease the size of such an implementation enormously – perhaps making such systems practical. This section presents an outline of a scheme for accomplishing this consolidation by means of a new hierarchical design.

Figure 2 shows a design for a self-organizing spatiotemporal feature detector layer. This layer learns short time sequences of patterns in a way that makes it insensitive to small time warps. Perhaps the best way to describe the function of this layer is to begin with a description of how it is trained. Then its function during normal operation

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

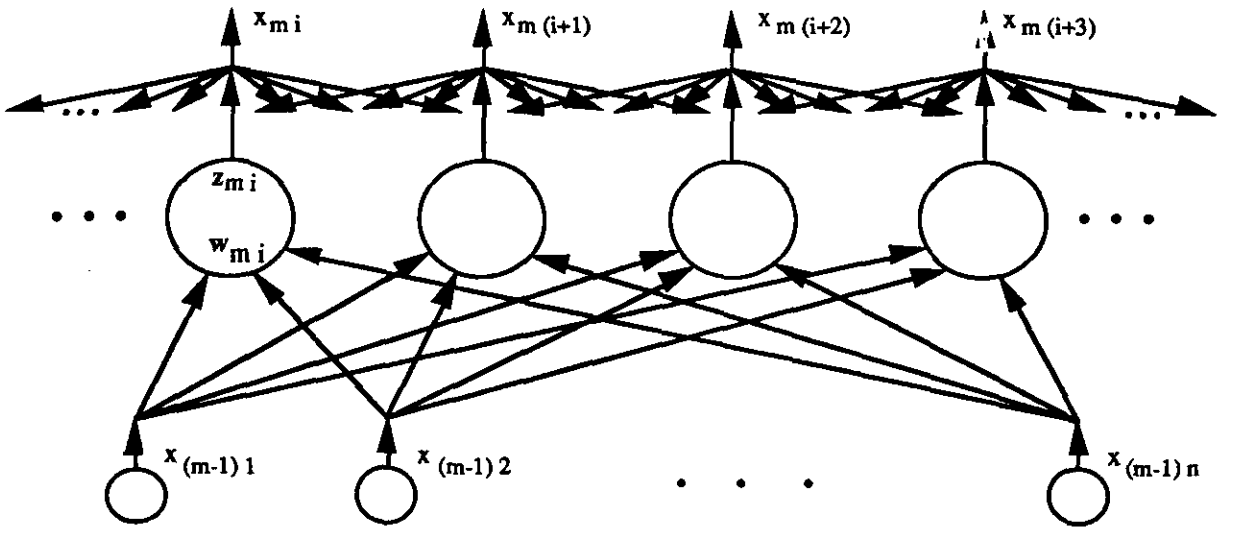


Figure 2: Schematic for a self-organizing spatiotemporal feature detector layer.

will be described.

During both training and normal operation of the hierarchical neural network classifier, we assume that the spatiotemporal patterns are entered into the hierarchy at the bottom as sequences of vector inputs in discrete time. The sample rate is greater than the Nyquist rate for the fastest varying component of the pattern. Further, we assume that the individual patterns to be classified have durations that are all approximately the same (this condition is not necessary, but relaxing it adds complications that will be avoided in this paper). The patterns are assumed to arrive in a random order described by a fixed probability density. The only spatiotemporal warping transformations are assumed to be mild time warps. Given these assumptions, we now consider the training of layer m of the hierarchy. We assume that all of the previous layers have already been trained and their weights have been frozen.

The first step in training layer m is to train the spatial weight vectors $\mathbf{w}_{m1}, \mathbf{w}_{m2}, \ldots, \mathbf{w}_{mN}$. These are trained using Kohonen learning with conscience (see [10] for details), with each successive training trial utilizing the next discrete time sample of input $\mathbf{x}_{(m-1)}(t)$ from the previous layer as the training vector. The α learning rate constant starts off at a value near 1.0 and decreases to 0 in accordance with a cooling schedule.

After this training process converges, the $\mathbf{w}_{mi}$ vectors will be distributed in $\mathbf{x}_{(m-1)}$ space such that each time sample $\mathbf{x}_{(m-1)}(t)$ of the input to layer m is equally likely to be closest (measured using Euclidean distance) to each of the $\mathbf{w}_{mi}$ weight vectors. At this point these spatial weight vectors are frozen and the training of the $\mathbf{z}_{mi}$ temporal weight vectors begins.

Before temporal weight training begins, the processing elements are modified. Unlike spatial weight training, where the processing elements simply respond at each discrete time to the distance from the current input to the unit's spatial weight vector, now in temporal weight training, the reaction to inputs will have a temporal behavior. Specifically, each processing element will now be governed by equations such as

$$x_{mi}(t) = \alpha(-c\, x_{mi}(t-1) + d\, U(\psi - |\mathbf{w}_{li} - \mathbf{x}_{m-1}(t)|)),$$

$$0 \leq x_{mi}(t) \leq 1,$$

$$U(\zeta) = \begin{cases} 1 & \text{if } \zeta > 0 \\ 0 & \text{if } \zeta \leq 0, \end{cases}$$

and

$$\alpha_{li}(\xi) = \begin{cases} \xi & \text{if } \xi \geq 0 \\ \phi\, \xi & \text{if } \xi < 0, \end{cases}$$

and where $\mathbf{w}_{mi}$ is the spatial weight vector of unit i of layer m, and c, d, ψ, and ϕ are positive constants, with $c, \phi < 1$.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

These equations ensure that each unit is activated only if the input vector $\mathbf{x}_{(m-1)}(t)$ is sufficiently close to the spatial weight vector $\mathbf{w}_{mi}$ of that unit. The *attack function* α is used to ensure that the "spin up" of each unit is faster than than the "spin down".

Given equations of the above sort, each processing element within ψ range of the input vector $\mathbf{x}_{(m-1)}(t)$ will become activated. The constants are chosen so that this activation always hard limits at 1 within a few time units after the input vector enters the ψ sphere surrounding its weight vector. After the input vector leaves this sphere, the activity of the processing element slowly decays. Note that by setting the value of ψ correctly it will be possible to ensure that an approximately constant fraction of the units is always active—obviating the need for the development (as yet unachieved) of a "local" competition mechanism.

Given the above unit behaviors, a steady stream of input patterns is then entered into the system, and the temporal weights z_{mij} (which are all initially zero) are modified by means of the Kosko/Klopf learning law (see Section 3.6 of [10] for details). This establishes temporal weights in accordance with commonly encountered sequences of unit activation.

Following equilibration, the temporal weights are frozen (if desired, to improve later performance, the weights can first be "sharpened" via a sigmoidal transformation before freezing). The layer is now ready to be prepared for use. To do this, yet another transfer function is introduced.

Following the freezing of the weights of the unit, the transfer functions employed during operational use of the layer are inserted into the units. This transfer function has a form such as

$$\begin{aligned} x_{mi}(t) \;=\; & \alpha(-c\,x_{mi}(t-1) \;+ \\ & d\,U(\psi - |\mathbf{w}_{li} - \mathbf{x}_{m-1}(t)|\,) \\ & [\theta + \sum_{j \neq i} z_{mij}\; x_{mj}]\,) \end{aligned}$$

$$0 \leq x_{mi}(t) \leq 1.$$

The behavior of this transfer function is now briefly described. First, for activation of unit i of layer m to occur, the input vector $\mathbf{x}_{(m-1)}(t)$ from the previous layer must be passing through the sphere of radius ψ surrounding the unit's spatial weight vector $\mathbf{w}_{mi}$. Second, the activation level reached (if not hard limited at 1.0) will depend on the sum of the quantity θ and the temporal input intensity $\sum_{j \neq i} z_{mij}\; x_{mj}$. To achieve full activation, the temporal input intensity must be quite large (this ensures that during training unit i is frequently active following the layer m units currently supplying it highly weighted input). The offset $\theta > 0$ is used to ensure that units that lie at the start of learned spatiotemporal sequences will become at least modestly active, even though they do not have any predecessor units helping to get them activated. In the end, this scheme (and other variants) provides a spatial pattern of activity that represents a history of the trajectory of the input $\mathbf{x}_{(m-1)}(t)$ over the last brief interval of time. The history recorded by this network layer is, in terms of a set of spatiotemporal segments, burned into the network during training. If the input pattern deviates too much from one of these trajectory segments, the layer will not respond much at all.

From the above observation it is clear that this spatiotemporal layer is, in fact, acting as a generalized matched filter bank over a brief interval of time, with each activity constellation representing a pattern trajectory segment learned during training. The transfer function used precludes constellations from becoming highly active unless this is so (unless, of course, the layer has been overloaded). Note that if overloading occurs the layer can simply be made larger and the ψ constant can be lowered. This allows the use of larger numbers of (more spatially discriminating) units to learn the spatiotemporal subtrajectories. Note that this layer will be insensitive to modest time warps, due to the gradual activation and deactivation behavior of the operational transfer functions.

The above discussion has reviewed the definition of a new matched filter for spatiotemporal patterns and introduced a hierarchical layered neural network designed to efficiently implement a bank of such matched filters for the purpose of achieving spatiotemporal pattern recognition that is insensitive to small time warps. In order to derive the desired classification information, a mapping network must be employed that will transform the spatial constellations of activity at the highest layers into a class number and a confidence level.

The self-organizing layer defined here has only limited redundancy of spatial pattern representation (in contrast to the Spatiotemporal Pattern Recognizer network presented in Section

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

6.1 of [10], which has enormous redundancy). Each subsequent layer in the hierarchy has a time constant $1/c$ that is twice as long as the layer below. This "temporal compression" property ensures that the activity constellations at higher and higher layers act as codes for longer and longer sequences of spatiotemporal pattern. It is conjectured that, if the layers are not overloaded and if the spatiotemporal patterns are sufficiently distinct, these constellation codes will be unique. Further, in general, if the input pattern does not resemble a pattern presented during training, then none of the layers will respond significantly.

The architecture presented here moves us one step closer towards efficient implementation of large matched filter banks for spatiotemporal pattern classification.

7 Applications to Guidance and Control

In this section we will review a number of applications of interest to guidanace and control problems.

7.1 Recurrent Backpropagation

Recurrent backpropagation has demonstrated the ability to model complex dynamical systems. Such a capability could be very useful in guidance and control applications. For example, consider a seeker system that must distinguish between different types of objects such as a fighter aircraft and a flare. One approach to distinguishing between these objects is by their dynamical behavior. Flares exhibit very simple dynamical behavior (they fall) while fighter aircraft have significantly more complex dynamical behavior (they turn, accelerate, etc.). Such dynamical behavior models could be developed using a recurrent backpropagation network. The network would learn to predict the next location of an object given its recent dynamical behavior. The predicted location could then be compared with the sensed location to make a targeting decision. The network could be trained using actual examples of human pilots flying either real or simulated aircraft.

7.2 Graded Learning

The graded learning network is most applicable to control problems in which the objective of the controller is difficult or impossible to measure. For example, it may be desirable to design a missile control system which maximizes its range. Such a control system does not have an absolute measure of performance since the maximum range attainable is a function of the mission. However, we can determine how often the missile reaches its target and use this value to assign a success measure to the control system. The graded learning network can use this success information integrated over a number of trial mission (either actual or simulated) to learn an appropriate control law.

7.3 Hierarchical Matched Filter

The hierarchical matched filter network is most applicable to spatiotemporal pattern classification problems in which insensitivity to time warp transformations is desired. An example of such a problem is speech recognition in which we desire a system that can classify speech independent of how fast the speaker is speaking.

8 Conclusions

This paper has presented three new neural network architectures for addressing complex spatiotemporal mapping problems such as those encountered in guidance and control. The structure and operation of each network was reviewed, and application suggestions were given. From this discussion, it is clear that advanced neural network architectures hold great promise for developing next generation guidance and control systems. Additional research and development aimed at better characterizing the properties of these networks and exploring their applications is required to realize this promise.

References

[1] Banks, S. P., **Mathematical Theories of Nonlinear Systems**, Prentice Hall, New York, 1988.

[2] Chen, D., and Zhou, F., "A neurally-based controller for target-tracking sight control system", *Proceedings of the International AMSE (Association for the Advancement of Modeling and Simulation techniques in Enterprises - Paris, France) Conference*, San Diego, CA, 29-31 May 1991.

THIS DOCUMENT PROVIDED BY THE ABBOTT AEROSPACE
TECHNICAL LIBRARY
ABBOTTAEROSPACE.COM

[3] Cover, T.M. and Hart, P.E., "Nearest neighbor pattern classification", *IEEE Trans. on Infor. Th.*, **IT-13** 21–27, January 1967.

[4] DeSieno, D., "Graded Learning", U.S. Patent, 1988.

[5] Devijver, P. A., and Kittler, J., **Pattern Recognition: A Statistical Approach**, Prentice–Hall, Englewood Cliffs NJ, 1982.

[6] Duda, R. O., and Hart, P. E., **Pattern Classification and Scene Analysis**, Wiley, New York, 1973.

[7] Geman, S., and Geman, D., "Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images", *IEEE PAMI*, **6**, 721-741, 1984.

[8] Grossberg, S. [Ed.], **Neural Networks and Natural Intelligence**, MIT Press, Cambridge MA, 1988.

[9] Hecht-Nielsen, R., "Towards hierarchical matched filtering", in: Mammone, R. J., and Zeevi, Y.Y. [Eds.], **Neural Networks: Theory and Applications**, Academic Press, Boston, MA, 1991.

[10] Hecht-Nielsen, R., **Neurocomputing**, Corrected First Edition, Addison-Wesley, Reading, MA, 1991.

[11] Hecht-Nielsen, R., "Nearest matched filter classification of spatiotemporal patterns", *Applied Optics*, **26**, (10), 1892–1899, 1987.

[12] Hecht-Nielsen, R., "Neural Analog Processing", *Proc. SPIE*, **360**, 180–189, 1982.

[13] Kirkpatrick, S., Gelatt, Jr., C. D., and Vecchi, M. P., "Optimization by bSimulated Annealing", *Science*, **220**, 671-68-, 1983.

[14] Lighthill, M. J., **Introduction to Fourier Analysis and Generalized Functions**, Cambridge University Press, London, 1964.

[15] Szu, H., Hartley, J., "Fast Simulated Annealing", *Physics Letters A*, **122**, 157–162, 1987.

[16] Waibel, A., "Modular construction of time-delay neural networks for speech recognition", *Neural Computation*, **1**, 39–46, Spring 1989.

[17] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K., "Phoneme recognition: Neural networks vs. hidden Markov models", *Proc. ICASSP*, **S3.3**, 107–110, April 1988.

[18] Williams, R. J., and Zipser, D., "A learning algorithm for continually running fully recurrent neural networks", *Neural Computation*, **1**, 270–280, Summer 1989.

[19] Hornik, K., Stinchcombe, M., and White, H., "Universal Approximation of an Unknown Mapping and Its Derivatives Using Multilayer Feedforward Networks", *Neural Networks*, **3**, 551–560, 1990.