MULTISENSOR DATA FUSION

Arthur Filippidis, B.E

Department of Electrical and Electronic Engineering

The University of Adelaide,

Adelaide, South Australia.

September 1993.

Awarded 1994

Table of Contents

Abstract

Kinematic Data Fusion:

Sensor Data Fusion involves the entire process of Correlation and Fusion. The thesis addresses the application of Artificial Neural Networks (ANN), to the correlation problem of sensor level tracks. A comparison is then made between two track-to-track correlation techniques for multisensor fusion using simulated and real track data (using DSTOS FPS-16 Radar and Adelaide Airports Surveillance Radar). The techniques are, Classical Inference using Hypothesis Testing and ART2 (Adaptive Resonance Theory 2 Neural Network).

Attribute Data Fusion:

The thesis addresses the application of the Backpropagation neural network to Data Fusion for automatic target recognition using three knowledge sources: a Continuous Wave (CW) Coherent (X band) Radar, which provides us with high resolution doppler signature measurements, together with a Surveillance Radar, which provides positional information of airborne targets, and priori information of flight times of targets flying regular flight paths, obtained from Adelaide Airport Flight time tables. A comparison is then made between three data fusion techniques, on the trial data obtained. They are, backpropagation neural network, Dempster-Shafer and Fuzzy Reasoning.

Declaration

This thesis has been submitted to the Faculty of Engineering at the University of Adelaide for examination in respect for the Degree of Masters of Engineering Science.

This thesis contains no material which has been accepted for the award of any other degree or diploma in any University, and to the best of the author's knowledge and belief contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

The author hereby consents to this thesis being made available for photocopying and for loans as the University deems fitting, should the thesis be accepted for award of the Degree.

Arthur Filippidis

September 1993.

Acknowledgments

The author would like to express his very sincere gratitude to the following for the way each has assisted in this project.

Professor R. E. Bogner, for his invaluable assistance as supervisor for this project, and for his enthusiasm and personal encouragement generally.

Peter Calder (current supervisor at DSTO, Information Technology Division), Jeff Schapel (DSTO), Dr. R. Jarrott (Telecom), Dr. C Marlin (Flinders University), Dr. D Kewley (DSTO) and Dr. A. Bouzerdoum (Adelaide University) for their helpful comments, conversations, support and encouragement.

The Department of Aviation, by making available Adelaide Airports radar facilities for recording and tracking of selected airborne targets and the very helpful assistance of their staff.

Microwave Radar Division for the use of their CW radar, and assistance of their staff.

Members of the Neural Group at the University of Adelaide and the Data Fusion SIG. at D.S.T.O for sharing their understanding of the rapidly progressing field of artificial neural networks and data fusion through participation in their meetings.

Financial support from Defence Science and Technology Organisation is gratefully acknowledged.

Finally, I thank my wife for her encouragement and support and dedicate this thesis to our son John born on the 18th. of January 1992.

Publications

A. Filippidis, J.G. Schapel, "Automatic Allocation of Identity," Working Paper DSTO ITD/CSI-WP-92-4, (unpublished) 25 March 1992. Main Library P.O. Box 1500, Salisbury, S.A , 5108.

A. Filippidis, R.E Bogner, "Track-to-Track Correlation for Sensor Level Tracking," Proc. ISSPA92, Gold Coast, Australia, August 1992, pp. 666-669.

A. Filippidis, R.E Bogner, "Neural Networks for Sensor Data Fusion," Proc. OE/Aerospace and Remote Sensing, Multisensor Fusion1, Orlando, Florida, USA, April 1993.

List of Figures

Figure

List of Tables

Chapter 1

Introduction

1.1 Outline of the Thesis

In this introductory chapter, I begin by defining and describing some of the operational benefits and application areas of multiple sensor fusion, and by summarising a number of the basic techniques used to assist or perform data fusion.

In chapters 2 and 3 I define and discuss some of the principles of kinematic and attribute data fusion and some of the background theory for the techniques used in the experimental chapters 4 and 5.

In chapter 4 (Experimental Procedures and Results for Kinematic Data-Fusion) a comparison is made between two track-to-track correlation techniques for multisensor fusion using simulated and real track data. They are, Classical Inference using Hypothesis Testing and the Adaptive Resonance Theory 2 Neural Network. The simulated track data is generated using matlab software shown in Appendix C and the real track data is obtained from the Surveillance radar at Adelaide Airport and the FPS-16 tracking radar at DSTO. For the real data (in section 4.3) I address the problem of producing a common space co-ordinate system for both radars, before correlating the tracks.

In chapter 5 (Experimental Procedure and results for Attribute Data Fusion) I address the application of the backpropagation neural network to data fusion for automatic target recognition using three knowledge sources. They are, a Continuous Wave (CW) radar at DSTO, which provides us with high resolution doppler signature measurements, together with a Surveillance radar, which provides positional information of airborne targets, and priori information of flight times of targets flying regular flights, obtained from Adelaide Airport flight time tables. A comparison is then made between three data fusion techniques, on trial data obtained. They are, backpropagation neural network, Dempster-Shafer and Fuzzy Reasoning. I introduce this chapter (section 5.1.1.1) by discussing some of the differences of jet engine and propeller aircraft modulation obtained from the CW radar (discussed in more depth in Appendix F). Also an introduction to some of the signal processing techniques used to process the doppler data for input to the neural network is discussed in section 5.3 with a more in depth exposition in Appendix A. A numerical example showing how Dempster-Shafer and Fuzzy reasoning combine the information from the three knowledge sources is discussed in sections 5.7 and 5.8 respectively.

Conclusions on experimental results for both kinematic and attribute data fusion are made in chapter 6. For the kinematic fusion case I discuss the advantages obtained in using ART2 over the Classical Inference approach. And for the attribute fusion case I discussed how objectives were met to achieve automatic allocation of identity of airborne targets with a neural network data fusion based system. The advantages are listed and a comparison is made of using the backpropagation neural network over the Dempster-Shafer and fuzzy reasoning data fusion methods.

1.2 Applications, Benefits and Definitions for Multisensor Fusion

The application of multiple sensors (and the fusion of their data) to the problems of detection, tracking and identification offer numerous potential performance benefits over traditional single sensor approaches [1],[2],[3]. These performance benefits must of course, be weighed against additional cost, complexity, and interface requirements introduced for any given application. Characteristics of multisensor systems that provide operational benefits include the following:-

Robust Operational Performance is provided because any one sensor has the potential to contribute information while others are unavailable, denied (jammed), or lacking in coverage of an event or target.
Extended Spatial Coverage is provided because one sensor can look where another sensor cannot.
Extended Temporal Coverage is also provided because one sensor can detect or measure an event at times that others cannot.
Increased Confidence (a relative measure of an uncertainty in the measured information) is accrued when multiple independent measurements are made on the same event or target.
Reduced Ambiguity in measured information is achieved when information provided by multiple sensors reduces the set of hypotheses about the target.
Improved Detection Performance results from the effective integration of multiple, separate measurements of the same target.
Enhanced Spatial Resolution is provided when multiple sensors can geometrically form a sensor aperture capable of greater resolution than that of a single sensor.
Improved System Operational Reliability may result from the inherent redundancy of a multisensor suite.
Increased Dimensionality of the measurement space (ie., different sensors measure various portions of the electromagnetic spectrum) reduces vulnerability to denial (countermeasures, jamming, weather, noise etc.) of any single portion of the measurement space.

Kinematic Data Fusion involves the use of kinematic quantities such as position, range rate, etc. However, with thc usc of advanced radar systems and advances in radar signal processing techniques, the efficient use of attribute data fusion is becoming more

important.
Attributes are sensed target quantities that are associated with a particular class of target. These may include such quantities as wheel (or tread) type for ground targets, engine type for aircraft (ie. propeller or jet), type of emitting radar for either ground or aircraft targets, or target image shape. Also, the class or type of target (ie. truck or tank) may itself be considered an attribute. In the next few paragraphs I will consider kinematic and attribute data fusion separately.

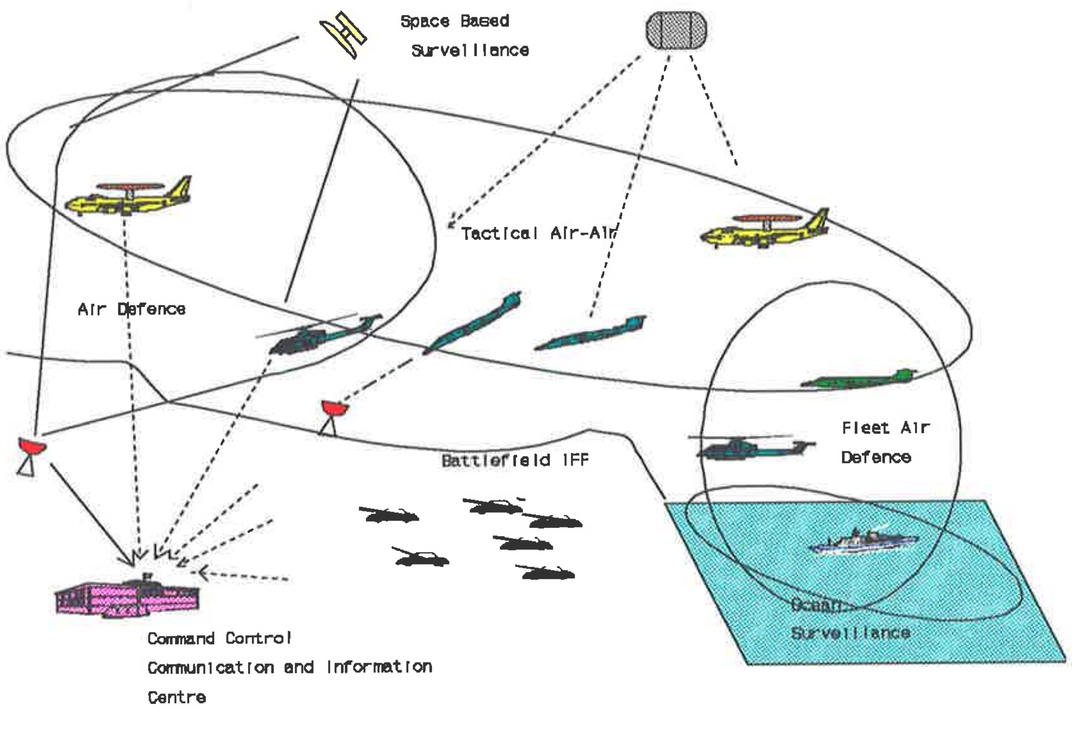


Figure 1 Application Areas for Fusion. (Taken from [4]).

In general, Data Fusion techniques seek to combine observational data from multiple sensors (and types of sensors) to locate and identify both emitting and non-emitting sources of targets. These sensors may include active sensors such as radars, synthetic aperture radar ,sonar or other devices as well as passive sensors such as imagery, infra-red detectors, electronics intelligence collectors, and other types of collectors. Generally, fusion systems aim to combine multi-sensor data to achieve a higher degree of accuracy and identification specificity than can be obtained via a single data source [4].

Direct fusion, is also referred to as local or autonomous fusion, and is defined as the fusion of all sensors on board a single weapons platform. Indirect fusion, also referred to as global or regional fusion, is defined as the fusion of widely distributed sensors which may look at targets from different spatial and temporal aspects. These concepts improve the effectiveness of both tactical identification and strategic Command, Control and Communications (C^3) systems for a wide variety of applications. Figure 1 shows the primary military application for which fusion solutions have been used or considered. Air Defence is a candidate for indirect fusion through use of C^2 assets of an entire sensor network composed of ground radars, airborne warning and control systems, and intelligence sensors. Ocean Surveillance systems use multiple sensor data (ship, air, space and underwater) to derive tracking data for ocean traffic. Battlefield IFF (identification friend or foe) refer to direct fusion on the battlefield. Space-Based Surveillance provides a wide range of sensor measurements (IR, radar, ESM) for detection and tracking of strategic threats. In the future the fusion of these sensor inputs from spatially and temporally separated spacecraft will be a requirement [4].

The basic fusion model (Figure 2) incorporates the aspect of sensing, tracking, and identification. The model highlights the primary functions :
Sensors may be located at the fusion node, or may be remotely located, passing information along a data link. The sensing process may be cooperative (question-answer) or non-co-operative and active or passive. Sensor reports may not be synchronised in time.
Tracking and Report Correlation is required to correlate the various sensor reports to determine which sensor reports are associated with distinct targets in space. Once this correlation is performed, target data sets (or track files for dynamic targets) may be formed and maintained.
Combination and Classification must be performed on each target data set (or track file) to determine if the set can be uniquely identified as known target class. This requires an optimal combination of the data from multiple sensors and a decision process to establish class and decision confidence.

1.3 Summary of Data Fusion Techniques

The next few paragraphs summarise the basic techniques used to assist in, or perform data fusion [2],[3]. These techniques include: classical methods of statistics and inference, Bayesian inference, Dempster-Shafer modification to Bayesian inference, fuzzy-set theory, cluster analysis, estimation techniques, templating, figure of merit, expert systems and entropy methods. The particular techniques used (in experimentation), will be discussed in more detail later.

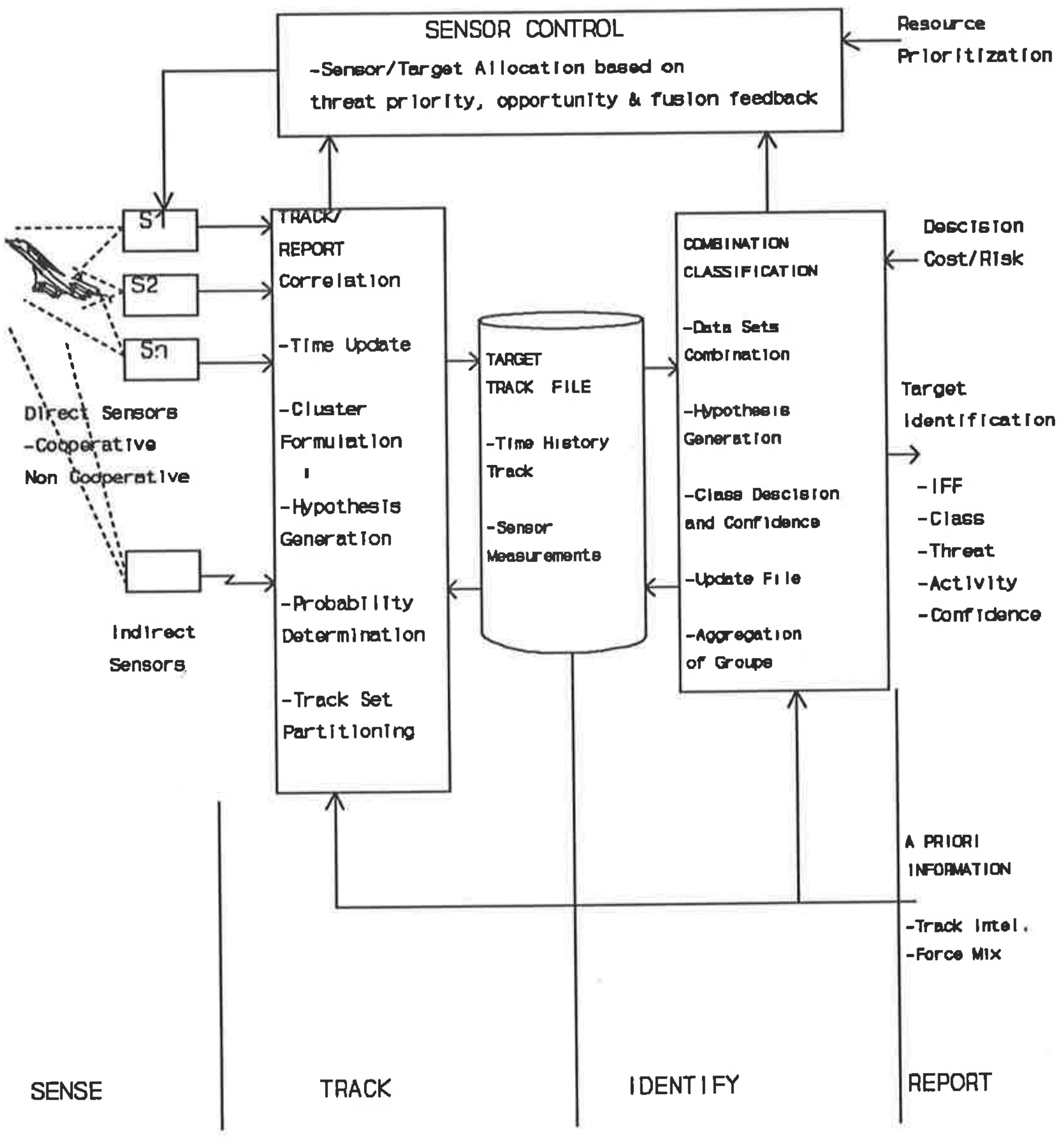


Figure 2 General Functional Elements of the Fusion Process. (Taken from [4]).

Classical inference techniques compute the probability of an observed event given an assumption of a priori probabilities. Hence, classical inference describes the probability of an observed event given a hypothesis. Typically, however, we seek the probability of a hypothesised situation given observations of events. Classical inference is well based on mathematical theory. Strict application requires knowledge of a priori probability distributions which are clearly unknown in some realistic applications. Disadvantages of classical inference techniques are: they require a priori sampling distribution; can only assess two hypothesis at a time (the hypothesis H0 versus an alternative hypothesis H1); complexities can arise for multivariate data; and the classical inference does not take advantages of prior likelihood assessments, as does the Bayesian inference technique.

The Bayesian inference technique resolves some of the difficulties with the classical inference methodology. Bayesian inference updates the likelihood of a hypothesis given a previous likelihood estimate and additional evidence (observations). This methodology allows the use of subjective probability. The disadvantages of the Bayesian inference include: the difficulty in defining prior likelihood; complexities when there are multiple potential hypotheses and multiple conditionally dependent events; the requirement that competing hypotheses be mutually exclusive; and the lack of an ability to assign general uncertainty.

Shafer and Dempster created a generalisation of Bayesian theory which allows for general level of uncertainty. Based on this model of human inference, the Dempster-Shafer (D-S) method utilises probability intervals and uncertainty intervals to determine the likelihood of hypothesis based on multiple evidence. In addition D-S methodology computes a likelihood that any hypothesis is true.(For more detail refer to section 3.1).

Fuzzy set theory applies a generalised set theory to determine membership of entities in specified sets. A fuzzy set is one in which membership is not a boolean decision (eg. the set of tall people clearly contains marginal members- is a person 1.7 metres tall or not-tall). Fuzzy set theory supplies an algebra of set manipulations (such as union, disjunction , etc.) for fuzzy sets and their members. Fuzzy set theory is beginning to be applied in decision analysis involving imprecise events. (For more detail refer to section 3.3)

Cluster analysis embraces a number of methods for sorting observations into natural groups based on a prespecified similarity measure. Such techniques are useful for fingerprinting or unit identification. Cluster methods are basically ad hoc schemes for data sorting without underlying statistical theory. Such methods may be useful for identity declarations and analysing observational data when no theory exists for relating the observations to assigned identity or classification.

Estimation theory encompasses the techniques of maximum likelihood, Kalman filtering, weighted least squares, and Baycsian cstimation. These techniques obtain the best estimate of a state, given an observation corrupted by noise. An example is the estimation of an emitter's location given multiple line-of-bearing observations. Application include tracking and direction finding.

The Entropy method computes a measure of information content associated with a hypothesis. Applications exists for systems utilising empirical or subjective assessments of alternative hypotheses.

The Figure of Merit (FOM) algorithm computes a degree of similarity between two entities based on observational data and a priori weights. FOMs are frequently used in correlation schemes to make quantitative declarations of association.

Expert systems are computer programs which seek to mimic the ability of human specialists or experts to make decisions and inferences. Observational data are used to derive inferences based on a knowledge base which may contain facts, rule of thumb, and heuristic information. They are used for threat identification, situation assessment and tasks currently performed by military analysts.

Templating utilises general data records to perform pattern recognition for complex associations. Examples include event detection and recognition of high value targets. Observational data is matched against a priori template (or patterns) to determine if the data supports a hypothesis characterised by the templates. A template may contain parameter lists, Boolean conditions, weighting factors and thresholds to describe conditions for an event, activity or hypothesis.

Chapter 2

Principles of Kinematic Data Fusion

2.1 Track-to-Track Correlation for Sensor Level Tracking

Sensor data fusion involves the entire process of Correlation and Fusion . Fundamental to the problem of combining sensor-level tracks is determining whether two tracks from different systems (sensors) potentially represent the same track.

In a multiple-sensor tracking system the first major issue is to define the level at which sensor data will be combined into tracks. The choices are Sensor or Central level tracking [5],[6]. Unlike the Central level tracking approach which forms tracks from raw observations the Sensor level tracking

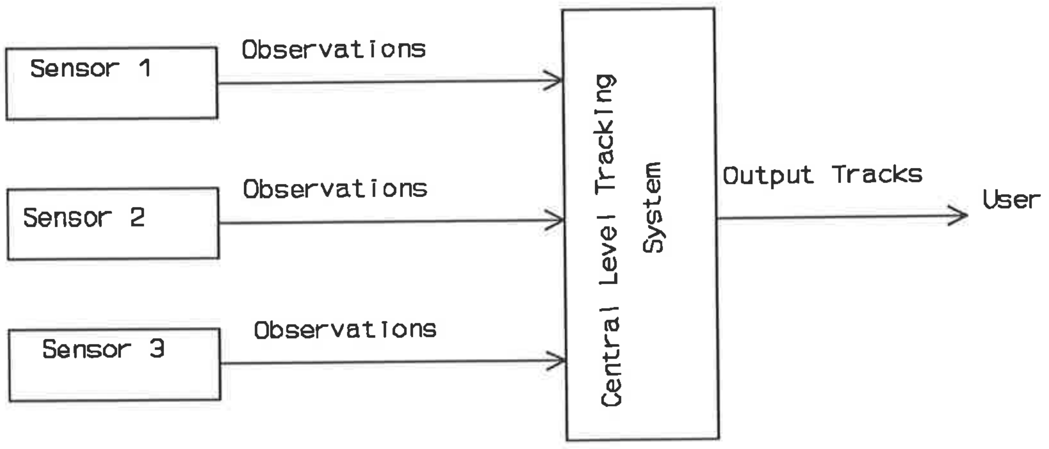


Figure 3 Central-level tracking approach forms tracks from raw observations (Taken from [5]).

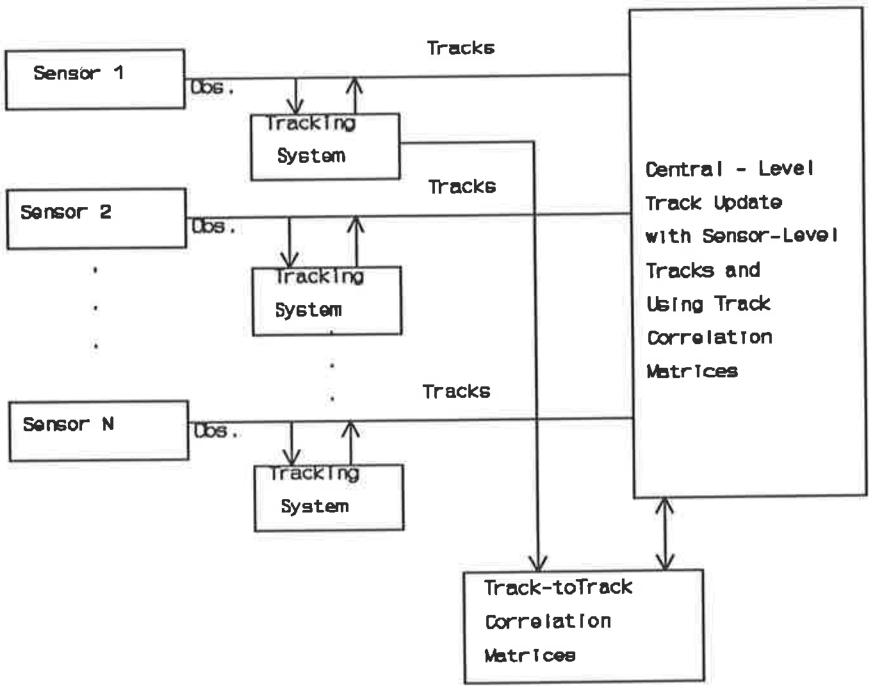

Sensor 1
Sensor 2
Sensor N
Obs.
Tracking System
Tracks
Central - Level Track Update with Sensor-Level Tracks and Using Track Correlation Matrices
Track-toTrack Correlation Matrices

Figure 4 Sensor-level tracking approach forms tracks and then combines them (Taken from [5]).

approach forms sensor tracks and then combines them (Refer to Figures 3 and 4) Points cited in favour of sensor-level tracking, over central-level tracking, are reduced data-bus loading, and higher survivability due to distributed tracking capabilities. Certain computational advantages may result from the parallel processing that is possible using sensor-level track approach. Also, if one sensor becomes degraded, its observations do not affect the sensor-level tracks of other sensors. Finally, the use of sensor-level tracking allows for filter design that is specifically tailored to the individual sensors.

The problem of track-to-track correlation arises when multiple sensors report tracks from a common surveillance volume. An important question is how to decide whether two tracks from two different sensors (using sensor-level tracking) represent the same target.

In the following paragraphs I will review the theory for two track-to-track correlation techniques for multisensor fusion. They are, Classical Inference using Hypothesis Testing (as discussed by Bar Shalom in references [6] and [7]) and ART2 (Adaptive Resonance Theory 2 Neural Network) renowned for its use in pattern recognition and its ability to respond in real time. Bar Shalom [6], outlined the Classical Inference technique based upon the use of the Chi-Squared properties of the difference in state estimation vectors.

2.1.1 Classical Inference Theory

2.1.1.1 Hypothesis Testing

Many problems require that we decide whether or not a statement about some parameter is true or false. The statement is usually called a hypothesis, and the decision making procedure about the truth or falsity of the hypothesis is called hypothesis testing [8].

We are interested in making a decision about the truth or falsity of a hypothesis. A procedure leading to such a decision is called a "test of a hypothesis". Hypothesis-testing procedures rely on using the information in a random sample from the population of interest. If this information is consistent with the hypothesis, then we conclude that the hypothesis is true; however if the information is inconsistent with the hypothesis, we would conclude that the hypothesis is false.

To test a hypothesis, we must take a random sample from the sample data, compute an appropriate test statistic, and then use the information contained in the test statistic to make a decision. When a decision is made using the information in a random sample, the decision is subject to error. Two kinds of error may be made when testing hypothesis. If the null hypothesis is rejected when it is true, then a type 1 error has been made. If the null hypothesis is accepted when it is false, then a type 2 error has been made. The situation is described in Table 1.

Table 1 Decisions in Hypothesis Testing

	H0 IS TRUE	H0 IS FALSE
ACCEPT H0	NO ERROR	TYPE2 ERROR
REJECT H0	TYPE 1 ERROR	NO ERROR

The probabilities of occurrence of type 1 and type 2 errors are given by the following:

α= P{type 1 error} = P{reject H0 / H0 is true}
β= P{type 2 error} = P{accept H0 / H0 is false}

Because the results of a test of a hypothesis are subject to error, we cannot "prove" or "disprovc" a statistical hypothesis. However it is possible to design test procedures that control the error probabilities α and β to suitably small values.

The probability of type 1 error is often called the "significance level" or size of the test. For example consider the following sensor target track problem used in the experimental section 4.0. We have two tracks from two independent radars tracking targets in a common surveillance volume. In reference [6] Bar Shalom outlines a technique based on the use of the Chi-Squared properties of the difference in the state estimation vector of Y1 and Y2. Consider two tracks with state estimation vectors Y and covariance matrices P as defined by:-

TRACK1 : Y1(K) , P1(K)
TRACK2 : Y2(K) , P2(K)

where,

Y1(K) = True state of target by sensor1 at time (K)
Y2(K) = True state of target by sensor2 at time (K)

Let $E_{12} = Y1(K)-Y2(K)$

The problem of track association can be regarded as the following hypothesis testing problem

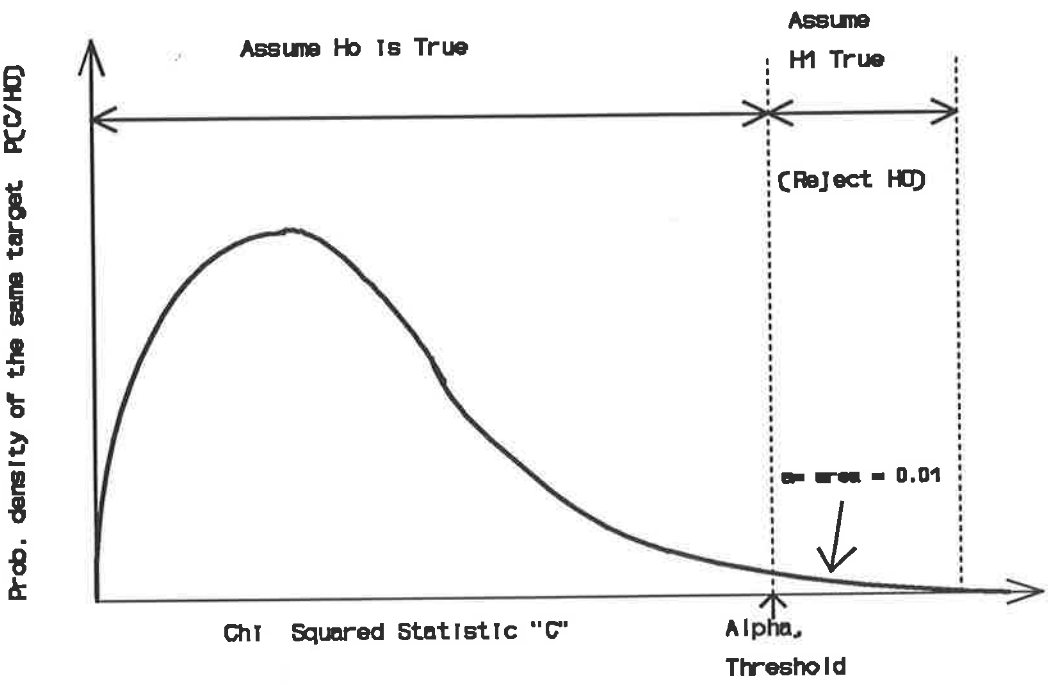
H0 : $E_{12}(K) = 0$ ---- SAME TARGETS
H1 : $E_{12}(K) \neq 0$ ---- DIFFERENT TARGETS

The test statistic used for the problem in section 4.0 is the Mahalanobis distance (which is a measure of similarity among vectors Y1 and Y2), calculated and summed for each time instant of the track. The hypothesis that the two targets are the same is accepted if the Mahalanobis distance is below a certain threshold obtained using the Chi-Squared distribution. More detail is provided in the next section (2.1.1.2).

We can establish the significance level for the decision rule as being $P_{FC} = 0.01$ (probability of false correlation). With P_{FC}=0.01, if H0 is true there is a 1% chance that the chi-squared statistic "c" (refer to Fig. 5) is above the threshold (or 1% chance of rejecting H0).

2.1.1.2 Track-to-Track Correlation and Fusion for Independent State Estimation Errors.

The Classical Inference techniques compute the probability of an observation given the assumption of an "a priori" hypothesis. Bar Shalom [6],[7] , outlined a technique based upon the use of the Chi-squared distribution of the difference in the state estimation vectors of Y1 and Y2.



H0: $E_{12} = 0$ same targets

H1: $E_{12} \neq 0$ diff. targets

If H0 is true there is a 1% chance that the chi-squared statistic "C" is above the threshold ie. 1% chance of rejecting H0.

Figure 5 Indicates the probability density that the targets are the same for Chi-squared test statistic "c". We are establishing the significance level with testing type 1 error.

Consider two tracks with state estimation vectors and covariance matrices of the estimates as follows.

- $\hat{Y}1(K)$, $\hat{Y}2(K)$, Are the estimated positions of a target from sensor 1 & 2, at time K.
- P1(K), P2(K), Are covariance matrices of estimates (assume independent errors)

Hypothesis Testing: Estimates pertaining to the same target ?

- $\hat{E}_{12}(K)=\hat{Y}1(K)- \hat{Y}2(K)$, the error estimates,

The above denotes estimate of:-

- $E_{12}(K)=Y1(K)-Y2(K)$, the true difference of position in multidimensional space.
Where $Y1(K)$ and $Y2(K)$ are the true states:

The problem of track association can be regarded as the following Hypothesis Testing problem:

$H0{:}E_{12}(K)=0$ ... same target.
$H1{:}E_{12}(K)\neq 0$ ... different targets.

The error in the difference between state estimates:

$\grave{E}_{12}(K)=E_{12}(K)-\hat{E}_{12}(K)$, is zero mean. If the state estimation errors

$\grave{Y}1(K)=Y1(K)-\hat{Y}1(K)$
$\grave{Y}2(K)=Y2(K)-\hat{Y}2(K)$,
are independent, then the covariance matrix $\Sigma_{12}(k)$ of $\grave{E}_{12}$ is:-

$$\begin{aligned}\Sigma_{12}(K)&=E\{\grave{E}_{12}(K).\grave{E}^T{}_{12}(K)\}\\&=E\{[\grave{Y}1(K)-\grave{Y}2(K)][\grave{Y}1(K)-\grave{Y}2(K)]^T\\&=P1(K)+P2(K)\end{aligned}$$

Assuming that errors from both radars 1 & 2 are independent you can add the covariance matrix estimates.

The Test Statistic is as follows:

$e\approx\hat{E}^T{}_{12}(K).\Sigma^{-1}{}_{12}(K).\hat{E}_{12}(K) < \alpha$ H0 is true
$e\approx\hat{E}^T{}_{12}(K).\Sigma^{-1}{}_{12}(K).\hat{E}_{12}(K) > \alpha$ H1 is true

Where "e" is called the Mahalanobis distance, and the threshold α is such :
Probability $(e>\alpha/H0)$ = P_{FC}=P(rejecting H0/H0 is true) = Type 1 Error= Level of Significance or Probability of False Correlation.

The P_{FC} may be set to 0.01 for instance (ie. 1% chance we will reject the hypothesis).
The threshold α can be selected by exploiting the fact that the random variable is Chi-squared distributed with N_X (dimension X) degrees of freedom. If H0 is accepted the track $\hat{Y}1(k)$ and $\hat{Y}2(k)$ can be combined:

$$\hat{Y}_c=P2.(P1+P2)^{-1}.\hat{Y}1+P1.(P1+P2)^{-1}.\hat{Y}2$$

The justification for this formulation is given in [6].

2.1.2 ART2 Neural Network

2.1.2.1 Art & Pattern Recognition

Neural networks are often used for solving pattern recognition problems. The pattern recognition is basically a signal processing problem. One of the advanced neural network structures proposed for pattern recognition is based on Adaptive Resonance Theory (ART). The ART neural network and its theory was proposed by Grossberg and Carpenter [9].

Art is used for pattern recognition. The network responds in real time to input vectors with stable, self- organised pattern recognition codes. Recognition occurs with the network matching invariant properties in the input pattern, with exemplars in a recognition category. The ART2 neural net uses unsupervised learning to develop pattern categories (ie the desired output need not be known to train the network). An additional vigilance parameter determines the degree of recognition between two objects.

There are generally two classes of ART architecture. ART1 is used for classifying binary input patterns, and ART2 for analogue patterns. ART1 illustrates many of the important aspects of ART2. ART2 was used in my application, to cluster tracks targets from radars with overlapping surveillance areas, because the output format from both radars is analogue. This will be discussed in more detail in the experimental chapters.

2.1.2.2 Operation of Art Network [9], [10], [11], [12], [13]

Art consists of two interconnected layers of neurons, F1 and F2, (as shown in Fig. 6), which comprises the attentional system. The input leads to activity in the feature detector neurons in F1 (this short term memory activity is represented by the shaded bars over the neurons, refer Fig. 7). This activity passes through connections (synapses) to the neurons in F2. Each F2 neuron adds together its input from all the F1 neurons and responds. Neurons in F2 compete with each other, so that at any instant, at most one neuron is active (winner take all).

The network gets organised through learning, in the following way. First consider a network that has already undergone some learning. In this case, the stimulus input leads to activity in the neurons of F1, the feature detecting neurons. Such activity represents short-term memory, since the neurons regularly relax back to their quiescent state after the stimulus is removed. The activity in the F1 neurons leads to activity through the interconnections and synapses to the neurons in F2.

The set of interconnections from F1 to F2 is called a filter; since it transforms the activity in F1 into a set of inputs to F2. The strengths of these interconnections are represented by the relative size of the half ovals as shown in Fig. 7. Each neuron of F2 represents a different category. eg. category for target 1,2,3 etc.

Early network architectures of Art were unstable when learning new categories. A particular neuron in F2 might at one instant represent a category A, whereas at a latter time (after more learning) it might represent a different category, category B. A network is of limited use if it cannot form stable categories.

New architectures of Art solves its instability problem in part through top-down priming, (also called attentional priming) as shown in Fig. 7 . Activity in the second F2 node reinforces the activity in the first and third neurons in F1 (Refer to Fig. 7). In general, each neuron in F1 is connected to every neuron in F2 by a bottom-up pathway, and each neuron in F2 is connected to every neuron in F1 by a top-down pathway.

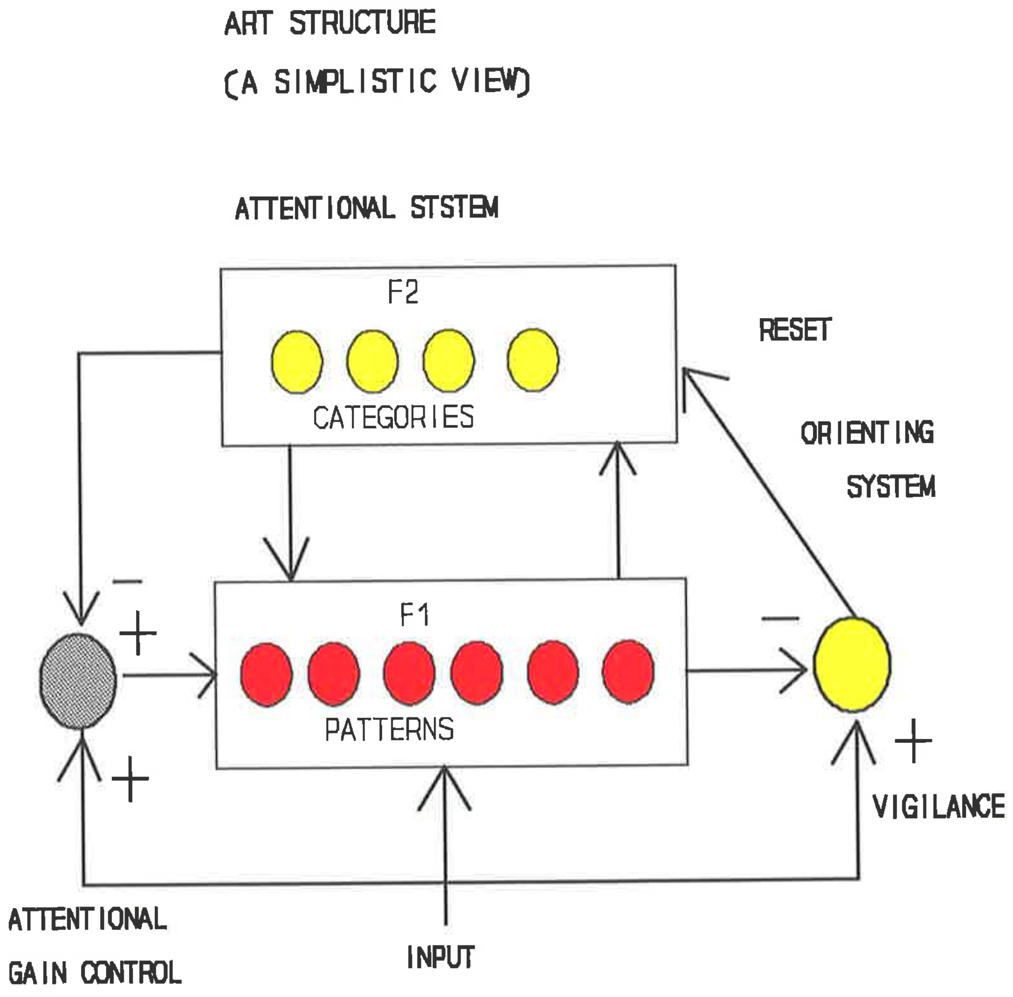


Figure 6 Art pattern classification network (Taken from [13].

The top-down signal represents a sort of template or set of critical features in the category. Any neuron in F1 might be receiving two activities, one due to the bottom-up input, the other due to the top-down priming. During recall and categorisation, the exchange of bottom-up and top-down information leads to a resonance in neural activity (resonance occurs due to exchange of bottom-up and top-down information). Critical features in F1 are reinforced, and have the greatest activity (refer to Fig. 7).

Activity in category neurons (in F2) leads via a top-down filter to activity in the neurons of F1.

2.1.2.3 Learning

In the previous paragraphs I have discussed the way input data leads to resonance between the input feature detectors and the categories. I will now describe how the network organised itself to give this behaviour. As mentioned earlier, the activity in the F1 and F2 neurons represents short-term memory; long term memory is encoded in the synaptic connections.

In Art, this long term memory is encoded in both the bottom-up and the top-down synaptic weights. Learning occurs when these synaptic weights are changed in response to the presentation of input patterns. The precise mathematical form of learning for Art2 is given latter. Briefly stated, a synapse in the top-down adaptive filter will approach a strength of 1.0 (in arbitrary units), if it links two active neurons (ie. one F1 neuron and one F2 neuron). If both neurons are inactive, it will remain unchanged. If the (pre-synaptic) F2 neuron is active but the (post-synaptic) F1 neuron is not, then the synaptic weight decays towards 0.0.

The learning rule for the bottom-up adaptive filter is similar to that just mentioned, except in the case where both neurons are active. In that case the synaptic weight increases, but not to a value 1.0. Instead the synapses at the post-synaptic neuron in F2 compete for resources, and thus reach a value dependent on the number of active neurons. The greater the number of active pre-synaptic neurons, the lower the asymptotic strength of each synapse that will be achieved. This learning is called Weber's Law. Such synaptic learning rules are non-Hebbian, by virtue of associative decay and synaptic competition. During learning, the synaptic strengths approach a composite or average of those which would be expected for each pattern presented independently. The composite pattern of synaptic strengths are weighted by the presentation statistics of those exemplars. Because learning requires that only one F2 neuron be active, the categories associated with inactive neurons are not degraded.

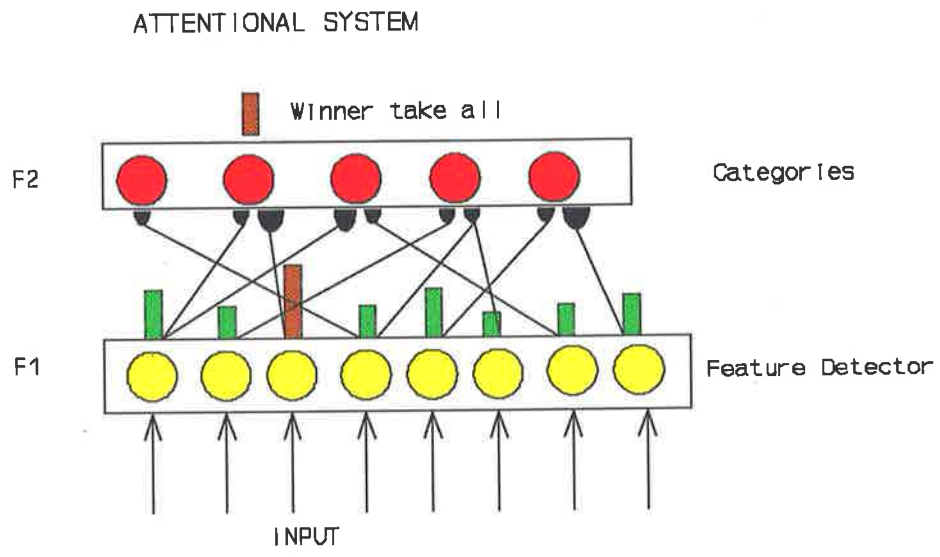


Figure 7 Activity in the category neuron (in F2) leads via a top-down Filter activity in neurons of F1.(Taken from [13]).

2.1.2.4 Gain Control

Art has an attentional gain control unit that prevents top-down synaptic signals alone from leading to F1 activity. This gain control system permits F1 to distinguish between top-down (from F2) and bottom-up (input) signals (refer to Fig. 8). As long as the input is present, the gain is high. If there is no stimulus input but just F2 is active, then the gain is low, and thus only a small activity can arise in F1.

The gain control system has 3 inputs and one output. The inputs are:-

(a) The stimulus input itself (excitatory)
(b) An activity signal from F2 (inhibitory)
(c) Intermodal inhibition.

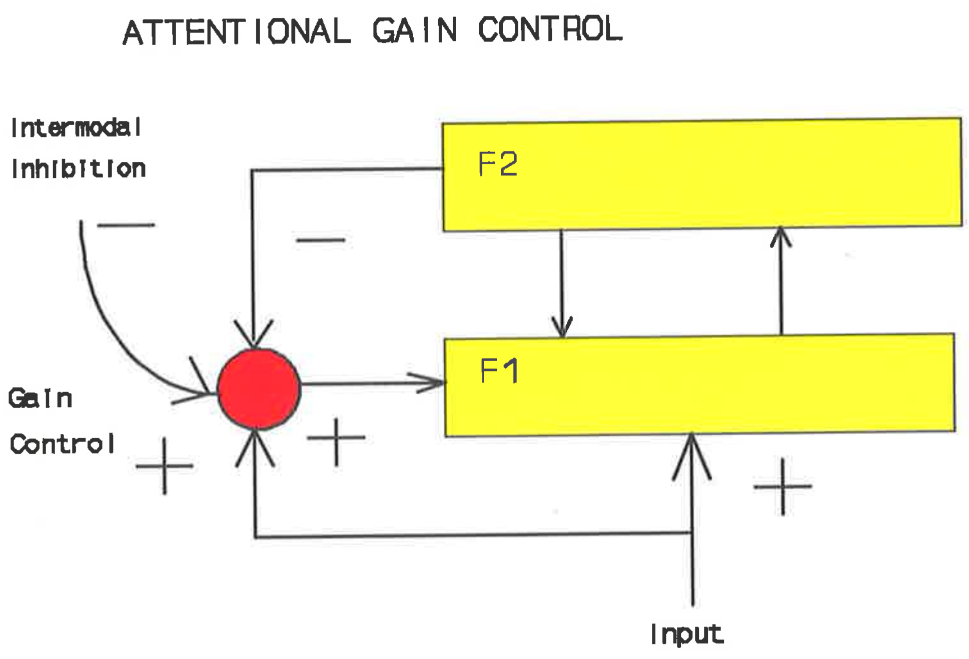


Figure 8 The attentional gain control unit permits F1 to distinguish between purely bottom-up and top-down signals.

The gain control output acts as an overall gain or amplification signal for F1 activity. If we begin with no inputs, but activity in an F2 neuron, the gain control has an uncentered inhibitory input (refer to Fig. 8), resulting in little or no output. Thus, gain in F1 is low. Even though there is attentional priming, the activity in F1 is small because of the low gain. Consequently, the pattern of activity in F1 is insufficient to lead to significant bottom-up activity and hence no resonance can result. But if there is an input stimulus, the gain control unit will be excited and the gain in F1 will be high. There will be activity in F1. (Note that regardless of the activity in F2, the system will shut down if there is no input).

To understand the intermodal inhibitory input to the gain control unit, suppose the Art network is applied to a dinner party example. At a lively dinner party, you are concentrating on what a new acquaintance is saying to you from across the table rather than on what you are eating or drinking. Even though you drink your wine, you might not taste it. This type of behaviour is duplicated in Art by having an intermodal inhibition signal to the gain control unit. In the dinner party example, your auditory system (attending to what your acquaintance is saying) decreases (inhibits) the gain control in your taste system. In general, intermodal competition helps to restrict the signal to the higher stages of cognition, thereby preventing cognitive overload.

2.1.2.5 Novelty Detector (Vigilance parameter) & Category Size

The novelty detector determines how fine or course a category will be (ie. how much different input patterns can vary and still be in the same category). The orientation system has two inputs and one output, (refer to Fig. 9). The 2 inputs are the data input itself and the overall activity in F1. The stimulus input is connected to the orienting subsystem with an excitatory connection. The F1 activity level is communicated to the orienting subsystem through inhibitory connections. The single output of the orienting system goes to F2 and is a reset wave. The orienting system acts as a novelty detector by sending a reset wave to F2 whenever the activity pattern in F1 caused by the input pattern differs significantly from that caused by the top-down readout of a category neuron.

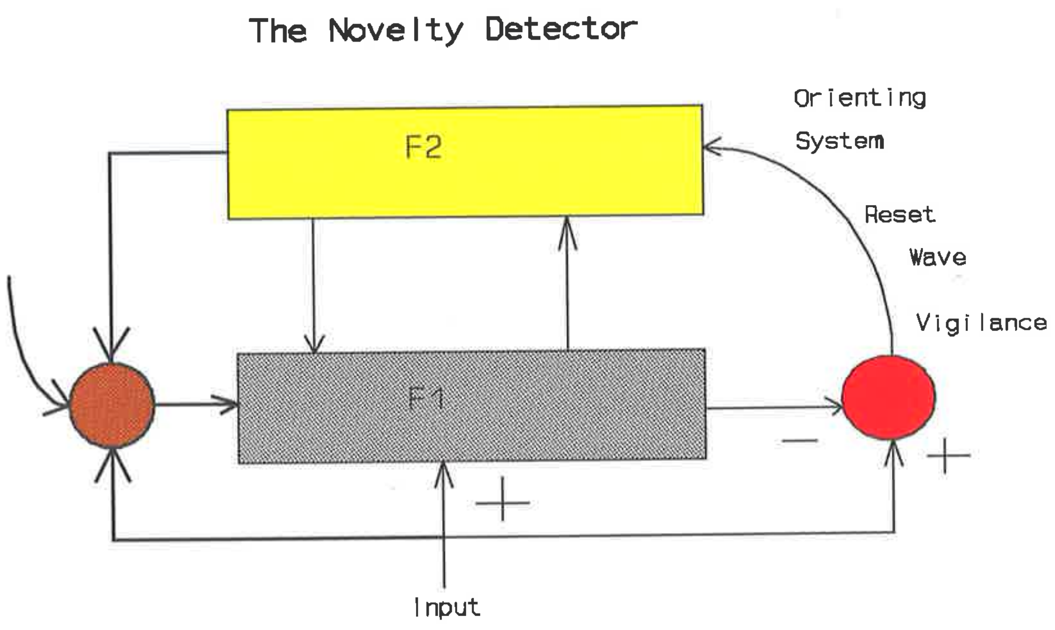


Figure 9 The orienting subsystem. (Taken from [13]).

The orienting subsystem sends its reset wave when the new input pattern differs significantly from any previously coded. The Art orienting system has a single parameter, called "vigilance parameter (v)" that tells how large a mismatch at F1 (between the top-down template and the bottom-up activity) can be tolerated before the orienting systems reset wave is emitted (refer to Fig. 9).

If "v" is large (high vigilance) only a very slight mismatch will be tolerated before a reset wave is emitted. If "v" is small(low vigilance), large mismatch will be tolerated before a reset wave (and subsequent new coding) will occur. Because category formation is dependent in this way on the relative similarity or difference in patterns, we say that Art has the property of self-scaling.

For instance, consider an example involving wine tasting. A novice wine taster may have only two categories: "good wine" and "bad wine". If the vigilance is set low, implying that subtle differences between wines is unimportant, then every new wine, regardless of its "features", will be classified as either good wine or as a bad wine. But suppose the vigilance is set high and a new wine is presented. The wine differs significantly from the "good wine" or "bad wine" categories. The Art network may test those categories, but because of the mismatch and high vigilance, it will ultimately recruit another category neuron in F2. In this way, for example you can generate a new category for "semi-sweet, fruity wine".

2.1.2.6 ART2 Network Equations & Structure Using HNC software [14]

The HNC (Hecht-Nielsen Neurocomputers) software implementation of ART2 is a 2 layer neural network with multiple slabs. The two layers (F1 and F2), or fields, are shown in Figures 6 and 7. The F1 field has been subdivided into 7 state vectors which represent the short- term memory (stm) of the F1 field. The neurosoftware has assigned a slab to each of these state vectors: the P, Q, R, U, V, W and the X slabs (refer to Fig. 10). The F1 field refers to all 7 of the slabs collectively. The F2 field is contained in the F2 slab.

Both F1 and F2 contain a state vector which represents the network's short term memory. The network's weights represent the long-term memory (ltm). They are maintained by the P slab and the F2 slab, and are applied at each connection between the 2 fields. Each processing element (PE) (or neuron) on the P slab is connected to each PE on the F2 slab, and each PE on the F2 slab is connected to each PE on the P slab.

Processing Equations:

Data is presented to the network through the input slab. When a pattern is presented to the network the states of the PE's of the F2 fields (or layer) are set to inactive, and the states of the "U" slab are set to zero. The F1 field is iterated until the states of the R slab change less than the tolerance parameter (defined later in real time parameter definitions).

The F2 field is then iterated to find an initial choice for the active F2 PE. Next, the F1 field is iterated until stability is reached. If the F2 choice is close enough to the input pattern the resonance state is achieved. If a mismatch occurs, the F2 PE is inhibited and the process is repeated until a match is achieved. If learning is enabled, the long term memory weights are updated. Figure 10, taken from reference [12] represents the calculations that take place within the F1 and F2 fields for short term memory processing. The large filled circles (inhibitory interneurons) in Fig. 10, referred to as "gain control nuclei" in ref. [12], (used for the normalisation of activation patterns across F1) nonspecifically inhibit target nodes (neurons) in proportion to the L2 norm of the short term memory activity in their source fields.

F1 Processing:

The F1 field is updated by the following equations.

(a) W slab - The output of the i^th^ W slab PE is given by

$$w_i = I_i + a.f(u_i)$$

Where I_i is the state of the ith. input slab PE, and u_i is the state of the ith "U" slab PE and

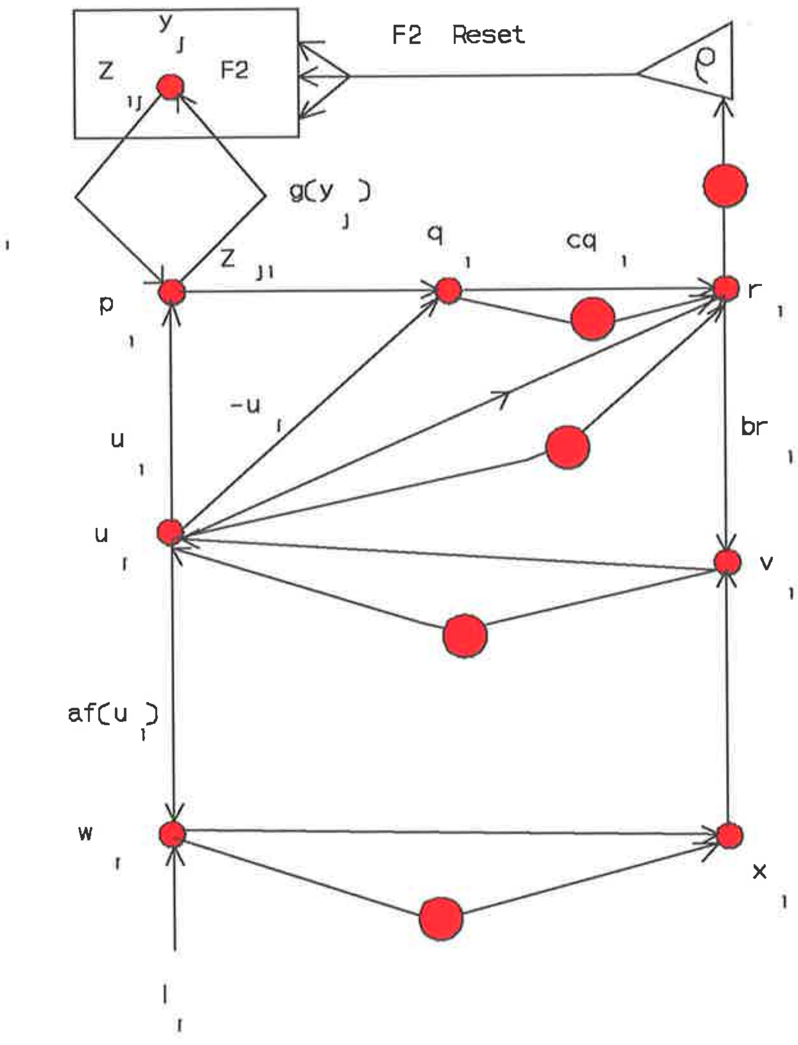


Figure 10 ART2 Architecture for F1 (Taken from [12]).

"*a*" (a positive gain term) is a user parameter.
The Signal Function is false in our application, so "*f*" is given by

$$f(x_i) = 0 \text{----} \textit{if}\ ,0 \leq x < \theta$$
$$\textit{else}$$
$$f(x_i) = x \text{----} \textit{if}\ ,x \geq \theta$$

where Θ (signal threshold) is a user-specific parameter equal to one over the square of the number of nodes (neurons) to be normalised.

(b) X slab - The output of the ith X slab PE is given by

$$x_i = \frac{w_i}{(e+\|w\|)}$$

where "*e*" (a positive small number used so that you cannot divide by zero in the above equation) is a user specific parameter and ‖ *w* ‖ is the L_2 norm of the w slab.

(c) P slab - The output state of the ith P slab PE is given by

$$p_i = u_i + \sum_{j=1}^{\infty} .(y_i).z_{ji}$$

where y_i is the state of the jth F2 slab PE, Z_{ji} is the weight associated with the connection between the ith P slab PE, and the jth F2 slab PE and the function "*g*" is given by

$g(y_i) = d$	*if the jth F2 PE is active*
$g(y_i) = 0$	*otherwise.*

where "*d*" (gain value which limits the maximum value of the winning neuron) is a user - specific parameter. Since the value of $g(y_i)$ is either "*0*" or "*d*", the output states for the P slab PE's reduces to

$$p_i = u_i + dz_{Ji}$$

where "*J*" is the index of the active F2 PE.

(d) Q slab - The output of the ith Q slab PE is given by

$$q_i = p_i - u_i$$

(e) R slab - The output slab of the ith "R" slab PE is given by

$$r_i = \frac{(u_i + c.q_i)}{(e+\|u\|+c.\|q\|)}$$

where ‖ *u* ‖ and ‖ *q* ‖ are the L_2 norms of the U and Q slabs.

(f) U slab - The output state of the ith U slab is given by

$$u_i=\frac{v_i}{(e+\|v\|)}$$

where $\| v \|$ is the L_2 norm of the V slab.

(g) V slab - The output state of the ith V slab is given by

$$v_i=f(x_i)+b.r_i$$

where x_i is the state of the ith X slab PE, "*b*" (a positive feedback gain term which amplifies r_i, used for stability) is a user parameter, and "*f*" is as defined in the W slab equations.

F2 Processing:

On the F2 slab, each PE calculates the dot product between the P slab vector and its weight vector. This weight vector represents the F1 to F2 bottom-up long-term memory values. Only F2 PE that are not in the inhibit state perform this calculation. The PE with the largest dot product is selected as the active PE. All other non-inhibited F2 PE's are set to inactive. The equations for these calculations are given by

$$d_j=\sum_{i=1}^{\infty} .p_i.z_{ij}$$

y_i = *inhibit*	*if the current value is inhibit*
y_i = *active*	*if d_j = max(d_i) for all i values*
y_i = *inactive*	*otherwise.*

After the active F2 PE is selected, the F1 field is iterated. Once the F1 field has stabilised, the following inequality is checked

$$\frac{\rho}{(e+\|r\|)}>1$$

where ρ is the vigilance parameter. The vigilance parameter is user-selected and lies between 0 and 1. This inequality is the reset condition for the F2 slab. If the reset condition is not met, resonance has occurred and the input pattern is categorised as belonging to the class associated with the active F2 PE. If the next condition is met, the active F2 PE is set to the inhibit state, the F1 field is cleared, and a new F2 PE is selected.

Learning:

If learning is enabled and resonance state is achieved, the network weights z_{ji} (top-down) and z_{ij} (bottom-up) are updated. Normal learning incrementally moves the weights towards their asymptotic values. This causes the weights to average over all examples of a category.
Weights change as follows:

$$z_{Ji}^{NEW}=z_{Ji}^{OLD}+\alpha.d.(1-d).[\frac{u_i}{1-d}-z_{Ji}^{OLD}]$$

and

$$z_{iJ}^{NEW}=z_{iJ}^{OLD}+\alpha.d.(1-d).[\frac{u_i}{(1-d)}-z_{iJ}^{OLD}]$$

where J is the index of the active F2 PE and α is the learning rate.

Run Time Parameter Definitions using HNC software: [14]

The Tolerance parameter controls the number of iterations needed to reach F1 short-term memory (STM) stability. Stability occurs on the F1 field STM calculation when the largest change in activity of an R slab PE is less than the tolerance. The Learn Rate parameter "α" controls the rate at which weights are modified.

Parameter "A" is the parameter in the F1 field STM calculations, a.
Parameter "B" is the parameter in the F1 field STM calculations, b.
Parameter "C" is the parameter in the F1 field STM calculations, c.
Parameter "D" is the parameter in the F1 field STM calculations, d.
Parameter "E" is the parameter in the F1 field STM calculations, e.
Parameter "T" is the threshold parameter, Θ, used in the F1 field signal function.
Parameter "V" is the vigilance parameter, ρ, used in determining when an F2 PE has reached resonance.

The HNC software package suggests the use of the following typical values for the network parameters (refer to Table 2), which were appropriate for my application, (except for the vigilance parameter which was adjusted as indicated in the experimental section).

Table 2 ART2 network parameters used in target track correlation problems (discussed in the experimental section).

Parameter	**Typical values**
Initial weight max.	0.01
Tolerance	0.00001
Parameter A	10.0
Parameter B	10.0
Parameter C	0.10
Parameter D	0.9
Parameter E	0.000001
Parameter J	0.05
Parameter V	0.995

Chapter 3

Principles of Attribute Data Fusion

3.1 Dempster-Shafer (Evidential Reasoning) Method [15], [5]

3.1.1 Background Introduction

Evidential Reasoning requires no prior distribution of the existence of threat types as does the Bayesian approach. The Dempster-Shafer (D-S) method is based on a model of human inferences; it utilises probability intervals and uncertainty intervals to determine the likelihood of a hypothesis based on mutual evidence. In addition, the D-S methodology computes a likelihood that any hypothesis is true. The D-S method can be used for representing and combining data in a multiple sensor (or knowledge source) fusion application.

Using this method we have the provision for representing incomplete or uncertain sensor measurements (ie. D-S is a way of representing exactly what is and is not known). Each sensor contributes information at its own level of detail. The evidential reasoning structure is general enough to utilise fully each sensor's information regardless of its form.

The evidential reasoning approach is more general than the Bayesian. A weakness of the Bayesian approach is the lack of convenient representation for ignorance or uncertainty. For example, a question arises concerning the representation of an uncertain prior distribution with the standard Bayesian approach. The evidential reasoning method handles this situation quite simply by allowing the assignments of a probability mass value directly to uncertainty. It also handles the problem of incomplete or uncertain sensor measurements. Sensor error can be conveniently represented by a probability mass assignment directly to uncertainty.

The implementation of Evidential Reasoning is illustrated with the following example. If sensors contribute information in the following form:
"Sensor 1 indicates that the target is one of the three possible types: T1,T2, or T3."
"Sensor 2 indicates that the target is type T1." However the certainty of this is only 90%. Because there is no evidence yet to support that the target is of type T4, this type is ignored in all subsequent processing and only relevant target types are considered.

The process of data fusion consists of finding the intersection of two sensor statements. For instance we know that the intersection of (T1 or T2 or T3) and (T1) is equal to (T1).

However, only a probability of 0.9 is assigned to this product, owing to the 90% confidence on the second sensor report. The remaining probability (0.1) is assigned to the disjunction or union (T1 or T2 or T3). These statements are stored directly in the computer in the form of assigned target sets and associated probabilities.

3.1.2 Implementation of Evidential Reasoning (Support and Plausibility)

The method of evidential reasoning assigns a probability mass $m(T_i)$ to any of the "n" propositions (ie. target types T1,T2 ... T_N), or to disjunctions of propositions. For example, a disjunction is the proposition that the target is of type T1 or T2 (denoted T1 v T2) and the mass assignments is denoted m(T1 v T2).

The representation of uncertainty is a mass assignment to the disjunction of all the original propositions and is denoted

$$m(\Theta) = m(T1 \text{ v } T2 \text{ v } \dots \text{ v } T_N)$$

Also mass can be assigned to the negation of a proposition. For example, the mass assigned to the negation of T1 (the target is not type T1) is denoted

$$m(\bar{T}1)=m(T2 \text{ v } T3 \text{ v } \dots \text{ v } T_N)$$

The sum of the probability masses mentioned must equal to unity.

The likelihood of a proposition "A" is represented as a subinterval [S(A), P(A)] of the unit interval [0,1] (which represents total ignorance). Referring to Fig. 11, S(A) represents the support for proposition "A" and sets a minimum value for its likelihood. P(A), on the other hand denotes the "plausibility" of proposition "A" and establishes a maximum likelihood. Support may be interpreted as the total positive effect a body of evidence has on a proposition, while plausibility represents the total extent to which a body of evidence fails to refute a proposition. Thus $P(A)=1-S(\bar{A})$, where the negation of "A" ($\bar{A}$) is all propositions which are not "A". The degree of uncertainty about the actual probability value for a proposition corresponds to the width of its interval (refer to Fig. 11)

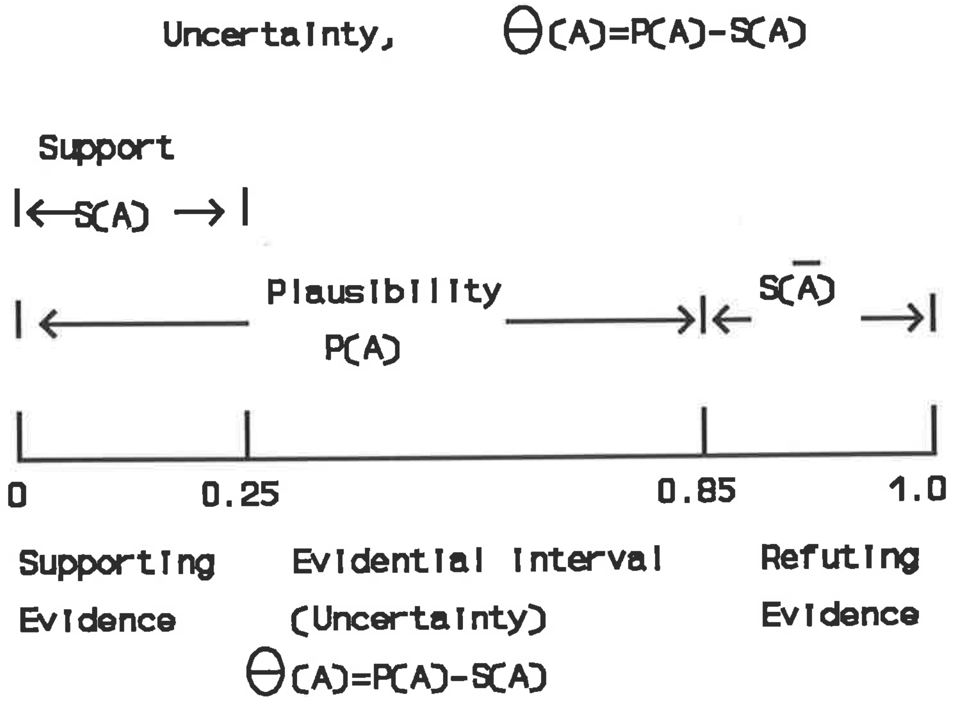


Figure 11 The Uncertainty Interval [S(A),P(A)]

Example:-

A[0.25,0.85] => The likelihood of "A" is between 0.25 and 0.85; the evidence simultaneously provides support for "A" and "$\bar{A}$ " (ie. S(A)=0.25, S($\bar{A}$)=0.15). Theta is 0.6.

To illustrate further, again consider the target type example; the support S(T1) for the basic proposition that the target is T1 is just the mass associated with T1(S(T1)=m(T1)). For a more complex proposition such that the target is either type T1, T2, or T3 we have

S(T1 v T2 v T3)= m(T1)+m(T2)+m(T3)+m(T1 v T2)+m(T1 v T3)+m(T2 v T3)+m(T1 v T2 v T3)

The plausibility of a given proposition is the sum of all mass not assigned to its negation. Thus

$P(T_i) = 1\text{-}S(\bar{T}_i)$

Alternatively, $P(T_i)$ can be computed by summing all masses associated with T_i and all disjunctions, including Θ, that contain T_i. For example

$P(T1) = m(T1) + m(T1 \vee T2) + ... + m(\Theta)$.

3.1.3 D-S Rules of Combination of Mass Assignments

D-S rules of combination with the probability mass assignments (of mass vectors m1 and m2 for the 2 sensors or knowledge sources to form the resulting mass vector) are as follows.

(a) The product of mass assignments to two propositions that are consistent leads to an assignment to another proposition contained within the two original propositions.
For example

$m1(T_i).m2(T_i) = m(T_i)$
$m1(T1 \vee T3).m2(T3 \vee T4) = m(T3)$

(b) Multiplying the mass assignments to uncertainty by the mass assignments to any other proposition leads to contribution to that proposition.

$m1(\Theta).m2(T3 \vee T4) = m(T3 \vee T4)$
or $m1(\Theta).m2(T2) = m(T2)$

(c) Multiplying uncertainty by uncertainty leads to a new assignment to uncertainty.

$m1(\Theta).m2(\Theta) = m(\Theta)$

(d) Inconsistency occurs, for example, when one knowledge source assigns mass to T2 (m1(T2)) while a second assigns mass to T1 (m2(T1)). The product of these mass values is assigned a measure of inconsistency, denoted "k", of the form:

$m1(T2).m2(T1) = k$

The following numerical example illustrates D-S rules of combination, and the manner in which inconsistency is handled.

Consider an example where there are four target aircraft types as defined:

T1 = friendly interceptor (fighter aircraft)
T2 = friendly bomber
T3 = hostile interceptor
T4 = hostile bomber

Assume that our first knowledge source indicates that the aircraft behaviour appears to be that of the class of interceptor. However, this information is not certain so that the following mass assignments are defined:

m1(Θ)=0.4
m1(T1 v T3)=0.6

The assignment of 0.4 to m1(Θ) represents the uncertainty that the aircraft is of the interceptor class.

The second knowledge source indicates that the target is probably hostile, but again this is not certain. Thus we assign to this knowledge source the following mass values:

m2(Θ)=0.3
m2(T3 v T4)=0.7

Table 3 Application of D-S rules of combination

m1(Θ) = 0.4	m(T3 v T4) = 0.28	m(Θ) = 0.12
m1 (T1 v T3) = 0.6	m(T3) = 0.42	m(T1 v T3) = 0.18
	m2(T3 v T4) = 0.7	**m2(Θ) = 0.3**

Table 3 illustrates how D-S rules were used to combine the two knowledge sources to produce the resulting masses below:

m12(Θ)=0.12
m12(T1 v T3)=0.18
m12(T3)=0.42
m12(T3 v T4)=0.28

The above example illustrated D-S's rule for the condition where there was no inconsistency (or assignment to a null hypothesis) between the knowledge sources. The manner in which inconsistency is handled is illustrated by introducing another hypothetical knowledge source.
Assume that a third knowledge source gives the following target type declaration:

m3(Θ)=0.2
m3(T1)=0.1
m3(T2)=0.2
m3(T3)=0.3
m3(T4)=0.2

Table 4 A second application of D-S's rule.

m3(Θ)=0.2	m(T3 v T4)= 0.056	m(T3)=0.084	m(T1 v T3)= 0.036	m(Θ)=0.024
m3(T1)=0.1	k=0.028	k=0.042	m(T1)=0.018	m(T1)=.012
m3(T2)=0.2	k=0.056	k=0.084	k=0.036	m(T2)=.024
m3(T3)=0.3	m(T3)=0.084	m(T3)=0.126	m(T3)=0.054	m(T3)=.036
m3(T4)=0.2	m(T4)=0.056	k=0.084	k=0.036	m(T4)=.024
	m12 (T3 v T4) = 0.28	**m12(T3) = 0.42**	**m12 (T1 v T3) = 0.18**	**m12(Θ) = 0.12**

Table 4 shows how D-S's rules is used to combine the previous masses "m12", obtained from Table 3 , with the third knowledge source masses "m3".

In order to compute the new masses, we first sum all assignments to k, which for this example leads to the value k=0.366. Then, the new masses are computed by summing the appropriate entries in the matrix and dividing by the normalisation factor (1-k=0.634). Thus the new values are:

$m(\Theta) = 0.024/0.634 = 0.038$
$m(T1) = (0.018+0.012)/0.634 = 0.047$
$m(T2) = (0.024/0.634) = 0.038$
$m(T3) = (0.084+0.084+0.126+0.054+0.036)/0.634 = 0.606$
$m(T4) = (0.056+0.024)/0.634 = 0.126$
$m(T1vT3) = 0.036/0.634 = 0.057$
$m(T3vT4) = 0.056/0.634 = 0.088$

3.2 Backpropagation Neural Network [16], [17], [19]

3.2.1 Background Introduction

An Artificial Neural Network is a distributed processing structure. Processing Elements (PE's) are its fundamental building blocks. They receive multiple input connections and generate, a single output, which may fan out to many other PE's. Processing Elements may have local memory and a transfer function which can use this memory. Links between the processing elements carry signals between them. Each connection may have a weight associated with it, for altering the strength of signals passing through (refer to Fig. 18). The artificial neural network variously known as the Gamba-perceptron, Multilayer Perceptron (MLP) and very loosely as the Backpropagation Network (BPN), accepts as input continuous-valued data, is able to learn complex distributions under supervision and is able to indicate a classification at its output.

Backpropagation is a learning rule for multilayer feedforward networks, in which the weights are adjusted by backward propagation of the error signal outputs to inputs. It uses supervised learning in which the network is presented with a set of input pattern target pairs. The network compares its output to the target and adapts itself according to the learning rules. Art2 discussed earlier uses unsupervised learning, ie. it adapts itself according to statistical association in the input pattern.

A typical backpropagation network always has an input layer and an output layer, and at least one hidden layer. There is no theoretical limit on the number of hidden layers but typically there will be one or two. Each layer is fully connected to the succeeding layer. The arrows in Fig. 18 (3 layer backpropagation neural network) indicate flow of information during recall. During learning, information is also propagated back through the network and used to update the connection weights. The output of each PE in the network is the sum of the weights multiplied by its inputs and its transfer-function as shown in Fig. 14.

3.2.2 How the Backpropagation Works [19,20,21]

Recognition Task: A common task is to divide the input space into several distinct regions (also called decision regions and domains, refer to Fig. 19). The job of a recognition system or classifier is to give outputs of 1 or 0 depending on whether an input vector $x'=(x_1,x_2 \ldots x_n)$ lies inside a domain (as shown in Figures 19).

Learning the Training Set: A backpropagation network learns a decision region by being exposed to many training examples. Each training example is known to be inside or outside the desired region (refer to Fig. 20).

Forming Decision Regions: A backpropagation network can identify vectors in any arbitrarily shaped decision region in the input space. It does this in two steps (refer to Fig. 12). Each processing element in the hidden layer (the first processing layer) divides the input space into two, along a plane. And each processing element in the output stage (second processing layer) combines one or more planes to form a convex open or closed region. A large number of training examples are usually required for a close approximation to the true boundary.

The types of decision regions (as shown in the second column of Fig. 13) that can be formed using a MLP with one, two and three layers that use hard limiting nonlinearities are illustrated in Fig. 13. The rightmost column gives examples of the most general decision region that can be formed. A single layer perceptron forms half-plane decision regions. A two layer perceptron can form any possible unbounded convex regions in the space spanned by the inputs. A three layer perceptron can form arbitrarily complex decision regions. The discussion of Fig. 13 is centred primarily on the multilayer perceptron with one output when hardlimiting non linearities are used. Similar behaviour is exhibited by the multilayer perceptron with multiple output nodes when sigmoid non linearities are used. The behaviour of these nets is more complex because decision regions are typically bounded by smooth curves instead of straight line segments.

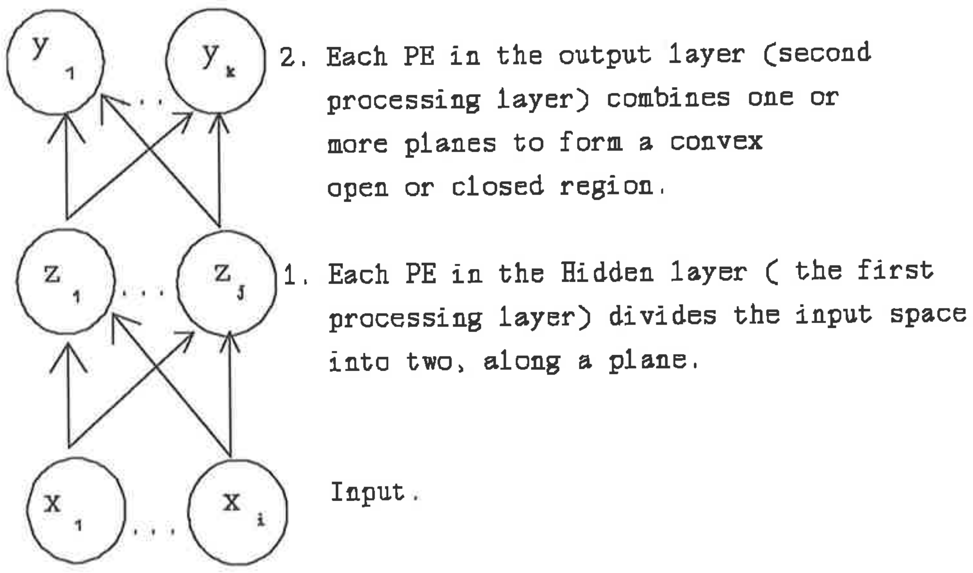

y_1
y_k
2. Each PE in the output layer (second processing layer) combines one or more planes to form a convex open or closed region.
z_1
z_j
1. Each PE in the Hidden layer (the first processing layer) divides the input space into two, along a plane.
x_1
x_i
Input.

Figure 12 Forming decision regions.

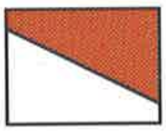
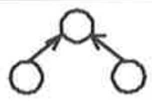
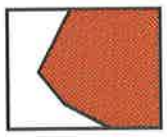
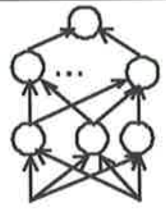
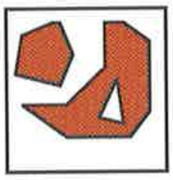
Number of Active Layers	Types of Descision Boundaries	Most General Region Shapes
1 layer	**Half plane bounded by hyperplane**	
 **2 layers**	**Convex open or closed regions**	
 **3 layers**	**Regions of arbitary complexity limited by the number of nodes**	

Figure 13 Types of decision regions that can be formed by single and multilayer perceptrons with one or two layers of hidden units and two inputs. Nodes in all nets use hard limiting nonlinearities.

The Role of Hidden Processing Elements:

A processing element performs the function defined by:

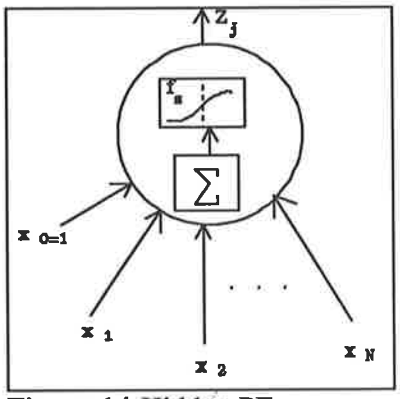


Figure 14 Hidden PE.

$$Z_j = f_s\left(\sum_{i=0}^{n} w_{ji} \cdot x_i\right)$$

Where, *fs* is the PE activation function (refer to fig. 17, ie. can be sigmoidal), w_{ji} is the weight on the connection joining the ith neuron to the jth neuron. To simplify the algorithm, the threshold offset is defined as w_0,(ie. = $w_0 \cdot x_0$, where x_0=1, a fixed value input).

The equation $z=w_1x_1+w_2x_2+ \ldots +w_nx_n+w_0$, represents a plane in $x_1,x_2 \ldots x_n$. (input space). The slope of the plane is determined by $w_1,w_2 \ldots w_n$, and the height at the origin (x' =0) is w_0. Assuming that the gradient is non-zero, there will be a region of space where z>0 and a region where z<0. The locus of points with z=0 is the decision boundary.

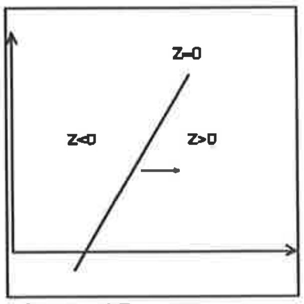


Figure 15

In two dimensions, the equation reduces to $z=w_1x_1+w_2x_2+w_0$ and the locus of points with z=0 is a line (ie. refer to Fig. 15). "Z" can be regarded as the height (or distance out of the page) of the plane at each point (x_1,x_2). The direction of the positive gradient is shown by the arrow (as shown in Fig. 15).

The Role of the Output Processing Elements:

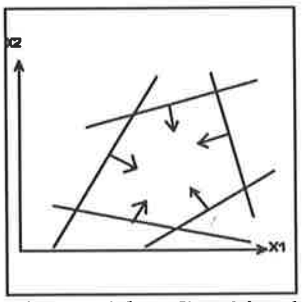


Figure 16 Combined planes form region.

Each output processing element receives its input from the hidden processing elements, which indicate whether the input vector lies on the high or the low side of each plane. By making a weighted sum of this information, an output processing element can combine the planes to form a region of input vectors (as shown in Fig. 16).

The Role of the Sigmoid Function:

The three types of activation functions possible for use in the backpropagation processing elements are hardlimiting, piecewise linear and sigmoidal (Fig. 17). Hardlimiting is discontinuous an non-biological. Piecewise linear is continuous but not differentiable. The sigmoid is continuous, monotonic and differentiable. The sigmoid function and its derivative are shown below. Note that $f_s'(x)$ can be evaluated from $f_s(x)$ alone.

$$f_s(x) = \frac{1}{(1+e^{-x})} \text{------}\textit{sigmoid}$$

$$f_s'(x) = \frac{e^{-x}}{(1+e^{-x})^2} \text{-----}\textit{derivative}$$

$$f'(x) = f(x).(1-f(x))$$

The sigmoid bounds the output of the hidden processing elements to be between 0 and 1.0. On a sigmoidal "plane", the locus of points with z=0 is still a line. However, when several sigmoids are added together, the locus of points with z=0 becomes a curved approximation to the planar boundaries. The rate of curvature is determined by the slope of the planes. A steep sigmoid begins to approximate a hard limiter, and the result is straight lines with a sharp corner between them.

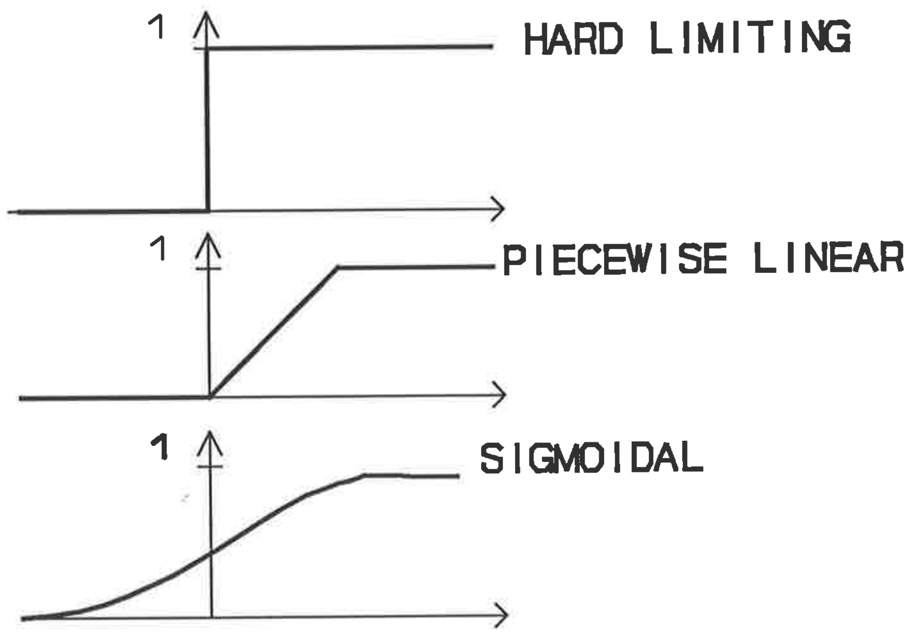


Figure 17 The 3 possible backpropagation P.E. activation functions.

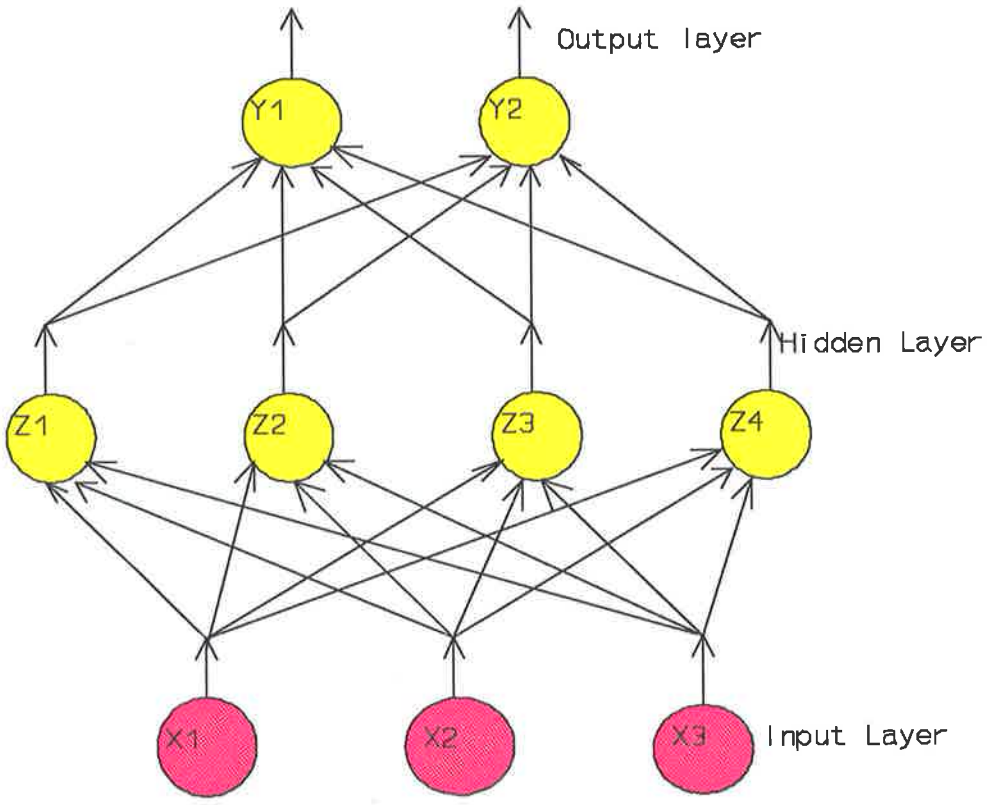


Fig. 18 A standard backpropagation network comprises of 3 layers of processing elements, each PE is connected to all PE's in the following layer and from all PE's in the preceding layer.

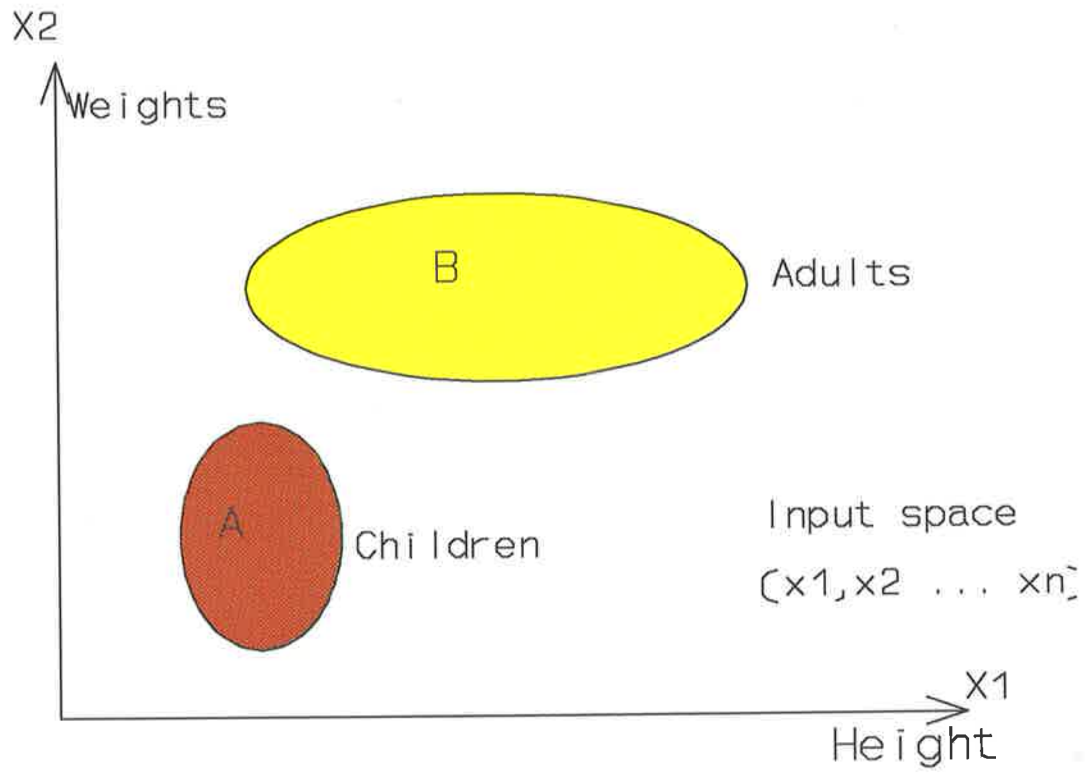


Figure 19 The backpropagation divides the input space (ie. height vs weight) into several distinct input classes (ie. children & adults).

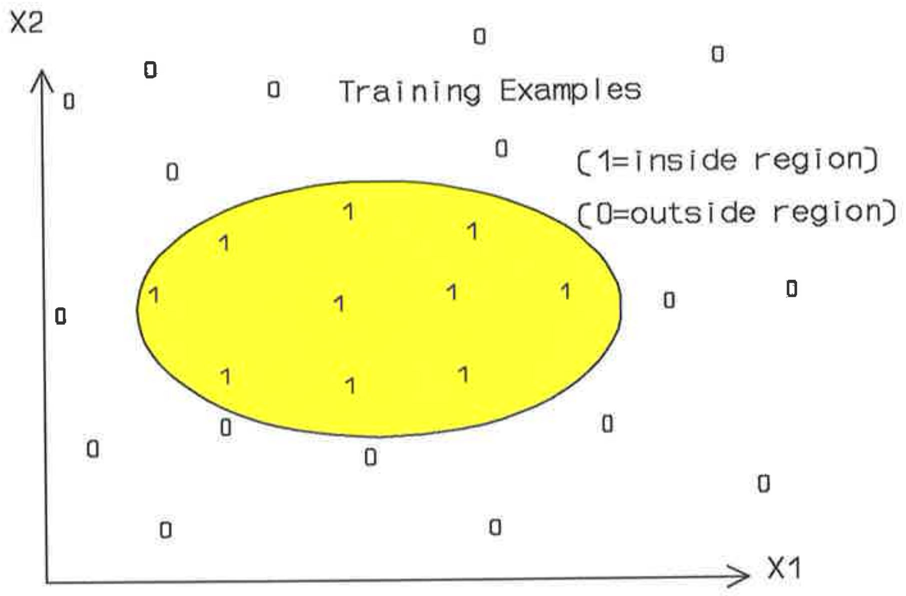


Figure 20 Each training example the backpropagation is given is known to be inside or outside the desired region.

Training Error: If a training input pattern "*x*" is presented with its corresponding training output "*t*" . The network can process "*x*" to give an output "*y*" (which is likely to differ from "*t*"). Given the actual output y_k of the kth processing element in the output layer, the error at that processing element will be

$$e_k=(t_k-y_k)$$

and the total squared error in the output layer, for this training pattern, is defined as

$$Er=\sum_k e_k^2$$

$$=\sum_k (t_k-y_k)^2$$

The mean square error (MSE) is the average "*Er*" (individual error) for all training pairs. The MSE is the function to be minimised, however, this requires evaluation of *Er* (individual error) for each pattern on the training set. It is simpler to minimise "*Er*" for each training pattern, one at a time. Large numbers of "*Er*", make a statistical approximation to the mean square error (MSE). Gradient descent is a method of finding the location of a minimum in a function of many dimensions. It locates a minimum by taking iteration steps down this gradient until the iterations converge to a single point.

3.2.3 Backpropagation Network Error Equations (Learning Rule) [13]

As mentioned in previous paragraphs, during learning, information is propagated back through the network and used to update the connection weights.

I will use the upper superscript in square brackets to indicate which layer of the network is being considered. The rest of the notations are as follows (Fig. 21 displays a backpropagation PE using this notation)

$x_j^{[s]}$ - Is the current output state of the jth neuron in layer "s".
$w_{ji}^{[s]}$ - Is the weight on the connection joining the ith neuron in layer (s-1) to the jth neuron in layer "s".
$j^{[s]}$ - Is the weighted summation of inputs to the jth neuron in layer "s".

A backpropagation processing element therefore transfers its input as follows:
Where "*f*" is the sigmoid function (also as shown in Fig. 21). The sigmoid is defined as

$$x_j^{[s]} = f\left(\sum_{i=1}^{\infty} (w_{ji}^{[s]} \cdot x_i^{[s-1]}\right) \text{------}(1)$$

$$= f(R_j^{[s]})$$

$$f(z) = \frac{1}{(1+e^{-z})} \text{-----}(2)$$

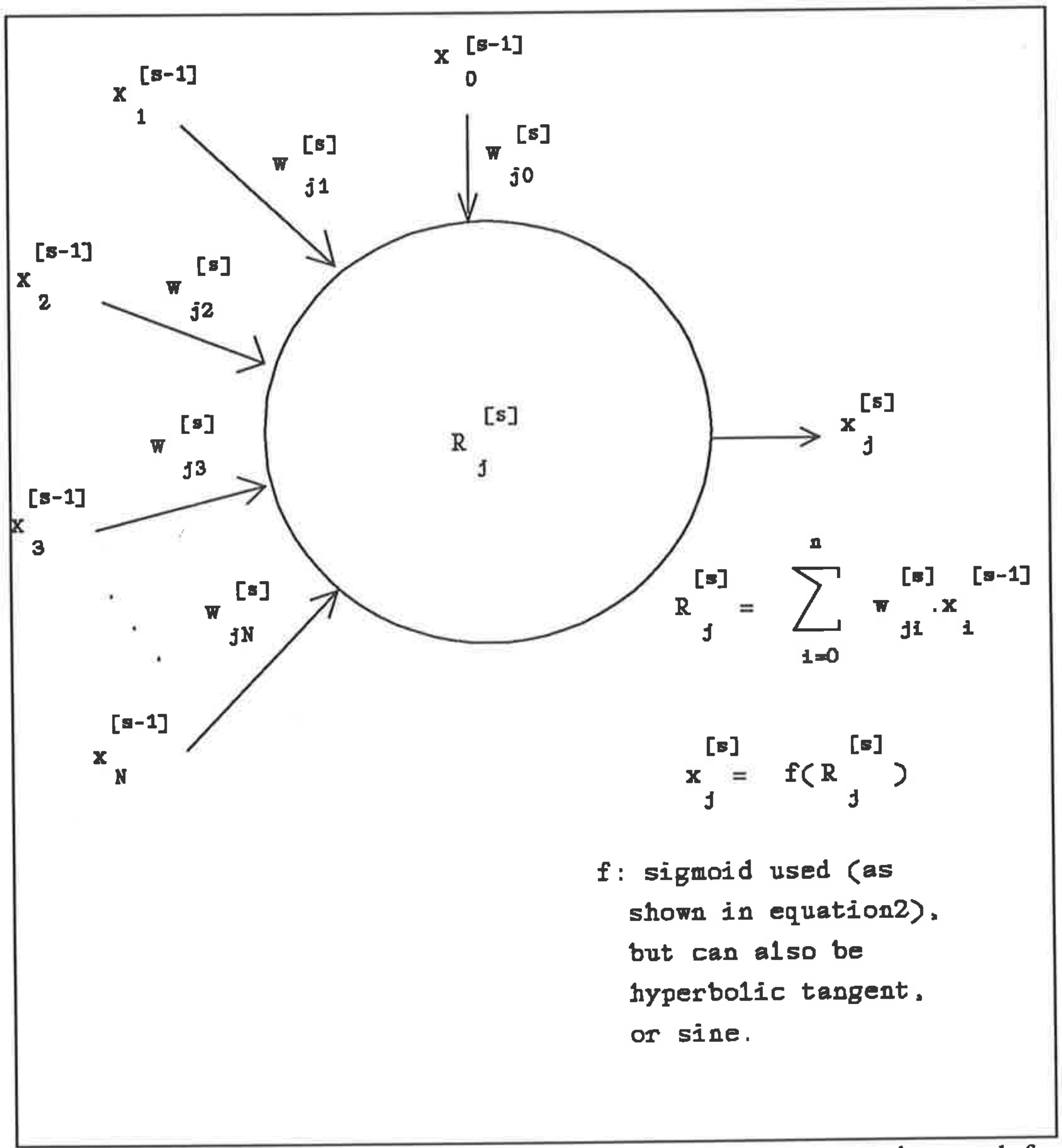

$x_0^{[s-1]}$
$x_1^{[s-1]}$
$w_{j0}^{[s]}$
$w_{j1}^{[s]}$
$x_2^{[s-1]}$
$w_{j2}^{[s]}$
$w_{j3}^{[s]}$
$R_j^{[s]}$
$x_j^{[s]}$
$x_3^{[s-1]}$
$w_{jN}^{[s]}$
$R_j^{[s]} = \sum_{i=0}^{n} w_{ji}^{[s]} . x_i^{[s-1]}$
$x_N^{[s-1]}$
$x_j^{[s]} = f(R_j^{[s]})$
f: sigmoid used (as shown in equation2), but can also be hyperbolic tangent, or sine.

Figure 21 Typical back-propagation processing element (shows notation used for equations (1) - (10), Taken from [13]).

3.2.3.1 The Local Error

Suppose the network has some global error function "E" associated with it which is a differentiable function of all the connection weights in the network. The parameter that is passed back through the layer is

$$e_j^{[s]} = -\frac{\delta E}{\delta R_j^{[s]}} \text{----}(3)$$

The following relationship between the local error at a particular processing element at level "s" and all the local errors at the level "s+1", is obtained using the chain rule twice in succession.

$$e_j^{[s]} = f'[R^{j^{[s]}}].\sum_{k=1}^{\infty}(e_k^{[s+1]}.w_{kj}^{[s+1]}) \text{----}(4)$$

In Equation 4, there is a layer above layer "s"; therefore, equation 4 can only be used for non-output layers.
If "f" is the sigmoid function as defined in equation 2, then its derivative can be expressed as a simple function of itself as follows

$$f'(z) = f(z).(1.0 - f(z)) \text{-------}(5)$$

Therefore, from equation 1, equation 4 can be rewritten as

$$e_j^{[s]} = x_j^{[s]}.(1.0 - x_j^{[s]}).\sum_{k=1}^{\infty}(e_k^{[s+1]}.w_{kj}^{[s+1]}) \text{----}(6)$$

provided the transfer function is a sigmoid. The summation term in equation 6, which is used to back-propagate errors is analogous to the summation term in equation 1 which is used to forward propagate the input through the network. Thus the main mechanism in a backpropagation network is to forward propagate the input through the layers to the output layer, determine the error at the output layer, and then propagate the errors back through the network from the output layer to the input layer using equation 6 or more generally equation 4. The multiplication of the error by the derivative of the transfer function scales the error.

3.2.3.2 Minimising the Global Error

Based on knowledge of the local error at each processing element, the aim of the learning process is to minimise the global error "*E*", of the system by modifying the weights.

Given the current set of weights $w_{ij}^{[s]}$ we use the gradient descent rule to increment or decrement them in order to decrease the global error

$$\Delta w_{ji}^{[s]} = -\alpha.(\frac{\delta E}{\delta w_{ji}^{[s]}}) \text{----}(7)$$

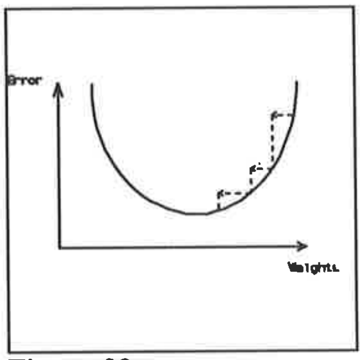


Figure 22

where α is the learning coefficient (ie. each weight is changed according to the size and direction of the negative gradient on the error surface, refer to Fig. 22).

The partial derivative in equation 7 can be calculated directly from the local error values discussed in the previous sub-section, by the chain rule and equation 1

$$\frac{\delta E}{\delta w_{ji}^{[s]}} = \frac{\delta E}{\delta R_j^{[s]}} . \frac{\delta R_j^{[s]}}{\delta w_{ji}^{[s]}} \text{----}(8)$$

$$= e_j^{[s]} . x_i^{[s-1]}$$

Combining equations 7 and 8 together gives

$$\Delta w_{ji}^{[s]} = \alpha . e_j^{[s]} . x_i^{[s-1]} \text{-----}(9)$$

3.2.3.3 Global Error Function

The global error function is needed to define the local errors at the output layer so that they can be propagated back through the network. Suppose "*i*" vector is presented at the input edge layer of the network, and suppose the desired output "*d*" is specified by a teacher. Let "*o*" denote the actual output produced by the network with its current set of weights. Then a measure of the error in achieving that desired output is given by

$$E=0.5\sum_{k=0}^{\infty}((d_k-o_k)^2)\text{-----}(10)$$

where the subscript "*k*" indexes the components of "*d*" and "*o*". Here, the raw local error is given by d_k-o_k. From equation 3, the scaled "local error" at each processing element of the output layer is given by

$$e_k^{(0)}=-\frac{\delta E}{\delta I_k^{(o)}}\text{-----}(11)$$

$$=-\frac{\delta E}{\delta o_k}.\frac{\delta o_k}{\delta I_k}$$

$$=(d_k-o_k).f'(I_K)$$

$$=(d_k-o_k).x_k^{[s]}.(1.0-x_k^{[s]})$$

Equation 10 "*E*" defines the global error of the network for a particular (*i,d*). The overall global error function is the sum of all the pattern specific error functions. Then each time a particular (*i,d*) is shown, the backpropagation algorithm modifies the weights to reduce that particular component of the overall error function.

3.3 Fuzzy Reasoning [22,23]

Fuzzy set theory was developed by Zadeh [24], [25]. The basic concept is that people frequently deal with concepts that are imprecise because of indistinct boundaries of definitions (ie. terms such as tall, short, attractive, ugly are imprecise). These imprecision can be addressed mathematically via an extension of Boolean set theory.

Fuzzy sets are defined as follows: A set "A" has members X_0, X_1, ... X_N. Each element X_1, of set "A" has an associated value $\mu A(X_1)$, which indicates the degree to which X_1 belongs to set "A". the function is called the "membership function, $\mu(X)$", and has a value between "0" and "1" with $\mu A(X)=0$ indicating that element "X" is not a member of set "A", and $\mu A(X)=1$ indicating that element "X" is completely a member of set "A". The values of $\mu(X)$, within this range, must be provided by the person defining the fuzzy sets. Hence, in fuzzy set theory, sets are defined by ordered pairs, [X, $\mu(X)$] in which "X" is an identified set element, and $\mu(X)$ is the associated membership value element "X". By contrast, Boolean sets are defined by identifying the elements that completely belong to a set (hence $\mu(X)$ is either 1 or 0).

Truth values (in fuzzy logic) or membership values (in fuzzy sets) are indicated by a value in the range [0.0, 1.0], with 0.0 representing absolute Falseness and 1.0 representing absolute Truth. For example, let us make the statement, "Jane is old". If Jane's age was 75 we might assign to the statement the truth value of 0.8. The statement could be translated as "Jane is a member of the set of old people", or using fuzzy set symbolically μOLD(Jane)=0.8. Where "μ" is the membership function, operating in this case on a fuzzy set of old people, which returns a value between 0.0 and 1.0. The probabilistic approach yields the statement, "There is an 80% chance Jane is old", while the fuzzy terminology corresponds to "Jane's degree of membership within the set of old people is 0.8". The differences are significant: the first view supposes that Jane is not old; it is just that we have an 80% chance of knowing which set she is in. By contrast, fuzzy terminology supposes that Jane is "more or less" old, or corresponding to the value of 0.8. The statement could be translated as "Jane is a new member of the set of old people", or using fuzzy symbolically μOLD(Jane)=0.8. Where "μ" is the membership function, operating in this case on a fuzzy set of old people, which returns a value between 0.0 and 1.0. The probabilistic approach yields the statement, " There is an 80% chance Jane is old " , while the fuzzy terminology correspond to "Jane's degree of membership within the set of old people is 0.8." The differences are significant: the first view supposes that Jane is not old; it is just that we have an 80% chance of knowing which she is in. By contrast, fuzzy terminology supposes that Jane is "more or less " old, or corresponding to the value of 0.8.

Fuzzy logic is also defined by the operations of Empty, Equal, Complement (Not), Containment, Union (Or), and Intersection (And), and the following formal definitions:-

Definition 1: **Let "X" be some set of objects, with elements noted as "x". Thus X = {x}.**

Definition 2: **A fuzzy set "A" in "X" is characterised by a membership function μA(x) which maps each point in "x" onto a real interval [0.0, 1.0]. As μA(x) approaches 1.0, the "grade of membership in A increases".**

Definition 3: **"A" is Empty if for all "x" , μA(x) = 0.0.**

Definition 4: **A=B if for all "x": μA(x) = μB(x) (or, μA = μB).**

Definition 5: **μA' = 1-μA.**

Definition 6: **"A" is Contained in "B" if μA ≤ μB.**

Definition 7: **C = A Union B, where: μC(x) = Max(μA(x), μB(x)).**

Definition 8: **C = A Intersection B, where: μC(x) = Min(μA(x), μB(x)).**

Note the last two operations, Union (Or) and Intersection (And), represent the clearest point of departure from probabilistic theory for sets to fuzzy sets. Operationally, the differences are as follows:

For example, let us assume x= Bob, "S" is the fuzzy set of smart people and T is the fuzzy set of tall people. Then, if μS(x) = 0.9 and μT(x) = 0.8, the probabilistic approach result would be:

μS(x) . μT(x) = 0.72

whereas the fuzzy result of would be:

Min(μS(x), μT(x)) = 0.80

The real value of the fuzzy set theory (developed by Zadch [24], [25]) to data fusion is the extension to fuzzy logic (as described above in definitions 1-8); because of this we classify this method in the " cognitive " group. Fuzzy logic deals with approximate modes of reasoning. In classical two-valued logic, a proposition, p, is either true or false. Classical logic uses truth tables and manipulative rules to follow a chain of reasoning to determine the truth (or falseness) of a proposition. By contrast, in fuzzy logic, a proposition has a membership value range from 0 (completely false) to 1.0 (completely true), representing membership of proposition in the truth value set.

Chapter 4

Experimental Procedure and Results for Kinematic Data-Fusion

4.1 Introduction

Sensor data fusion involves the entire process of correlation and fusion. In chapter 4 the thesis addresses the application of artificial neural networks (ANN), to the correlation problem of sensor level tracking. A comparison has been made between two track-to-track correlation techniques for multisensor fusion using simulated and real track data obtained from Adelaide Airport's Surveillance Radar and the FPS-16 Tracking Radar at DSTO.

Unlike the Central Level Tracking approach which forms tracks from raw observations the Sensor level tracking approach forms sensor tracks and then combines them. The advantages Sensor Level Tracking has over Central Level Tracking are:- decreased transfer load, the computational advantages of parallel processing, and decreased vulnerability since each tracking system has the ability to track independently [26]. The problem of Track-to-Track correlation arises when multiple sensors report tracks from a common surveillance volume. An important question is how to decide whether two tracks from two different sensors (using Sensor Level Tracking) represent the same target.

4.2 Simulated Track Data Results

The two track-to-track correlation techniques (Classical Inference and ART2 neural network) were used and compared to each other using the simulated scenario described below.

Consider two independent radars (since tracks are independent, the covariance matrix is the sum of covariance matrices P1 and P2 from Radars 1 and 2 respectively), Radar1 and Radar2 using a sensor level tracking approach to form sensor tracks. Both radars output data in the format of Range (metres), Azimuth (degrees), and Elevation (degrees). They have different accuracies with known variances in Range, Azimuth, and Elevation and corresponding covariance matrices. Both radars are tracking 3 targets each. Two of these targets are in the common surveillance volume of both radars (refer to Figures 23 & 24). Radar1 and Radar2 report 3 tracks each over a time period of ten seconds. Assume tracks are aligned in space over the time period.

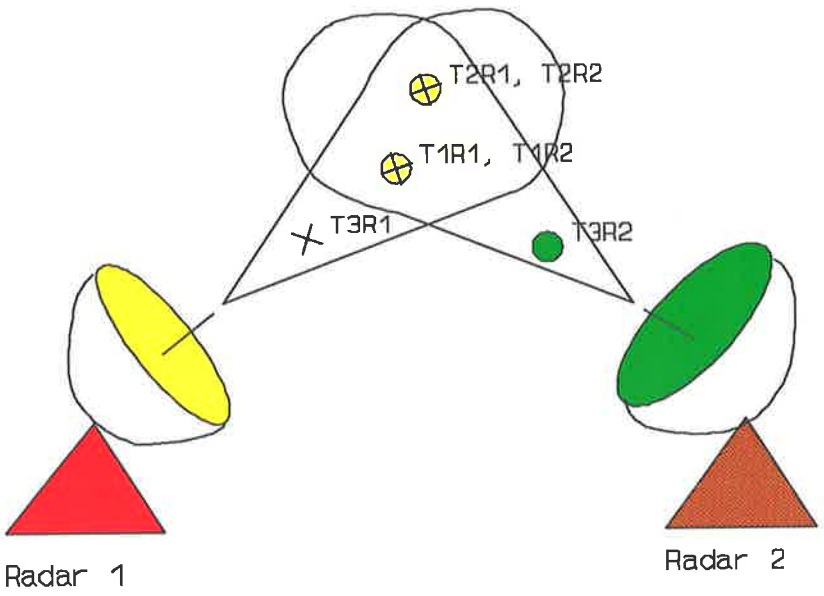


Figure 23 Radar1 and Radar2 are tracking three targets each. Two of these targets are in the common surveillance volume of both radars.

Assume the following variances (as shown in the simulation program in Appendix C) in Range(metres), Azimuth(degrees), Elevation(degrees) for the two independent Radars (whose accuracy differs in Range, Azimuth and Elevation.) ie.

Radar 1 -	Standard deviation in Range(m) =	1
	" " " Azimuth(deg)=	2
	" " " Elevation(deg)=	3
Radar 2 -	Standard deviation in Range(m) =	3
	" " " Azimuth(deg)=	4
	" " " Elevation(deg)=	1

Assuming independent Radars the corresponding covariance matrices for Radar 1 and Radar 2 are:-

$$P1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 9 \end{bmatrix}, \quad P2 = \begin{bmatrix} 9 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We notice that the accuracy of Radar 1 is better than that of Radar 2. Since both radars

R1 and R2 tracked three targets each, we have nine possible track-to-track combinations. ie.

T1R1T1R2 - (Track1 from radar1 and track1 from radar2)
T1R1T2R2 - (Track1 from radar1 and track2 from radar2)
T1R1T3R2 - (Track1 from radar1 and track3 from radar2)
T2R1T1R2 - (Track2 from radar1 and track1 from radar2)
T2R1T2R2 - (Track2 from radar1 and track2 from radar2)
T2R1T3R2 - (Track2 from radar1 and track3 from radar2)
T3R1T1R2 - (Track3 from radar1 and track1 from radar2)
T3R1T2R2 - (Track3 from radar1 and track2 from radar2)
T3R1T3R2 - (Track3 from radar1 and track3 from radar2)

As shown in Fig. 23 two targets are in the common surveillance of both radars, ie. Track-to-track combination T1R1T1R2 and T2R1T2R2.

Using Matlab I simulated the tracks (as shown in Appendix C, refer to Fig. 24 for the plots) with the added noise with standard deviations of 1,2,3 and 3,4,1 in range(m), azimuth(deg.) and elevation(deg.) for radar1 and radar 2 respectively .The Matlab function "rand", which is a gaussian random number generator was used; ie. I assumed that the noise errors from radars are gaussian.

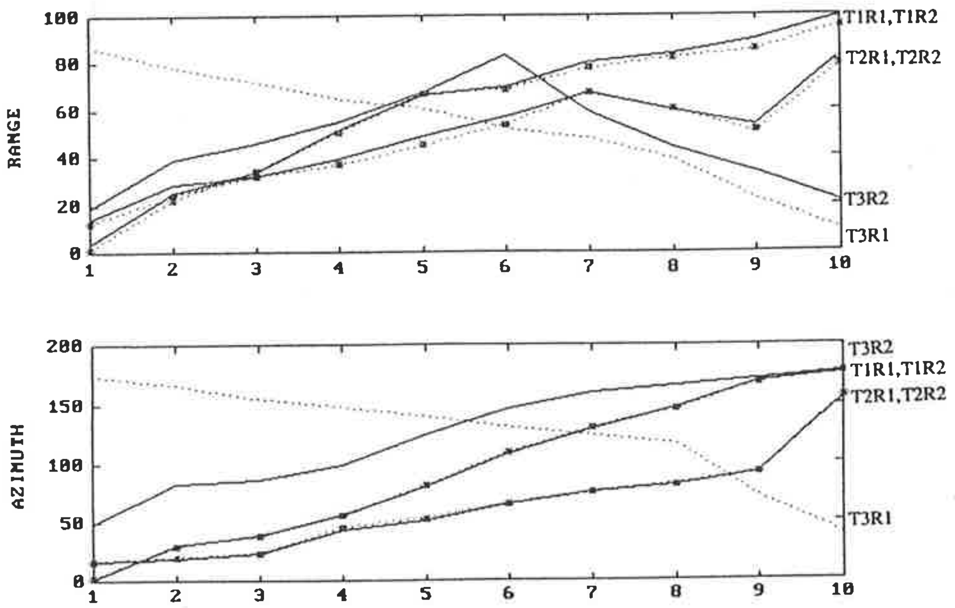


Fig. 24 Displays Azimuth (degrees) and Range (metres) versus Time (seconds) for example targets. Radar1 and Radar2 report three tracks each (T1R1,T2R1,T2R1 and T1R2,T2R2,T3R2). The six tracks above (Range and Azimuth vs Time) are displayed in a common space/time co-ordinate graph (elevation not displayed). The dotted plots represent target tracks from Radar1 and the solid plots from Radar2. The asterisk represents the fused plot.

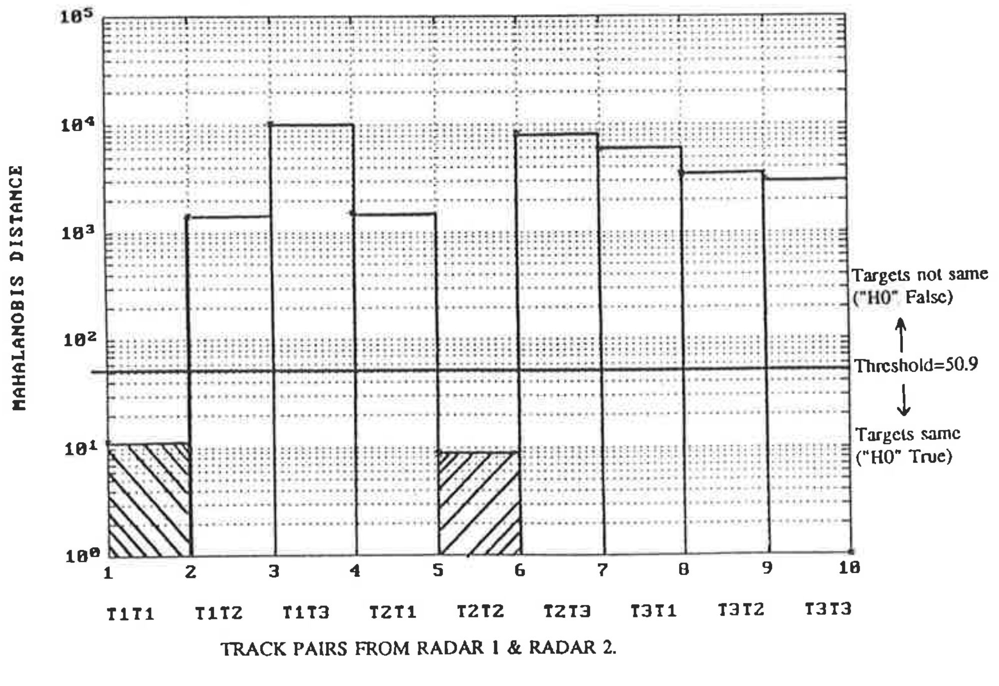


Fig. 25 Displays the Classical Inference results with the Mahalanobis distance (refer to section 2.1.1.1 & 2.1.1.2) between pairs of tracks from Radar1 and Radar2, together with the Level of Significance (threshold value).

4.2.1 Using Classical Inference on the Simulated Track Data

One wants to test the hypothesis that the 2 estimates *Y1(K)* and *Y2(K)* correspond to the same target for the nine track pair combinations from radar1 and 2, (refer to the section 2.1.1 on Classical Inference using hypothesis testing)
ie.

Y1(K) = 3X1 Vector Range1, Azimuth1 and Elevation1
Y2(K) = 3X1 Vector Range2, Azimuth2 and Elevation2.

The Mahalanobis distance was calculated and summed over the 10 second time period for each of the 9 track pair combinations from both radars. Two of the track pairs were found to be below the threshold value of 50.9 (determined as explained latter),(Refer to Fig. 25) which was obtained using the Chi-Squared distribution for 3x10 degrees of freedom ,thus indicating the same targets from both radars.

An example calculation of the Mahalanobis distance for simulated tracks T1R1 and T1R2 (shown in Table 5.) is given below:-

The Mahalanobis distance is defined as follows (Refer to review chapt. 2.1.1):-

$$e=E_{12}^{T}(k).\Sigma_{12}^{-1}.E_{12}(k) \ ,$$

$$where \ , \ E_{12}=Y1(k)-Y2(k)$$

$$vectors \quad Y1(k), \ Y2(k) \ are \begin{bmatrix} RANGE \\ AZIMUTH \\ ELEVATION \end{bmatrix} for \ radar1{,}2.$$

$$\Sigma_{12}^{-1}=(P1+P2)^{-1}=\left[\begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 9 \end{bmatrix}+\begin{bmatrix} 9 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 1 \end{bmatrix}\right]^{-1}$$

$$=\begin{bmatrix} 10 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 10 \end{bmatrix}^{-1}$$

Table 5 Shows the simulated data for tracks T1R1 (track1 from radar1) and T1R2 (track2 from radar2) over the 10 second time period. Used in example calculations.

TIME (Sec.)	TRACK1 FROM RADAR1 (T1R1)			TRACK1 FROM RADAR2 (T1R2)		
	RANGE (M)	**AZIM. (Deg.)**	**ELEV. (Deg.)**	**RANGE (M)**	**AZIM. (Deg.)**	**ELEV. (Deg.)**
1	1.21	1.57	1.489	3.3	1.71	2.97
2	27.67	30.07	12.66	26.75	29.8	12.43
3	34.9	38.08	24.6	37.16	37.7	24.89
4	50.2	55.2	36.19	52.19	55.7	36.6
5	66.86	80.82	41.91	70.12	80.01	42.02
6	68.47	108.15	52.86	71.4	107.88	53.76
7	77.5	129.75	68.89	80.5	128.5	68.87
8	81.6	146	79.57	85.93	146.4	79.9
9	85.8	169.4	80.13	89.77	168.08	81.6
10	95.75	177.7	88.45	99.25	176.7	88.7

$$H0\text{: } E12(k)=0\text{------}\textit{same targets}$$
$$H1\text{: } E12(k)\neq 0\text{------}\textit{different targets}$$

$$\textit{where for time instant } 1,\ e1=\begin{bmatrix}1.2-3.3\\1.57-1.71\\1.489-2.97\end{bmatrix}^T.\Sigma_{12}^{-1}.\begin{bmatrix}1.2-3.3\\1.57-1.71\\1.489-2.97\end{bmatrix}$$

$$\textit{Since there are } 10 \textit{ time instances, } e=e1+e2+...+e10=11.3766$$

We now establish the level, and the acceptance and rejection for the decision rule to be P_{FC}=0.01 (Probability of false correlation). (ie. Testing Type 1 error, testing to accept or reject H0).

α = Threshold ie. probability of rejecting H0 given H0 is true,
e is χ^2 (chi squared) distributed with N.X degrees of freedom,
where, N= The number of time instances =10,
X= The number of dimensions of vector "E12" =3.

So if $P_{FC} = 0.01$,this implies H0 is true and there is a 1% chance of rejecting the null hypotheses H0. Probability $(e>\alpha/H0)=P_{FC}$.

Using the χ^2 statistical table for $P_{FC} = 0.01$ for 30 degrees of freedom, the threshold is found to be 50.9 (Refer to Appendix G, for Chi squared distribution tables).

If $e<\alpha$ H0 is true (tracks same)
or $e>\alpha$ H1 is true (tracks not same)

For our example $11.3766 < 50.9$ hence H0 is true, and tracks T1R1 and T1R2 are the same.

Table 6 Shows the Mahalanobis distance for each of the 9 track pair combinations, summed over the 10 second time period.

TRACK PAIR COMBINATION	MAHALANOBIS DIST. (OVER 10 TIME INST)
T1R1T1R2	11.3766
T1R1T2R2	1.4E3
T1R1T3R2	1.01E4
T2R1T1R2	1.52E3
T2R1T2R2	8.5057
T2R1T3R2	8.19E3
T3R1T1R2	3.129E3
T3R1T2R2	3.446E3
T3R1T3R2	5.95E3

Simulated track data results showing the Classical Inference results with the Mahalanobis distance between the nine pairs of tracks from radar 1 and radar 2, are shown in Table 6 above.

T1R1T1R2 (track1 from radar1 and track1 from radar2) and T2R1T2R2 (track 2 from radar1 and track2 from radar2) are below the threshold indicating H0 is true, (ie. targets the same from radars 1 and 2). Hence the tracks can be fused together using the equations (assuming independent radars):-

$$Y_C = P2.(P1+P2)^{-1}Y1 + P1(P1+P2)^{-1}.Y2$$

4.2.2 Using ART2 For Simulated Track Data

Consider an ART2 neural net (used because of its pattern recognition and real time operating capabilities , as discussed in section 2.1.2) which is comprised of 30 processing elements in the F1 (Input layer) and 4 processing elements in the F2 (Output layer). Range, Azimuth, and Elevation data over the 10 time instances for each of the six tracks was input into the F1 layer (Refer to Fig. 26). The vigilance was initially adjusted with training tracks which were known to be the same from both radars because we need to set the degree of recognition needed in Art2 to cluster the test tracks (unknown) which are the same from both radars.

ART2 clustered the input tracks from both radars into 4 different category outputs, ie. category neurons 1 and 2 for the two track target pairs in the common surveillance volume of both radars, and the 2 other tracks into category neurons 3 and 4 respectively. Four processing elements were used in the output layer (F2) because I knew there were four different tracks from both radars. Similarly, by increasing the number of PE's in the (F2) output stage (ie. to 5,6...) and using the same vigilance value (0.995), ART2 clustered the 6 tracks into 4 different categories as before.
Hence in a scenario where you don't know the number of tracks to be clustered it is wiser to have a large number of PE in the output stage in-case the network is forced to cluster a track to an incorrect category neuron.

The F1 and F2 weights were initialised randomly between plus and minus the maximum initial weight parameter of 0.01 (input to a weight file using the HNC Neurosoftware ART2 package). The network parameters used in the F1 field STM (short term memory) calculations, (ie. parameters A, B, C, D, E, and maximum initial weight, as shown in Table 2), are assigned the values recommended by HNC(Hecht-Nielsen Neurocomputing) Neurosoftware ART2 manual [14] , as shown in Table 2 of the review chapter 2.1.2.6. Normal learning was enabled, this enables the network weights z_{ji} (top-down) and z_{ij} (bottom-up) to be updated. Normal learning incrementally moves the weights towards their asymptotic values.

The training tracks were created using the same random noise variances (using the Matlab random noise generator function "rand") in Range, Azimuth and Elevation from radars 1 and 2, as were the test tracks. Training tracks from each radar (R1 and R2) which were known to be from the same or different targets were presented to the network and the vigilance adjusted to a value of 0.995.
(The vigilance value was adjusted to the value of 0.995 because it was the degree of recognition needed in ART2 to cluster together the training tracks known to be the same from both radars.)

I presented the network with the same six tracks (ie. 3 from radar1 and 3 from radar2) as was used for hypothesis testing. The network correctly clustered T1R1 , T1R2 to category neuron 1, and T2R1, T2R2 to category neuron 2. T3R1 was assigned to category neuron 3 and T3R2 to category neuron 4. (Refer to sections 4.4 and 6.1 for summary of results and conclusions).

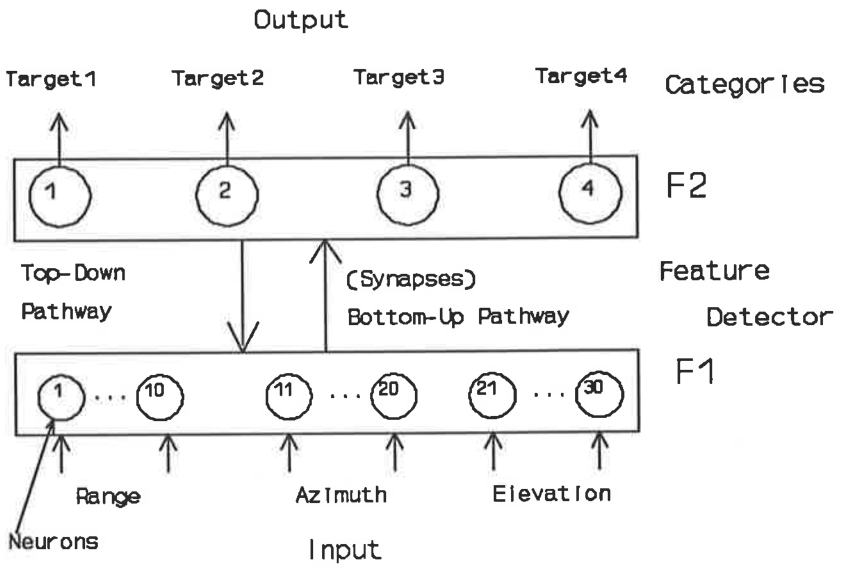


Fig. 26 The ART2 Neural Network consists of 30 Processing Elements used in the input stage and 4 Processing Elements in the output stage. The input consists of Range (metres), Azimuth (degrees), and Elevation (degrees) over 10 time instances.

4.3 Real Track Data Results

Consider two radars with different accuracies producing tracks which are independent (since tracks are independent, the covariance matrix is the sum of covariance matrices P1 and P2 from both radars). Adelaide Airport's Surveillance radar and Defence Science and Technology's (DSTOS) FPS-16 Tracking Radar (located at DSTO), using a sensor level tracking approach to form sensor tracks.

4.3.1 DSTO Radar

The DSTO FPS-16 tracking radar is a C-Band amplitude comparison monopulse tracking radar . Typical accuracy is ±10 metres in X, ±10 metres in Y, and ±3 metres in Z (where X, Y, Z are the cartesian co-ordinates, which will be explained latter). It has a maximum range of 40 nautical miles.

In summary the program shown in Appendix D which processes DSTO'S FPS-16 Radar target data, reads 3 binary data files indicating target positions in Range, Azimuth and Elevation. I convert the data to a real format, and then convert from polar to cartesian co-ordinates (so that data can be aligned in space from both radars) then finally adjust the cartesian x,y coordinates (offset) so that the targets position is with respect to Adelaide Airport's origin reference point (refer to details in Appendix 0).

4.3.2 Adelaide Airports Surveillance Radar

Adelaide Airport has two radars, primary and secondary. The primary Radar is a L Band (1320 MHz), its peak power is 2M watts and it has a maximum range of 160 nautical miles. It's antenna has a 1.3 degree azimuth beamwidth and has a 5 revolution per minute rotation rate. The secondary radar is co-mounted on the primary, it receives transponder altitude data from airborne targets. The primary radar's data format is in slant range(nm) and azimuth (degrees) with respect to true north.

The accuracy in X, Y, Z for targets at range less than 90nm is ± 407 metres in X; ± 648 metres in Y, and 20 metres in Z. The output format of the raw target data is: - Track no. (indicating a target track number), Slant Range (nm), Azimuth (Degrees), Altitude (feet) and time (local). The software shown in Appendix D converts the target track data to cartesian co-ordinates (with respect to its own radar head).

4.3.3 Trial Results

Radar1 (DSTO'S FPS-16 Radar) and Radar2 (Adelaide Airport's Surveillance Radar) tracked several targets for several minutes between the time period of 13:51:08 - 13:54:46 (APPENDIX K displays the cartesian plots over the entire tracking period 13:51:28-13:54:46). The DSTO Radar (R1) being a tracking radar can track one target (T1R1) at any one time instant and Adelaide Airport's Surveillance Radar (R2) tracked five targets (T1R2, T2R2, T3R2,T4R2,T5R2) for the same time period indicated (refer to Appendix E processed kinematic data).

The two track-to-track correlation techniques (Classical Inference and ART2) were used and results compared. Consider both radars as being independent using a sensor level tracking approach to form sensor tracks. After processing (ie. alignment of data in space and time) both radars output data in the format of X(metres), Y(metres) and Z(metres) (Cartesian co-ordinate system relative to Adelaide Airport's position). They have different accuracies with known variances in X, Y, and Z. ie. Thus their corresponding covariance matrices are:-

$$\textbf{\textit{DSTO RADAR (R1)}} \quad \boldsymbol{P1}=\begin{bmatrix}10 & 0 & 0\\ 0 & 10 & 0\\ 0 & 0 & 3\end{bmatrix}$$

$$\textbf{\textit{ADELAIDE AIRPORT RADAR (R2)}} \quad \boldsymbol{P2}=\begin{bmatrix}407 & 0 & 0\\ 0 & 648 & 0\\ 0 & 0 & 20\end{bmatrix}$$

The FPS-16 Radar (R1) is tracking one target and Adelaide Airport's Radar (R2) is tracking five targets. One of these targets is in the common surveillance volume of both radars (refer to Fig. 27)
Radar1 reports One target track and radar2 reports 5 target tracks over the time period 13:51:28 - 13:51:37. (APPENDIX K shows the cartesian X,Y & Z plots for the 10 second time period selected).

4.3.4 Using Classical Inference Results

As in the section 4.2.1 using simulated track data I want to test the hypothesis that the two estimates *Y1(K)* and *Y2(K)* are the same targets for the five track pair combinations from the DSTO (FPS-16) and Adelaide Airport radars. ie.

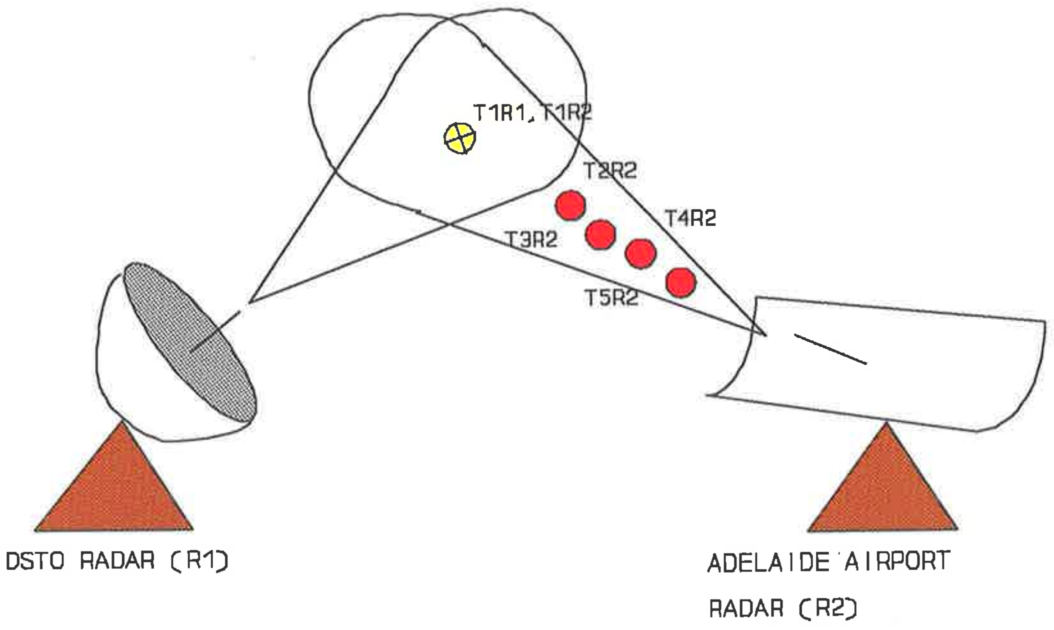


Figure 27 Adelaide Airport is tracking targets T1R2,T2R2,T3R2,T4R2,T5R2, and DSTO's radar is tracking target T1R1, which is in the common surveillance volume of both radars.

***Y1(k)* and *Y2(k)* = 3X1 Vector in x,y,z cartesian co-ordinates,**

at time "k" over the 10 second time period (from 13:51:28 - 13:51:37).

The Mahalanobis distance was calculated and summed over the 10 second time period for each of the 5 track pair combinations from both radars. One of the track pairs was found to be below the threshold value of 50.9 (Refer to log-linear plot Fig. 28) which was obtained using the Chi squared distribution for 30 (3x10) degrees of freedom, thus indicating the same target from both radars (Matlab program used to calculate Mahalanobis distance is similar to the one shown in Appendix C, except for the covariances P1 & P2 values , which are as shown above).

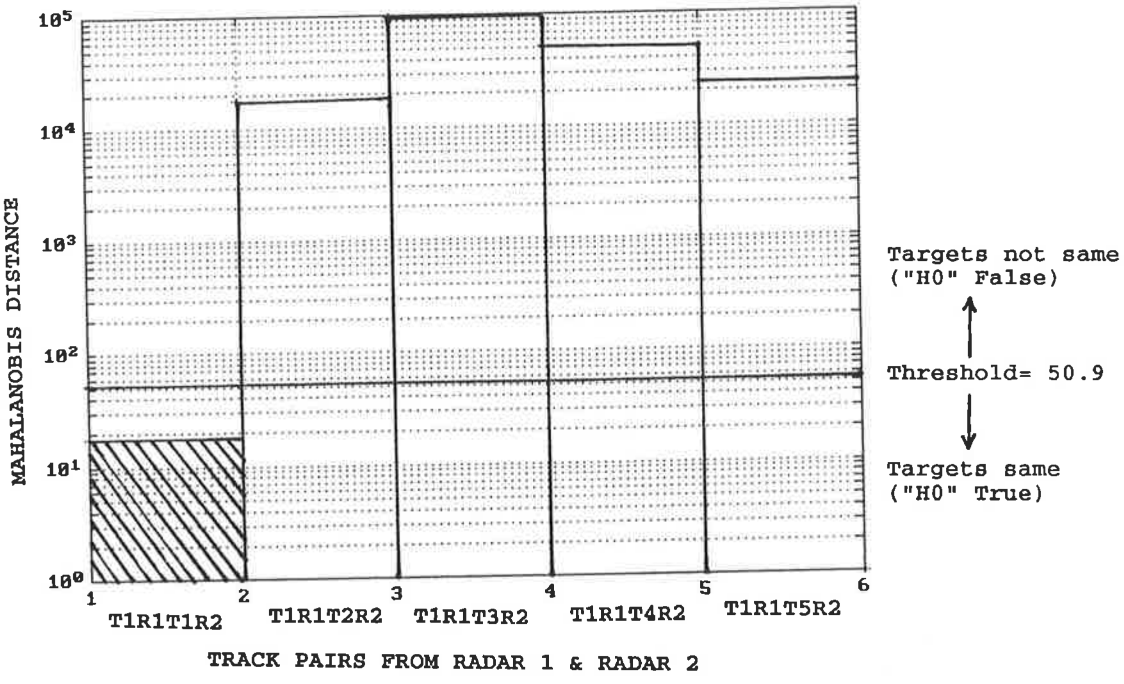


Fig. 28 Displays the Classical Inference results with the Mahalanobis distance between pairs of tracks from Radar1 and Radar2, together with the Level of Significance (threshold value).

Table 7 Shows the Mahalanobis distance for track pairs from Radar1 and Radar2.

TRACK PAIRS	MAHALANOBIS DISTANCE
T1R1T1R2	1.7847E1
T1R1T2R2	1.8424E4
T1R1T3R2	9.8913E4
T1R1T4R2	5.1815E4
T1R1T5R2	2.3502E4

4.3.5 Using ART2 Results

Consider an ART2 neural network which comprised of 30 processing elements in the F1 (Input Layer) and 5 processing elements in the F2 (Output layer). X, Y, and Z cartesian data, over the 10 time instances, for each of the six tracks was input into the F1 layer (refer to Fig. 29). The vigilance was initially adjusted with training tracks which were known to be the same from both radars, (V=0.9998), to set the degree of recognition needed in Art2 to cluster the unknown test tracks from both radars which are the same.

ART2 clustered the output into 5 different category outputs ie. category neuron 1 for the track target pairs T1R1T2R2 (T1R1 is track1 from radar1 and T1R2 is track1 from radar2), and other tracks T2R2, T3R2, T5R2, T4R2 into category neurons 2, 3, 4 and 5 respectively.

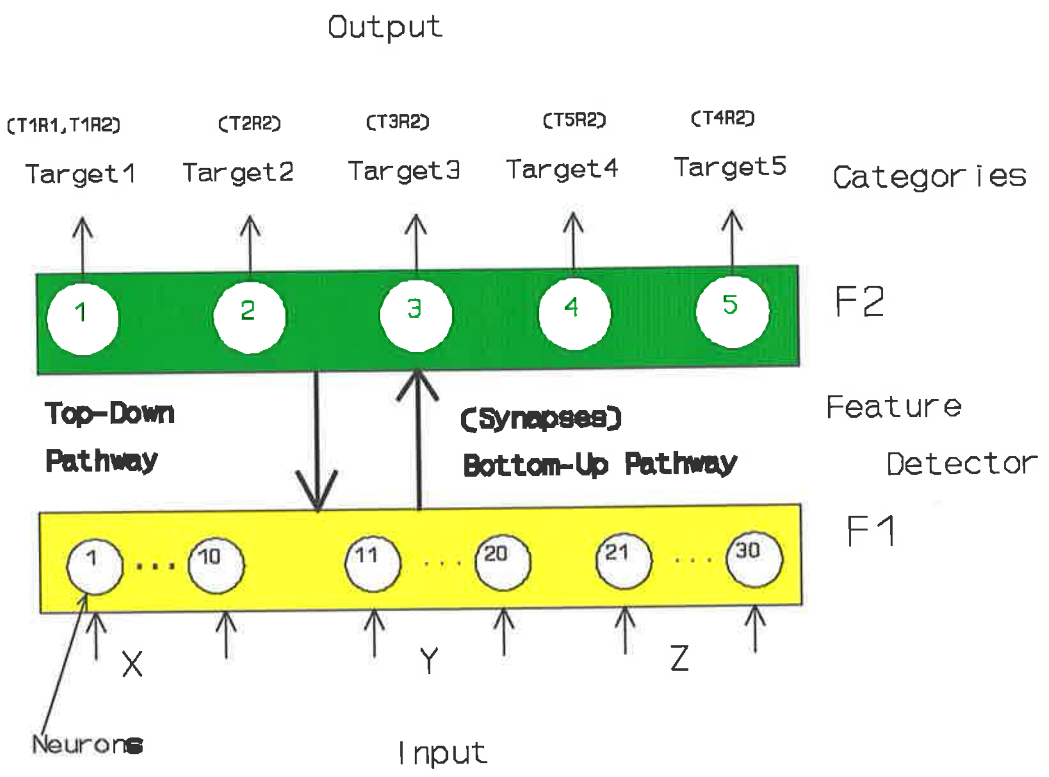


Figure 29 ART2 consists of 30 processing elements in the input stage & 5 in the output stage. The input consists of cartesian co-ordinates , X, Y & Z (metres) with respect to Adelaide Airport over the 10 time instances.

The F1 and F2 weights were initialised randomly between ± the maximum initial weight parameter of 0.01. The network parameters used in the F1 field short term memory (A, B, C, D, F) are assigned the values shown in Table 2 from the review paragraph 2.1.2.6. The input track data used is the same as that used for Classical Inference. Input track data is shown in Appendix E, (Note input data was offset to make the track data values from both radars positive.)

4.4 Summary of Results

The disadvantages of using the Classical Inference technique are derived from the need to know the accuracy of the sensors or standard deviation of tracks to obtain the covariance matrices P1(k) and P2(k). The Level of Significance or threshold has to be calculated which represents the probability of rejecting the null hypothesis "H0" (Targets Same), given that "H0" is true.

Time consuming computations (which can be crucial to a real time central computer in a military environment which is correlating and fusing hundreds of targets at any one time) are required in the hypothesis testing stage where the test statistic, which is the Mahalanobis distance (a measure of similarity between two vectors), summed at each point along the track, is compared with the Level of Significance.

The disadvantage of using Art2 is the requirement to adjust the vigilance parameter (which determines the degree of recognition required) with training tracks which are known to be the same from both radars.

The advantages of using the ART2 neural network are that there is no need to know the accuracy of the sensors. The network indirectly obtains this through the adjustment of the vigilance value when training the network with track data pairs which are the same targets from both sensors. No instructions are required to tell the network which category the track input belongs to it discovers it, on its own, in real-time. Where as in the Classical Inference case, the time consuming operation of first finding all possible track-to-track combinations from both sensors has to be made before any calculation (to determine if they are the same) are started. ART2 requires no preprocessing (as long as the data from both sensors are using the same unit of measure ie. metres) of the track input data from either sensor. The number of pattern classes or categories (ie. track pairs) need not be known in advance. ART2 can create new pattern categories that were not in the initial training set.

Chapter 5

Experimental Procedure and Results for Attribute Data Fusion

5.1 Introduction

Real-time target recognition can be achieved by integrating data from dissimilar sensor systems and priori information using artificial neural network technology.

Individual sensors, although effective, are limited in their capacity to identify targets. By selectively integrating knowledge sources, (ie. sensor outputs which provide us with a target's attributes) together with any priori information, sufficient information can be obtained to identify an airborne target with greater certainty.

In chapter 5 the thesis addresses the application of the backpropagation neural network technology for automatic target recognition fusing target features derived from a Continuous Wave (CW) Coherent (X band) Radar, which provides us with high resolution doppler signature measurements, together with a Surveillance Radar, which provides positional information of airborne targets, and priori information of flight times of targets flying regular flight paths, obtained from Adelaide Airport Flight time tables. Dempster-Shafer (Evidential Reasoning), and Fuzzy Reasoning (using the minimum method) data fusion techniques are compared with the neural network output results, when similar inputs are present in all three cases.

5.1.1 CW Radar Knowledge Source

From the CW Radar I obtain a high resolution doppler signature measurement. Signal processing techniques [27] such as Fourier analysis can be used to characterise the Doppler modulation of radar echoes from the airborne target returns (Refer to Fig. 30 and Fig. 32). The processed signal characteristics will provide an insight into distinguishing features between airborne targets with jet engines and propellers.

Doppler modulation is not only caused by flight motion (ie. difference in doppler frequency of each point scatterer around the aircraft), but also by rotating machinery which is dependent on engine RPM, the motion of the propeller and compressor or turbine

blades. The coherent radar, produces a continuous doppler spectrum due to the rate of phase change of the vector sum of the echoes from scattering points together with a distinct doppler line which is associated with the average echo due to the radial velocity of the target.
(Refer to Appendix F section 1.0 for details on the operation of CW Radars).

5.1.1.1 Jet Engine & Propeller Driven Aircraft Modulation

Jet aircraft modulation [28] is produced by the compressor or turbine blades of the engine. Since compressors and turbines contain relatively large number of blades rotating at high angular velocities, the modulation frequencies will be much higher than those of propeller driven aircraft.

At small aspect angles (0-10 degrees) the propeller doppler spectrum is confined mostly to the region around the airframe line (Refer to Fig. 32), and at larger angles (11-39 degrees) it has a much wider spread into the region lower in frequency than the airframe line.
The modulation sidebands produced by the jet engine compressor stage or turbine blades are spaced at different frequencies about the airframe line (usually more spread out in frequency than a propeller driven aircraft), (Refer to Fig. 30).

The body doppler (or airframe line) of both jet and propeller aircraft is obtained using the following Doppler equation:

$$f_{doppler} = \frac{(2.vel.f_c)}{c}$$

where,

vel = Radial velocity

fc = Operating frequency of CW Radar (9.83 GHz)

c = Speed of light.

For example, the commercial jet whose spectrum is shown in Fig. 30 was travelling at a radial velocity of 140 m/sec, using the above equation the body doppler frequency is calculated as being 9.174 KHz. You notice (from Fig. 30) that the harmonics are more spread out than the propeller driven aircraft spectrum (as shown in Fig. 32), due to the jet engine compressor stages producing different harmonics, spaced at different frequencies with respect to the body doppler. Similarly the propeller driven target whose spectrum is shown in Fig. 32 was travelling at a radial velocity of 50 m/sec, giving a body doppler of 3.27 KHz. Because of the low aspect angle of 7.7 degrees you notice from Fig. 32 that the propeller doppler spectrum is confined mostly to the region about the airframe line. (Refer to Appendix F section 2.0 for more details on Doppler Spectra for jet and propeller driven aircraft), (Appendix J displays the FFT spectrum of 14 airborne targets).

If the airspeed of commercial and propeller driven targets were different than the ones recorded in our experiments, the airframe lines would shift hence producing a different LPC spectrum. Due to the large number of blades rotating at high angular velocities of the jet engine, the spectrum will still be more spread out and evenly spaced at different frequencies about the airframe line [28], than that of a propeller driven aircraft whose propeller spectra will always be lower in frequency than the airframe line, as indicated by RE Gardner in reference [28] . Producing the characteristic spread of peaks found in the LPC spectrums, for commercial jet aircraft (shown in Fig. 31 and Appendix J).

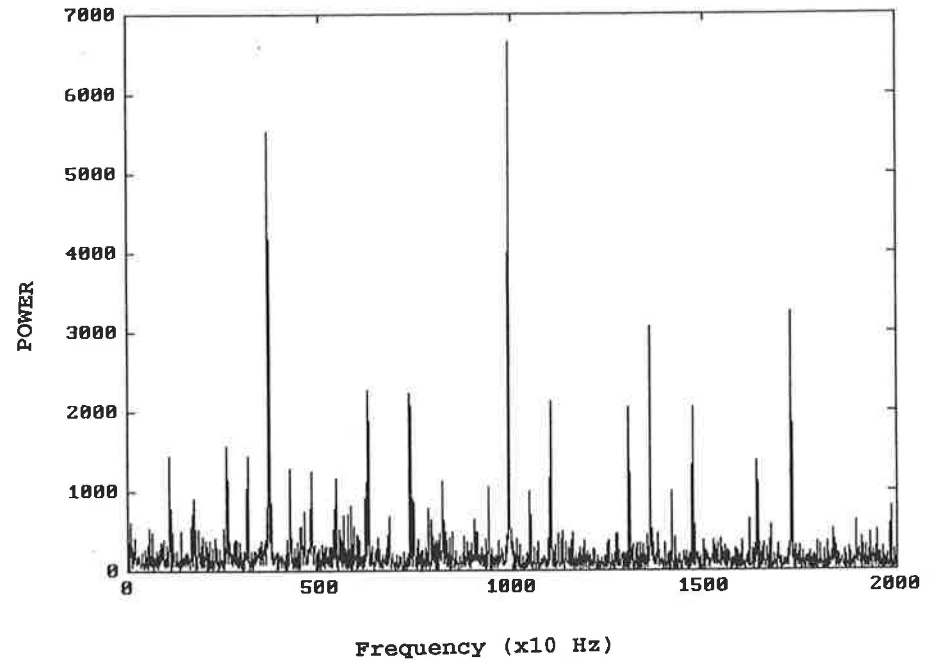


Figure 30 The FFT spectrum (magnitude squared, 4096 samples) of a commercial jet aircraft at an aspect angle of 20 degrees.(50 KHz sampling rate used). The calculated airframe line of the target is at 9.2KHz.

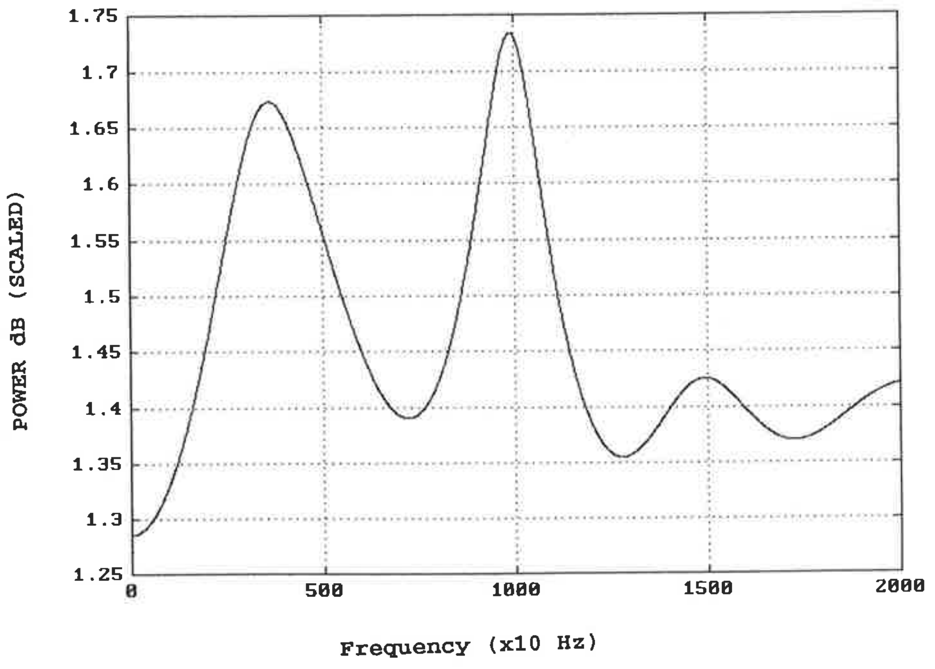


Figure 31 The Linear prediction spectral estimate of a commercial jet shown above.

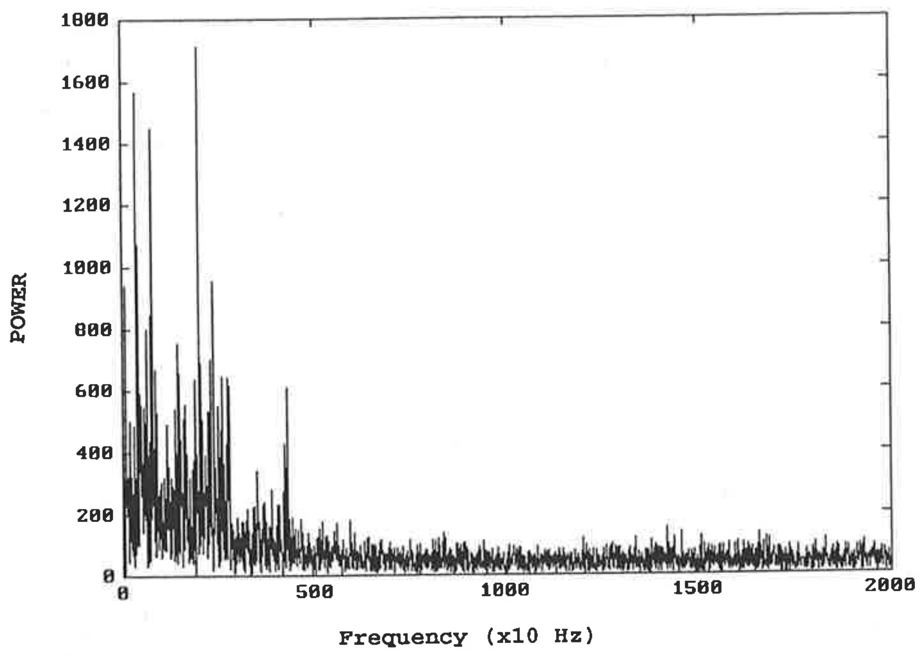


Figure 32 The FFT spectrum (magnitude squared, 4096 samples) of a propeller driven aircraft at an aspect angle of 7.7 degrees. The calculated airframe line of the target is at 3.2 KHz.

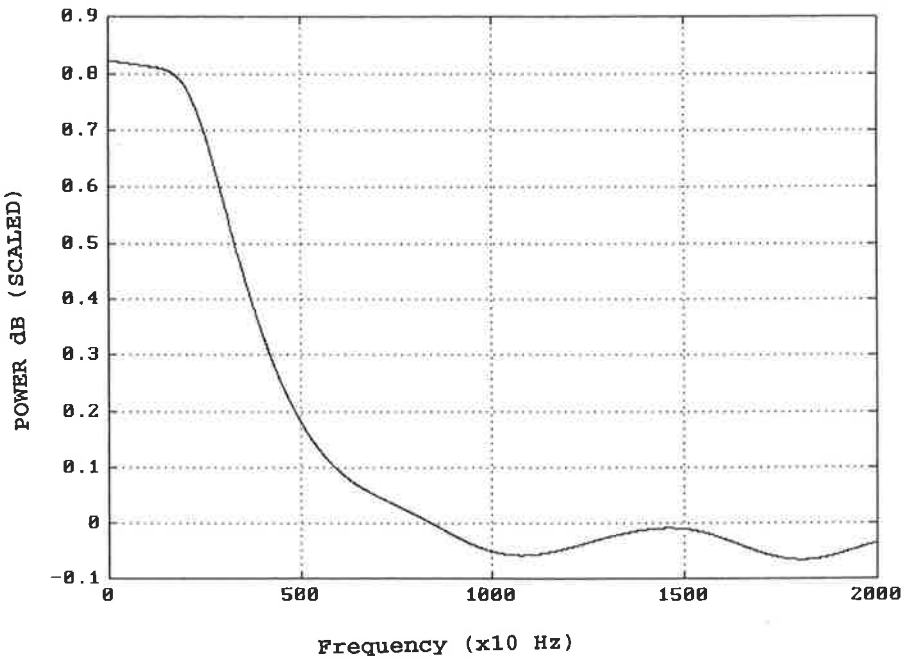


Figure 33 The linear prediction spectral estimate of a propeller driven aircraft shown in above.

5.1.2 Surveillance Radar and Priori Knowledge Source

From Adelaide Airport Surveillance radar I obtain positional information. Range(nm) and Azimuth(degrees) are converted into cartesian co-ordinates X(nm) and Y(nm) with respect to Adelaide Airport's position. Priori information on regular flight paths , and arrival/departure times (from flight time tables) can be obtained for selected commercial flights.

Information from the three knowledge sources (CW radar, Surveillance radar and flight time table information) can be used to train a neural network to provide a greater certainty (than would otherwise be obtained using fewer knowledge sources) on the identity of the airborne target in real time.

5.2 Experimental Investigation

Three trials to collect data from sensors were organised on the same day and times of the week to ensure flight timetables were consistent (ie. Tuesdays between hours of 3:00 - 5:30 pm). Two types of airborne targets (propeller driven and commercial jets) were tracked using the CW Radar (located at DSTO Salisbury) at distances not greater than 40nm and at aspect angles not greater than 60 degrees (with respect to the front of the aircraft). Six commercial jets and eight propeller driven aircraft were tracked at various aspect angles. The type of commercial jets ranged from Ansett Airlines Boeing 727, Australian Airlines 737 to British Aerospace BAE-134. The propeller driven aircraft were mainly single and twin engine cessnas.

Both radars recorded time information of airborne targets being tracked. The Radar used to collect the doppler records is an X Band (9.83 GHz) CW system developed at Microwave Radar Division (DSTO) using two six foot diameter antennas. The elevation on the azimuth mount is slaved with a servo loop to the FPS-16 Tracking Radar which directs the antennas at the target being tracked. Radar data records were taken from each experiment which consisted of a combination of FPS-16 Range, Azimuth, Aspect angle , Aspect rate and time the target is tracked. Each second the radar collected 640k bytes of complex data (In-Phase & Quadrature). The radar frequency was set to 9.83 GHz. An A/D sample rate of 50 KHz permits an adequate Doppler Spectrum for signal processing without aliasing.

5.3 Processing Data from Both Sensors [29]

Slant Range and Azimuth data from the Surveillance Radar is converted to cartesian coordinates X & Y.

Features associated with propeller or jet engine modulation of the doppler can be identified in the spectrum by performing a Fast Fourier Transform (FFT) on the radar returns. It is not always possible to extract all features (for input to a neural network) by the use of a single digital signal processing tool. A combination of Fourier Transform and Linear Prediction (LP) was used to process the radar returns from targets.

The idea in using FFT together with LPC comes from a paper [38] by D. Nandagopal, presented in Radarcon 90 in which he describes an experimental study of the characterisation of doppler returns from flat rotating blades carried out in the Microwave Radar Division of DSTO. One of the dominant features of radar echoes from rotating blades was the presence of a "plateau" in the frequency spectrum. The plateau of the radar returns is due to the variable doppler contributions of the blades. Signal processing techniques such as Fourier transforms and Linear Prediction were used to characterise the doppler modulation of radar echoes from rotating blades. The smoothed LP plots clearly defined the edges, lengths and heights of the plateau regions in Nandagopals [38] application (from which the identity of the rotating blade could be determined). Whereas in my application the smoothed plots were used as inputs into the backpropagation neural network over a defined frequency period (discussed in section 5.4.1 in more detail).

The basic idea in Linear Prediction (LP) [30],[27] analysis is that a signal sample can be approximated as a linear combination of past signals by minimising the sum of the squared differences (over a finite interval of time) between the actual signal sample and the linearly predicted ones. A unique set of predictor coefficients can be determined, which characterises the signal data. Once predictor constants are computed then an all pole model can be developed to fit the data. (Details on Linear Prediction are discussed in Appendix A).

The Doppler spectrum of the Radar returns can be modelled using LP. [Refer to Fig. 31 and Fig. 33]. In this particular application since I am using a small model order the LPC is basically acting as a smoother rather than a high resolution spectral estimator [29].

5.4 Experimental Procedure

Three backpropagation neural networks (NN1, NN2 & NN3) were used (Refer Fig.35) to fuse the targets attribute data from the sensors together with priori information on arrival times & flight paths. (APPENDIX N provides an exposition into data fusion of multiple classifiers).

5.4.1 Backpropagation neural network 1 (NN1)

The first neural network (NN1) is used to identify targets from the processed doppler modulation (using CW radar) as being either jet or propeller driven aircraft.
NN1 consists of :-

11 neurons in the input layer,
8 neurons in the hidden layer &
1 neuron in the output layer.

A backpropagation neural network with 2 (active) layers (ie. one hidden and one output layer) was chosen for the classification of processed doppler modulation data primarily because it should be easier to train than a network with more layers. The reason for using the 11 input neurons in NN1 was to cover the required range of frequencies between 2-12 KHz (in 1 KHz intervals). Eight neurons in the hidden layer produced the shortest time needed to train the network (using CW radar doppler data as inputs). A single output from the sensor network NN1 is all that was needed to pass it's decision (on the identity of the processed doppler data), to the fusion neural network NN3.

The real part of the complex data from the CW Radar (doppler modulation) was selected and processed for the target being tracked by both radars (for nose aspect angles less than 60 degrees). The FFT of the LP coefficients were calculated and plotted (Refer to Fig. 31 and Fig. 33). The use of LP using a small model order (details shown in Appendix A) produces a smoothing effect hence defining any plateau or peak regions of the spectrum (as shown in Ref. [38]), and simplifying the input data to the neural network NN1. Because most of the important modulation information on all the recorded targets was present between 2000 & 12000 Hz, and radar noise was present for frequencies less than 400 Hz for some targets, eleven data values were selected from the above plot (relative to the noise floor) between the frequency values of 2000-12000 Hz in 1000 Hz intervals as inputs to the backpropagation neural network NN1 . NN1 classifies the input data as either being that of a jet or propeller driven aircraft.

Continuous Wave (CW) radar data (obtained from trials 1 and 2) from four commercial jets and three propeller driven aircraft over the eleven frequencies were used to train NN1. CW radar data for two commercial jets and five propeller driven targets (obtained from trial 3) were used to test NN1 (Refer to Appendix H for data tables of training and testing pre-processed CW data).

In Appendix M, verification tests performed on NN1 are described using the CW radar data stored in array0. To test the integrity of the NN1's output when CW Radar data (real part, stored in array0) is input, noise was added before being processed using LP. Also test and train data were swapped and results noted.

The output result of NN1 is fed into the input of the third neural network NN3 (Refer Fig. 35) which fuses it with data from the other knowledge sources.

Training and test CW Radar data (pre-processed using LP) for NN1 is shown in Appendix H Tables 23 and 24. The FFT and LP plots for the corresponding training and test data for commercial jets and propeller driven aircraft are shown in Appendix J. The Linear Prediction spectral estimates, figures 60, 62, 64, 66, 68, 70, and 72 represent the training plots and Figures 46, 48, 50, 52, 54, 56 and 58 represent the test plots.

5.4.2 Backpropagation neural network 2 (NN2)

Since all three trials were performed on the same day and times of the week, arrival & departure times of regular flights were consistent. Three regular commercial flights arrive into Adelaide Airport on that day between the hours of 3:00 to 5:30 pm. They are from Darwin (at 4:30 and 5:30) and Perth at 4:30. (Appendix I contains the Darwin & Perth flight path training data tables, Fig. 34 indicates the flight-paths).

NN2 was trained to identify the targets on Perth and Darwin Flight paths and to identify all other flight paths as unknown. "X" and "Y" cartesian coordinates with respect to Adelaide Airport in Nautical Miles are the inputs to NN2. As shown in Appendix I (Tables 25 and 26) thirty four cartesian (X,Y) coordinates (with respect to the origin point at Adelaide Airport , X=0, Y=0) were used to train NN2 to identify the targets as being on one of the 3 flightpaths.

NN2 consists of :-

2 neurons in the input layer
8 neurons in the hidden layer, and
3 neurons in the output layer.

The 3 neurons in the output layer represent the three possible flight paths ie. Darwin (1 0

0), Perth (0 1 0) and unknown (0 0 1). Outputs from NN2 are fed into the input of NN3.

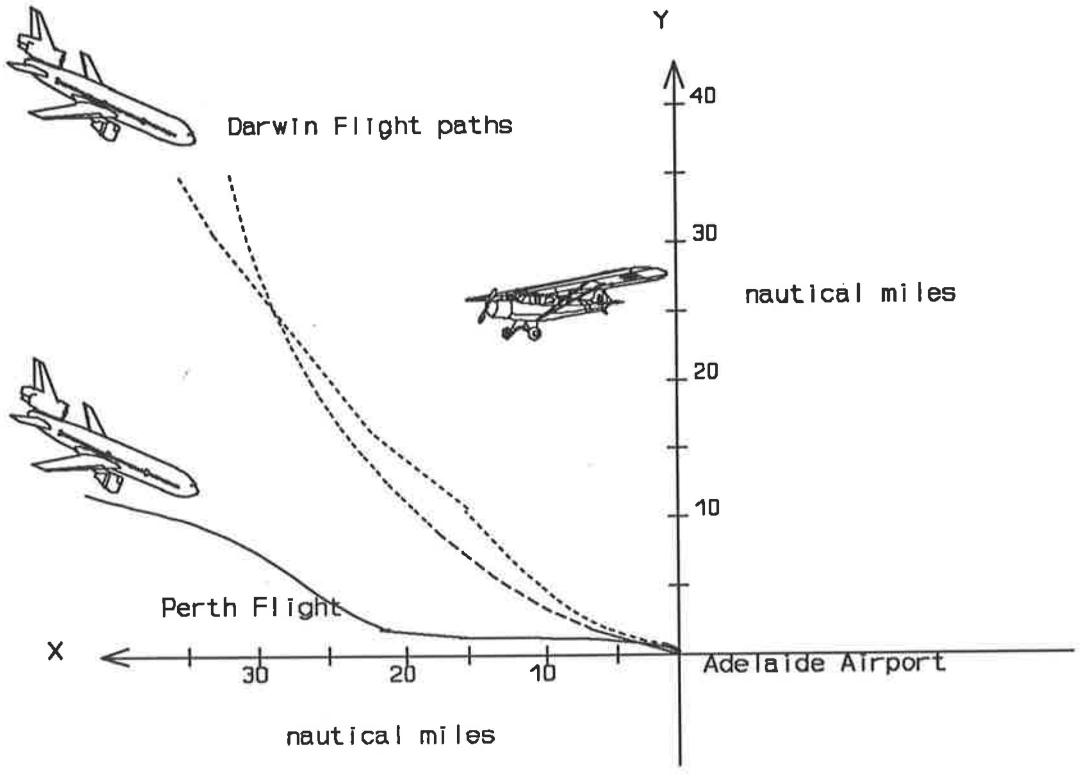


Figure 34 Typical Darwin and Perth jet arrival flight paths to Adelaide Airport.

5.4.3 (Fusion) Backpropagation neural network 3 (NN3)

The inputs to NN3 are the outputs of NN1 and NN2 and real time information (ie. time the target is locked on by the radar). Assuming due to delays and early arrivals, a commercial flight (Boeing 727/737) from either Perth or Darwin can still be on a flight path (within 35 Nm of Adelaide) ± 30 minutes from its scheduled arrival time.

Hence real time data is presented to the input of NN3 in one of the binary forms, as follows:-

1 0 0	represents any time between the hours of 3:00-4:00 pm,
0 1 0	" " " " " " " 4:01-5:00 pm,
0 0 1	" " " " " " " 5:01-6:00 pm.

The data used for training NN3 to identify possible target types, reflected the priori knowledge that Darwin flights are only possible (within 35 Nm of Adelaide Airport) during the hours of 4:00-5:00 and 5:00-6:00 pm, and Perth flights only possible during the hours of 4:00-5:00 pm.

NN3 consists of :-

7 neurons in the input layer,
12 " " " hidden layer, and
4 " " " output layer.

The output layer represents the certainties associated with target identification, as follows :-

(A) Darwin jet.
(B) Perth jet.
(C) Unknown jet. (ie. jet, on unknown flight path at any time, or known flight path and unknown time).
(D) Propeller driven aircraft (ie. on any flight path at any time.)

NN3 was trained on the binary truth table (refer Table 8) which reflects the above desired output (taking into account priori knowledge on flight arrival times) for the ideal inputs from both sensors .

INPUT TO NN3							OUTPUT FROM NN3				ID
JET/ PRP	FLIGHT PATH			TIME (pm)							
=1/0	Darw	Perth	Unkno	3-4	4-5	5-6					
1.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	C
1.0	1.0	0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	A
1.0	1.0	0.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	A
1.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	C
1.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	B
1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	C
1.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	C
1.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	C
1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	C
0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	D
0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	D
0.0	1.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	D
0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	D
0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	D
0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	D
0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1.0	D
0.0	0.0	0.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	D
0.0	0.0	0.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	D

Table 8 Is the truth table used to train NN3 back propagation neural network.
"A" INDICATES A JET FROM DARWIN.
"B" INDICATES A JET FROM PERTH.
"C" INDICATES AN UNKNOWN JET (ie.Unknown flight path & or unknown time).
"D" INDICATES A PROPELLER AIRCRAFT (ie. Which can be on any flightpath or time).

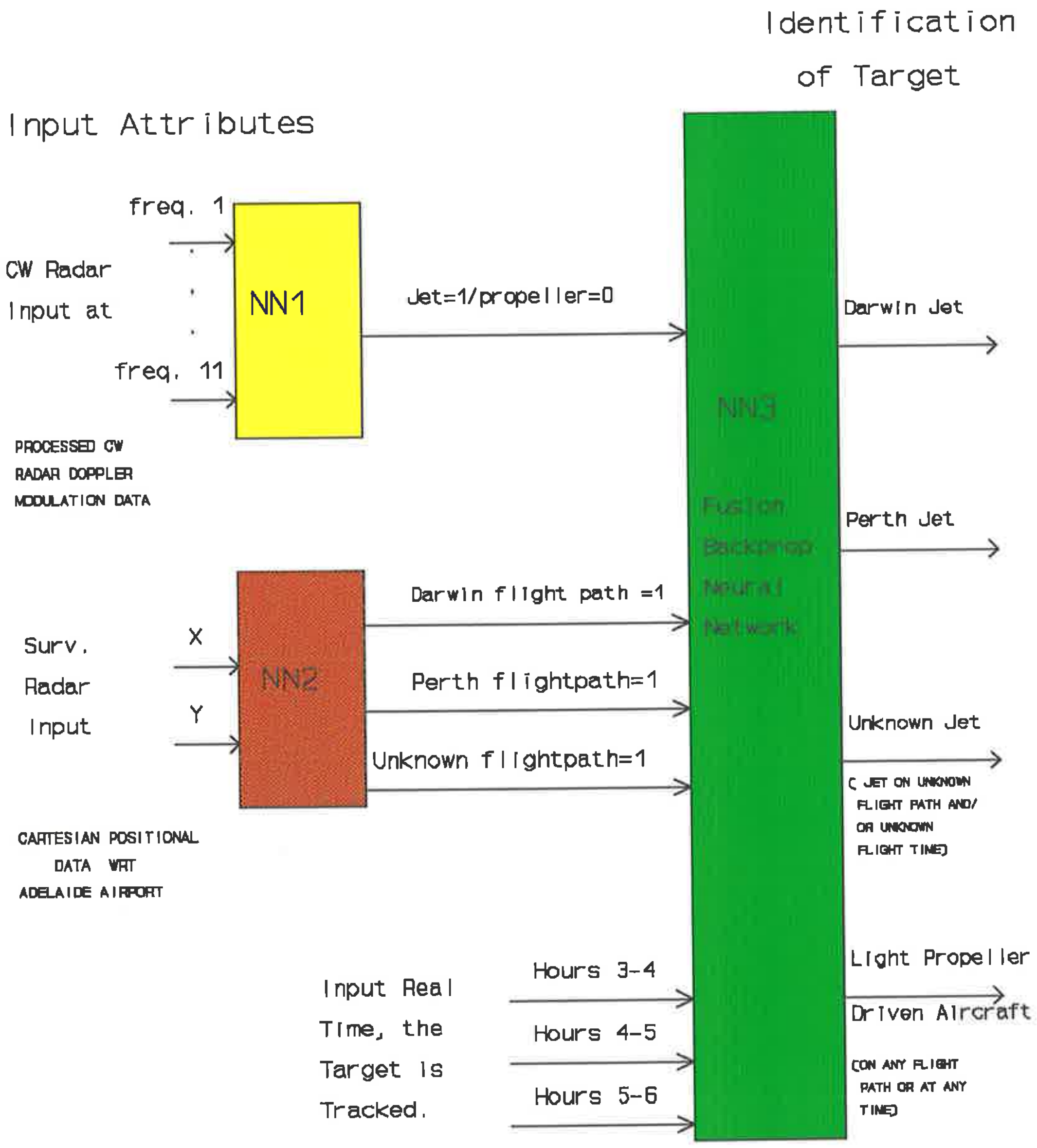

Identification of Target
Input Attributes
freq. 1
CW Radar Input at
freq. 11
NN1
PROCESSED CW RADAR DOPPLER MODULATION DATA
Jet=1/propeller=0
NN3
Fusion Backprop Neural Network
Darwin Jet
Perth Jet
Surv. Radar Input
X
Y
NN2
CARTESIAN POSITIONAL DATA WRT ADELAIDE AIRPORT
Darwin flight path =1
Perth flightpath=1
Unknown flightpath=1
Unknown Jet
(JET ON UNKNOWN FLIGHT PATH AND/ OR UNKNOWN FLIGHT TIME)
Input Real Time, the Target is Tracked.
Hours 3-4
Hours 4-5
Hours 5-6
Light Propeller Driven Aircraft
(ON ANY FLIGHT PATH OR AT ANY TIME)

Fig 35 Displays the 3 backpropagation neural nets used to fuse the doppler modulation attribute data from the CW Radar together with flight positional data from the Surveillance Radar and time information when the target is locked on by both sensors. Flight path and arrival time priori information has been used to train NN3.

5.5 Mass Computations for use with Dempster-Shafer and Fuzzy Reasoning

To do the mass computations for use with the D-S and Fuzzy Reasoning methods I first have to convert sensor measurements into a "probability mass distribution" over propositions, the propositions in this case being the proposition that the attribute from the three knowledge sources (CW Radar, Surveillance Radar and priori Flight Time-table information) have come from one of the following targets:- Darwin jet, Perth jet, Unknown jet, Propeller driven aircraft.

Since the output of NN1 gives us a measure of the certainty that the target is a jet/propeller driven aircraft (when attribute data from the CW Radar is present). We can derive a probability mass distribution on the four propositions.

ie. For target 1 (as shown in Table 10.) NN1's output indicates a 99.98 % probability that the target is a jet, this would indicate that the target is equally likely to be either a Darwin jet, Perth jet, or Unknown jet (ie. 99.98%, 99.98%, 99.98% certainty) respectively and 0.02% chance of it being a propeller driven aircraft as shown in Table 15. A set of "basic numbers" is then computed (for the four propositions) by normalising the resultant to bring their total value to one. This process is equivalent to computing the probability of the target being one of the four propositions. The measurement uncertainty "theta" of the sensor (knowledge source) is accounted for by weighting each basic mass number by a factor equal to "(1-theta)". This new set of mass numbers then represent the contribution of the knowledge source to the support of each proposition.

Raw mass assignments derived from NN1 for the remaining seven targets are shown in Table 15, with their normalised mass assignments, (assuming an uncertainty of 1% or 0.01 from the CW Radar Sensor) shown in Table 16 (ie. target example 1: 0.33, 0.33, 0.33, 0.0, for Darwin jet, Perth jet, Unknown jet, Propeller aircraft propositions respectively)

The output from NN2 gives us a measure of the certainty that the target is on either a Darwin, Perth or Unknown flight path. Using target 1 again (as shown in Table 10), NN2's output (when positional data is input from the Surveillance Radar) indicates a 99.01% certainty that the target is on a Darwin flight path, this in tern implies that the target is equally likely to be one of the following propositions, a Darwin jet (99.01%), Unknown jet (99.01%) or Propeller driven (99.01%) aircraft, and a negligible chance of it being a Perth jet as shown in Table 15. The normalised results for the 8 targets (assuming 1% uncertainty from the Surveillance Radar) is shown in Table 16 (ie. target example 1:- 0.33, 0.0, 0.33, 0.33, for the 4 propositions)

From Flight Time-table information I derive Figure 36. Thus I know that a Perth jet is due to arrive into Adelaide between the hours of 4:00-5:00 pm and two Darwin jets are due to arrive between the hours of 4:00-5:00 pm and 5:00-6:00 pm. I also assume that propeller driven aircraft are in the air at all times between 3:00-6:00 pm, and that commercial jets are unlikely to be in the air between the hours of 4:00-6:00, in the quadrant air space interested in.

Radars started tracking target 1 (within 35nm of Adelaide airport) in the time period of 3:00-4:00 pm. Using Fig. 36 I derive the following "basic numbers" 0,0,1,1; for the 4 proposition Darwin jet, Perth jet, Unknown jet and Propeller aircraft respectively. This indicates equally high likelihood of the target being either an unknown jet or propeller driven aircraft and negligible chance of it being a Darwin jet or a Perth jet. Normalising and taking into consideration the uncertainty (assume 1% or 0.01) I obtain the value of (0,0,0.495,0.495, for the 4 propositions) as shown in table 16.

Target No.	% Darwin Jet	% Perth Jet	% Unknown Jet	% Propeller Aircraft
1	1.78	0.0	98.02	0.2
2	96.0	1.76	0.59	0.78
3	0.0	0.04	0.06	99.91
4	0.0	0.0	1.16	98.84
5	0.0	0.0	0.02	99.97
6	0.0	0.01	0.02	99.97
7	5.25	0.09	0.0	94.75
8	72.4	4.9	22.8	0.00

Table 9 Shows the probability (as a percentage value) that the targets' ID is one of the 4 possible outputs from NN3.

5.6 Discussion of Experimental Results using NN3

Data from trials 1 and 2 were used to train NN1 and NN2 backpropagation neural networks. Results from trial 3, together with a simulated scenario input, were used for testing the neural network structure. For the third trial seven different targets (as shown in Table 9 & 10), were tracked during various times between the hours of 3:00-5:30 .

Table 11 displays the raw output results from the neural network (NN3) when trial 3 data is input. Note, that NN1 identifies the first target example as being a jet, and NN2

TARGET NO.	TRIAL3 & SIMULATED INPUT TO NN3						
	NN1 O/P [JET/PR]	NN2 O/P [FLIGHT PATH]			LOCKON TIME I/P [TIME]		
	[=1/0]	DAR.	PER.	UNK.	3-4	4-5	5-6
1	0.9998	0.9901	0.0000	0.0134	1.0	0.0	0.0
2	0.9994	0.9968	0.0000	0.0055	0.0	1.0	0.0
3	0.0032	0.4331	0.9474	0.0000	1.0	0.0	0.0
4	0.0029	0.0019	0.0000	0.9985	1.0	0.0	0.0
5	0.0092	0.0042	0.0000	0.9970	0.0	1.0	0.0
6	0.0500	0.0000	0.0000	0.9993	0.0	1.0	0.0
7	0.1608	0.9974	0.0000	0.0043	0.0	0.0	1.0
8	0.9998	0.3370	0.0000	0.4979	0.0	1.0	0.0

Table 10 Shows the raw data input to NN3, from NN1 & NN2 outputs.

identifies it as a target flying on a Darwin flight path (refer to Table 10). This normally would indicate a Darwin jet but because I have priori information from a third knowledge source (ie. timetable), which indicates that no flight from Darwin is possible that time of day. In consequence NN3 concludes (98% probability) that the target must be an unknown commercial jet flight, which happens to be flying in a Darwin flight path. The output of NN3 (fusion backpropagation neural network) will indicate a measure of certainty on the identity of the four possible target types, ie. (Refer Table 9) 1.78% probability that the first target from trial 3 is a Darwin jet, and 98% that it is an unknown commercial jet.

For the second target example, NN1 produces an output of (0.9994) (refer to Table 10), which indicates a high likelihood that the target is a jet. An output from NN2 (0.9968, 0.0, 0.0055) indicates a high probability that the target is on a Darwin Flight Path. The target was tracked between the hours of 4:00-5:00. For the above input data, NN3 (trained) output the following results (0.984, 0.018, 0.006, 0.008) as shown in Table 11, indicating a 96% probability that the target is a Darwin jet scheduled to arrive into Adelaide at 4:30 (as shown in Table 9).

For the next five targets (no. 3 - 7 , shown in Table 10), NN1 indicates that they are propeller driven aircraft ie. (0.0032, 0.0029, 0.0092, 0.05, 0.16) respectively. The output of

NN2 indicates that they are on (Perth, Unknown, Unknown, Unknown, Unknown) flight paths respectively. Fusing the above outputs from NN1 and NN2, together with real time information, NN3 produces an output (Table 9 & 11) which indicates a high probability that the targets are propeller driven, flying on any flight path and at any time.

Target no. 8 was simulated to produce the scenario of a Darwin jet which has strayed a couple of nautical miles from the known Darwin flight path (during the hours of 4:00-5:00). ie. Assume NN1 indicates a 99.9% probability that the target is a jet (output of 0.9998), and NN2 indicates a 60% probability that the target is on a Darwin Flight Path. The output results, from the fusion network NN3, indicate a 72.4% probability that it is a jet from Darwin and a 22% probability that it is an Unknown jet on an unknown flight path (refer to Table 9).

TARGET NO.	**OUTPUT FROM NN3**			
	DARWIN JET	**PERTH JET**	**UNKNOWN JET**	**PROP. AIRC.**
1	0.011029	0.000237	0.986725	0.002814
2	0.984128	0.018254	0.006511	0.008746
3	0.000000	0.000056	0.000672	0.996808
4	0.000000	0.000000	0.0111724	0.994852
5	0.000000	0.000223	0.000013	0.997230
6	0.000000	0.000272	0.000021	0.995945
7	0.054552	0.000000	0.000000	0.984433
8	0.516379	0.035242	0.163251	0.006903

Table 11 Shows the output from the fusion neural network NN3 when presented with the inputs of table 10.

5.7 Discussion of Results using Dempster-Shafer

5.7.1 Numerical Example Target No.1

The normalised mass assignments for target example 1 (obtained from Table 16) will be used to show how the attributes are fused from the 3 knowledge sources to facilitate automatic processing using Dempster-Shafers combination rules [15,5].

From Table 16 I obtain the normalised mass assignments for the 4 propositions or target types (Darwin jet, Perth jet, Unknown jet, Propeller aircraft ie T1,T2,T3,T4 respectively) of target no. 1 , which was derived from the outputs of NN1 and NN2 and Fig. 36 ; assuming an uncertainty of 1% from the 3 knowledge sources, which could be a reflection of information/sensor error or accuracy.

Mass Function derived from CW Radar (m1) and Surveillance Radar (m2):-

m1=(T1,T2,T3,T4)=(0.33, 0.33, 0.33, 0.0) ; theta1 = 0.01

m2=(T1,T2,T3,T4)=(0.33, 0.0, 0.33, 0.33) ; theta2= 0.01

Dempster's rules of combination are used to obtain the matrix shown in Table 12, with the probability mass assignments that are to be combined given along the first column (m2) and the last row (m1). The computed elements (for a given row and column) of the matrix are the product of the probability mass values in the same row of the first column and the same column of the first row.

The assignments of the elements in Table 12 are according to the rules below [15,5]:

(1) The product of mass assignments to two propositions that are consistent leads to another proposition contained within the original.
m1(A1).m2(A1)=m(A1)

(2) The product of the mass assignments to uncertainty and the mass assignment to another proposition leads to a contribution of that proposition.
m1(theta).m2(A2)=m(A2)

(3) The product of uncertainty and uncertainty leads to a new assignment to uncertainty.
m1(theta).m2(theta)=m(theta)

(4) When inconsistency occurs between the knowledge sources, we assign a measure of inconsistency denoted "k" to their products.

m1(a1).m2(a2)=k

m2 theta=0.01	T1=0.0033	T2=0.0033	T3=0.0033	theta=.0001
m2 T4=0.33	K=0.1089	K=0.1089	K=0.1089	T4=0.0033
m2 T3=0.33	K=0.1089	K=0.1089	T3=0.1089	T3=0.0033
m2 T1=0.33	T1=0.1089	K=0.1089	K=0.1089	T1=0.0033
	m1 T1=0.33	**m1 T2=0.33**	**m1 T3=0.33**	**m1 theta=0.01**

Table 12 Shows how Dempster's rule is used to combine the mass vector m1, with the mass vector m2. (Note "k" represents a measure of inconsistency).

In order to compute the new mass vector, I first sum all the assignments to k (ie. k=0.7623 in our example. The new mass vector is computed by summing the appropriate entries in the matrix and dividing by the normalisation factor (1-k=0.2377).

The new entries are:-

theta=0.0001/0.2377 = 0.00042

T1=(0.0033+0.0033+0.1089)/0.2377 = 0.4859

T2=0.0033/0.2377 = 0.01388

T3=(0.0033+0.1089+0.0033)/0.2377 = 0.4859

T4=0.0033/0.2377 = 0.01388

Plausibility was computed as shown in this example:

P(T1)=m(T1)+m(theta) = 0.4859+0.00042 = 0.4863

	SUPPORT	PLAUSIBILITY
T1	0.4859	0.4863
T2	0.0139	0.0143
T3	0.4859	0.4863
T4	0.0140	0.0144

Table 13 Shows the resulting support and plausibility for the propositions listed.

Thus m1 and m2 are combined to produce a new mass m3.

m3 = (T1,T2,T3,T4)=(0.4859, 0.01388, 0.4859, 0.0138)
with resulting uncertainty of 0.00042

Based on sensor data alone (ie. CW Radar and Surveillance Radar) the method leads to two primary hypotheses, T1(0.4859, 0.4863) and T3(0.4849, 0.4863). The support and plausibility values of 0.4859 and 0.4863 for propositions T1 and T3 indicate that the target is either a Darwin jet or an Unknown jet. T1 and T3 are favoured over the other propositions. Both propositions at this stage are equally as likely to occur.

The flight timetable knowledge source (Fig. 36) indicates a high likelihood of encountering an Unknown or a Propeller aircraft target at the radar lockon time (between 3:00-4:00 pm) for target example 1, and I derive the following priori normalised mass assignments for the 4 propositions (assume an uncertainty of 1% or 0.01).

m_{priori} = (T1,T2,T3,T4) = (0, 0, 0.495, 0.495) ; theta = 0.01

which, when integrated with m3, results in a mass function:-

$m_{composite}$ = (T1,T2,T3,T4) = (0.0188, 5E-4, 0.952, 0.028)
with resulting uncertainty, theta= 1.63E-5

	SUPPORT	PLAUSIBILITY
T1	0.0188	0.018816
T2	5E-4	5.16E-4
T3	0.952	0.952016
T4	0.028	0.028016

Table 14 Shows new Support and Plausibility for the propositions after combining the new knowledge source $m_{priori.}$ (Note new resultant uncertainty has decreased to 1.63E-5 from 0.00042).

This leads to the following hypotheses. When new priori evidential evidence is brought to bear, the support for T3[0.952, 0.952016] (Unknown jet), becomes significantly greater than for all others. ie. Proposition 1 (Darwin jet) has a lower support than the previous combination as shown in Table 13 (0.4859 goes to 0.0188). In fact, as shown in Table 14 all other propositions except T3 drop to very insignificant levels of support.

5.8 Discussion of Results using Fuzzy Reasoning

5.8.1 Minimum Method

Using example target1, I will discuss an alternative to Dempster-Shafers (which is less computationaly intensive) of fusing the 3 knowledge sources,

ie **m1=(T1,T2,T3,T4) = (0.33, 0.33, 0.33, 0), theta1= 0.01**
m2=(T1,T2,T3,T4) = (0.33, 0, 0.33, 0.33), theta2= 0.01
mpriori=(T1,T2,T3,T4) = (0, 0, 0.495, 0.495), theta3= 0.01

Using the minimum method, I take the minimum mass value of propositions T1, T2, T3, T4, and uncertainty, for the first two knowledge sources m1 (CW Radar) and m2 (Surveillance Radar).
ie.
min(m1T1, m2T1) = 0.33
min(m1T2, m2T2) = 0.0
min(m1T3, m2T3) = 0.33

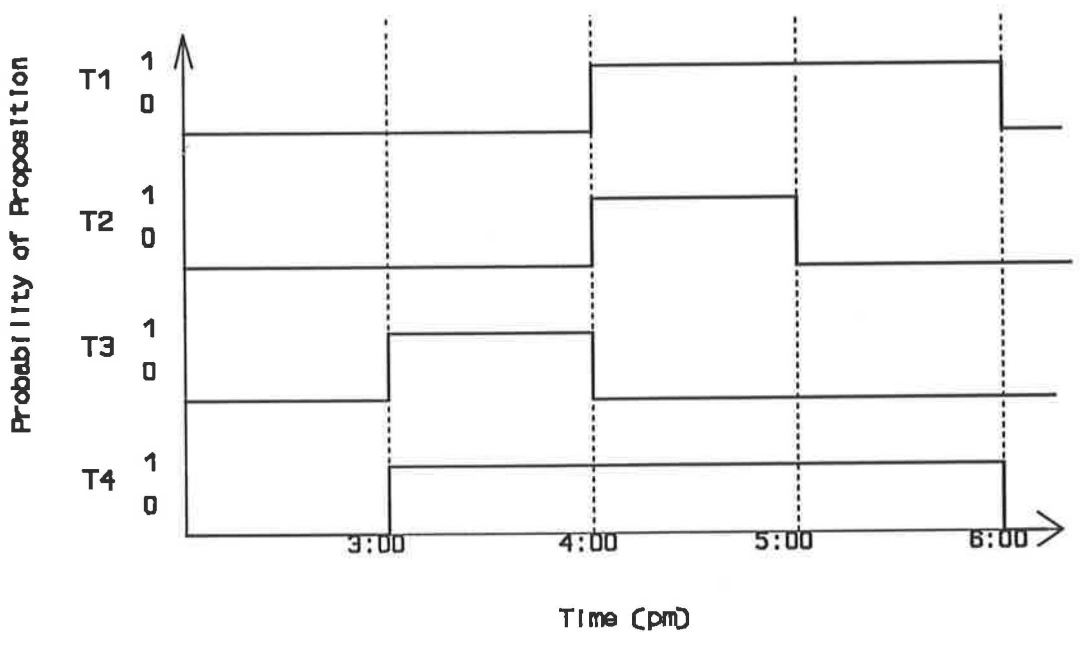


Figure 36 The curves (derived from flight timetable information) indicate the probability that any given target type (or proposition) T1,T2,T3,T4 will be flying within 40nm of Adelaide Airport at the time indicated on the x axis.

min(m1T4, m2T4) = 0.0
min(theta1, theta2) = 0.01

Normalising the resultant mass vector, I obtain the combined mass m3 ie. sum of above is 0.67, hence:-

T1 = T3 = 0.33/0.67 = 0.49
T2 = T4 = 0.0
theta = 0.01/0.77 = 0.0149

hence combined mass,

$m3 = (0.49, 0, 0.49, 0)$ theta = 0.0149;

integrating m3 with m_{priori} results in $m_{composite}$ (after normalisation)

$m_{composite} = (0, 0, 0.98, 0)$ theta = 0.02

Hence proposition T3 (Unknown jet) is the most likely target (0.98). Which compares favourably with the D-S method (0.952), and neural network method (0.9524), except for the fact that theta has increased in value, ie the uncertainty has increased when fusing **m3** with m_{priori} . Also by taking the minimum and normalising the mass vectors continuously we can lose the resultant masses of the remaining propositions. But the minimum method has the advantage over D-S in the reduced number of calculations needed. However, as mentioned, the uncertainty calculation is unsatisfactory. In the next paragraph, I will introduce and discuss the alternative to uncertainty using Fuzzy Reasoning (called Entropy) and calculate the uncertainty (Entropy) for target example 1.

TARGET N0.	**DEMPSTER SHAFER & FUZZY REASONING UNNORMALISED INPUTS FOR THE 4 PROPOSITIONS**		
	CW RADAR	**SURV. RADAR**	**FLIGHT TIMETABLE**
1	99.98,99.98,99.98,.02	99.01,0,99.01,99.01	0,0,1,1
2	99.94,99.94,99.94,.06	99.68,0,99.68,99.68	1,1,0,1
3	0.32,0.32,0.32,99.68	43.3,94.74,94.74,94.74	0,0,1,1
4	0.29,0.20,0.20,99.71	00.19,0,99.85,99.85	0,0,1,1
5	0.92,0.92,0.92,99.08	0.42,0,99.7,99.7	1,1,0,1
6	5.0,5.0,5.0,95.0	0,0,99.93,99.93	1,1,0,1
7	16.08,16.08,16.08,83.92	99.74,0,99.74,99.74	0,1,0,1
8	99.98,99.98,99.98,.02	33.7,0.0,49.79,49.79	1,1,0,1

Table 15 Raw mass assignments for 4 propositions (T1,T2,T3,T4) derived from table 10 and flight timetable information (Fig. 35).

Target No.	DEMPSTER SHAFER & FUZZY REASONING INPUT VECTORS (normalised) FOR THE 4 PROPOSITIONS (ASSUME INITIAL UNCERTAINTY OF 1% FOR THE 3 KNOWLEDGE SOURCES).		
	CW RADAR	SURV. RADAR	FLIGHT TIMETABLE
1	.33,.33,.33,0	.33,0,.33,.33	0,0,.495,.495
2	.33,.33,.33,1.9E-4	.33,0,.33,.33	.33,.33,0,.33
3	.003,.003,.003,.98	.1309,.286,.286,.286	0,0,.495,.495
4	.003,.003,.003,.98	9.4E-4,0,.495,.495	0,0,.495,.495
5	.009,.009,.009,.96	.0021,0,.494,.494	.33,.33,0,.33
6	.045,.045,.045,.85	0,0,.495,.495	.33,.33,0,.33
7	.12,.12,.12,.628	.33,0,.33,.33	0,0,.495,.495
8	.33,.33,.33,6.6E-5	.25,0,.3698,.3698	.33,.33,0,.33

Table 16 Normalised mass assignments for the 4 propositions (ie Darwin jet, Perth jet, Unknown jet, Propeller Aircraft) derived from the outputs of NN1 & NN2 (Table 10) and flight timetable prior information. And used as input vectors for Dempster Shafer & Fuzzy Reasoning. (Assuming an initial uncertainty of 1% (.01) from the 3 knowledge sources).

5.8.2 Fuzzy Reasoning (Entropy) [34]

Fuzzy variables are based upon fuzzy set theory of Zadeh [24,25], which is used to represent uncertainty. If the field of discourse "Y" has a variable "y" in a fuzzy set "A", then "y" has a membership function $\mu(y)$ in the unit interval [0,1]. Crisp sets have membership values that are either 1 or 0. Operations on fuzzy sets corresponding to logical AND, OR and NOT are defined by:-

$$A \text{ AND } B = A \cup B = \{ (y, \min(\mu_A(y), \mu_B(y))) \},$$
$$A \text{ OR } B = A \cap B = \{ (y, \max(\mu_A(y), \mu_B(y))) \} \text{ and}$$
$$A' = \{ (y, 1-\mu_A(y)) \}$$

Entropy measures the uncertainty of a system, and fuzzy entropy represents the uncertainty of the fuzzy set. Entropy is defined as follows [23] :-

$$E(A)=M(A\cap A')/M(A\cup A')$$

where M(A) is the fuzzy count (= $\Sigma\mu_A(y_i)$). Hence E(A) varies between certainty (=0) and maximum uncertainty (=1).

The normalised outputs of NN1 and NN2 can be used to represent the membership function for each target type or proposition (T1, T2, T3, T4). Probability represent a special case of fuzziness [23]. For example the fuzzy variable (jet/propeller aircraft) range , can be defined to have a membership of T1(jet/propeller aircraft) given by the normalised output of NN1 (between 0 and 1). Thus the probability mass values outputs from NN1 and NN2 can also be used for fuzzy reasoning.

For our target type example, fuzzy "rules" can be stated by:-

If jet/propeller aircraft range value from NN1 is T_N(jet/propeller) and flight path range values (in the flight paths Darwin, Perth or Unknown) from NN2 is T_N(flight path) and flight lockon time range is T_N(lockon time) the target is T_N.

Where T_N, N=1,2,3,4 represents the membership function to each of the target types.
For the fuzzy rule, T_N, has a value range 0-1.
Using example 1:-

m1=jet/propeller ={(T1,0.33), (T2,0.33), (T3,0.33), (T4,0)} theta1=.01
m2=flight path = {(T1,0.33), (T2,0), (T3,0.33), (T4,0.33)} theta2=.01
mpriori= lockon time = {(T1,0), (T2,0), (T3,0.495), (T4,0.495)} theta3=0.01

Calculate the entropy of each fuzzy proposition:-

sum of T1 ... T4 for m1 = 0.99
sum of T1 ... T4 for m2 = 0.99
sum of T1 ... T4 for mpriori = 0.99

sum of the complement of T1 ... T4 for m1 = 3.01
sum of the complement of T1 ... T4 for m2 = 3.01

Entropy of m1= 0.99/3.01 = 0.328

Entropy of m2= 0.99/3.01 = 0.328

Using fuzzy rules, we combine m1 and m2 to obtain the entropy of m3

Target type = (jet/propeller) $\cap$ (flight path)
= {(T1,0.33), (T2,0), (T3,0.33), (T4,0)}
sum1 = 0.66

The crisp value is the maximum , and so T1 and T3 are the most likely target types with an entropy value of 0.1967.

ie. **Entropy of m3 = sum1/complement of above**
= (0.33+0+0.33+0)/(0.67+1+0.67+1) = 0.1976

Using the prior knowledge we have

Target type = (jet/propeller) $\cap$ (flight path) $\cap$ (lockon time)
= {(T1,0), (T2,0), (T3,0.33), (T4,0)}

With the crisp value giving T3 as the chosen target type with an Entropy for $m_{composite}$ = 0.0899

ie. **Entropy of $m_{composite}$ = (0 + 0 + 0.33 + 0)/(1 + 1 + 0.67 + 1) = 0.0899**

When fusing m3′ with $m_{composite}$, the uncertainty using D-S, reduced from 4.2E-4 to 1.63E-5, and for fuzzy from 0.1976 to 0.0899. So the uncertainty (using fuzzy reasoning) of the combination has also been shown to reduce in a consistent manner as in D-S. (Appendix L, shows the results in a table of uncertainty and entropy for the remaining seven targets using both D-S and Fuzzy methods).

5.9 Summary of Results

Examining the results of fusing the target attribute data from the 3 knowledge sources using the 3 data fusion methods (ie. refer to Table 9 using NN3, Table 17 using D-S and Table 18 using fuzzy reasoning) you notice that they all reach the same result on which target (proposition) is most likely for each of the 8 examples.

The obvious advantage of using the fuzzy reasoning (minimum) method is the reduced number of calculations needed. The disadvantage with the fuzzy reasoning method is that in some cases we can lose the resultant masses of the remaining propositions, due to the continuous minimisation and normalisation of the mass vectors, which may not necessarily represent the magnitude of the resultant proposition accurately. A problem with fuzzy reasoning (stated in ref.[34]) is when combining information about a particular hypothesis, the fuzzy reasoning AND will represent it by the one low value despite the existence of a number of larger values. The D-S method has a similar problem, for example, a proposition of mass of 0.9 repeated n times causing in the worst case assignment of 0.9^n, which can be quite small, when different sets of evidence are combined.

As mentioned in Kewley [34], the disadvantage of using D-S method is its reliability. Zadeh [39] questions D-S's use of normalisation to remove mass assignments to the null set. He shows that for:-

m1(a)=0.0, m1(b)=0.1, m1(c)=0.9
m2(a)=0.9, m2(b)=0.1, m2(c)=0.0

the combined result is

m3(a)=0.0, m3(b)=1.0, m3(c)=0.0

This is not consistent with the low mass assignments to proposition "b", in both probability assignments. The normalisation has concealed the contradictory aspect of the sets of evidence. Shafer's counter example [40], slightly modifies m1 and m2 so that :-

m1'(a)=0.01, m1'(b)=0.1, m1'(c)=0.89
m2'(a)=0.89, m2'(b)=0.1, m2'(c)=0.01

with the new combined result of

m3'(a)=0.32, m3'(b)=0.36, m3'(c)=0.32

Kewley [34], stated that from these examples, there is great danger in assignment of zero or very low values to a probability due to the normalisation procedure.

Also the disadvantage of using the D-S method is the number time consuming computations needed in combining mass vectors. For both D-S and Fuzzy reasoning there exists the problem of how to obtain the initial mass assignments for each of our target types, (propositions T1 ... T4), and what initial uncertainties to assign to our knowledge sources (in our example we assumed an uncertainty of 1%).

Chapter 6

Conclusion

6.1 Kinematic Data Fusion

Even though the same results were achieved in the end using both techniques (Classical Inference and ART2) with kinematic data obtained from both radars, ART2 has the following advantages over the Classical Inference technique :

It can be trained and implemented into a real time environment very quickly without knowledge of the accuracy of the sensors. Radar data can be input to the neural net in parallel to obtain a quick result. Also, as the number of targets changes, continuously in an ever changing real time environment, a new pattern category can be formed quickly that was not in the initial training set.

Strict application of the Classical Inference technique requires knowledge of an 'a priori' probability distribution which is clearly unknown in a realistic application. Thus the threshold value used (50.9) is not necessarily the best in the real world and so there might arise a situation where the same target being tracked in the common surveillance volume of both radars is identified incorrectly as being two different target tracks (ie. tracks are not the same, the alternative hypothesis H0 is accepted). ART2 does not require knowledge of any 'a priori' probability distribution.

In conclusion the ART2 neural network technique is better than the Classical Inference approach using Hypothesis testing as first outlined by Bar Shalom [6], because we not only avoid the time consuming computations required in the hypothesis testing stage, where the test statistic (which is the Mahalanobis distance) is summed at each point along the track and compared with the level of significance (as shown in the example calculations of the Mahalanobis distance for simulated tracks in section 4.2.1). But also the need to find the accuracy of the sensors is avoided because the network indirectly obtains this from the training data through the adjustment of the vigilance value. The vigilance determines the degree of recognition between the two tracks when the network is trained with track pairs which are known to be from the same targets. Even though the vigilance value was adjusted manually when training ART2 with tracks from both radars which were known to be the same, it wouldn't be hard to write an algorithm to adjust the vigilance value automatically.

Also to find out if there is any correlation between track pairs when using the Classical

Inference approach one has to find the possible track pair combinations (ie. track1 from radar1 with track1 from radar2, track1 from radar1 with track2 from radar2, etc, refer to Table 6). This tedious and time consuming procedure of pairing tracks is avoided using the ART2 neural network because all we do, once the network is trained, is input the tracks into the network one at a time , and the network clusters the input tracks into one of the appropriate target categories as shown in Fig. 29, in real time.

Time consuming computations as discussed previously can be crucial in a military scenario with hundreds of targets being correlated and fused by a central computer whose data is sent to weaponry (ie. Rapier missiles) used to intercept the required targets in real time.

6.2 Attribute Data Fusion

The objective was to develop and evaluate a neural network based Data Fusion system for automatic allocation of identity of airborne targets using all available information, and compare the results using trial data with the D-S and fuzzy reasoning methods. Even though the target classifications were restricted to jet or propeller driven in NN1, and to Darwin/Perth/Unknown Flight paths in NN2, the fusion of the outputs, with flight-time information, in NN3, effected a considerable improvement in the final classifications over NN1 or NN2's target classification on their own.

To achieve the objective a neural network based Data Fusion system comprising of 3 backpropagation neural networks NN1, NN2 and NN3 was evaluated (as shown in Fig. 35). NN1 and NN2 produced classifications on the identity of target types (when presented with data from a CW radar and Surveillance radar respectively). NN1 successfully identified airborne targets as either jet or propeller driven from their doppler modulation (processed using LP), and achieved a recognition accuracy of 100% for the 7 airborne targets presented. (ie. Refer to the bar graphs in figures 74 and 75, Appendix M). Results from NN1 using the limited target numbers (as shown in Appendix H, Tables 23 and 24) are very promising. However, further investigations are necessary using large number of targets before drawing definitive conclusions. Also NN2 successfully identified the 7 airborne targets as being on Perth, Darwin or unknown flight paths from their cartesian coordinates, "X" and "Y" (with respect to Adelaide Airport). The fusion centre NN3 made the overall classification based on the outputs of NN1, NN2 and flight timetable information (refer Fig. 36). As discussed in section 5.6 using the eight target examples I conclude that combining the knowledge sources gives us a better assessment of the identity of the target (in real time).

One advantage neural networks have over the other data fusion methods is that, due to a massive system of parallel processing elements, it is possible for them to process the input data relatively rapidly (after they are trained) which makes them attractive in an Automatic

Target Recognition System working in Real Time. While D-S evidential reasoning can reach a useful conclusion for the identity attribute problem, it gets there with time consuming and tedious computations, which can effect the performance of an Automatic Target Recognition system working in Real Time.

In comparing the D-S and fuzzy reasoning data fusion methods with the backpropagation neural network method NN3 (as discussed in section 5.7 and 5.8), I used input vectors obtained from the normalised mass assignments for the four propositions (ie. Darwin jet, Perth jet, unknown jet, propeller aircraft) derived from the outputs of NN1, NN2 and flight time table priori information (as shown in Table 16). From the results of fusing the target attribute data from the three knowledge sources (ie. CW radar, Surveillance radar and time table information) using the three data fusion methods (as shown in Table 9, 17, and 18 using the NN3, D-S and fuzzy reasoning methods respectively), you notice that they all reach the same final hypothesis on which target (proposition) is most likely for the eight target examples. The main difference is the slight variations in magnitude for some of the propositions, as shown in Tables 9, 17, 18. But that's not surprising because for example, using the fuzzy method, by taking the minimum and normalising the mass vectors continuously we can lose the resultant masses of the remaining propositions, hence obtain zero values as shown in Table 18. Since D-S and fuzzy methods have calculated uncertainties and entropy measurements (as shown in Appendix L), the magnitude values of the resultant propositions are not a measure of the likelihood of one occurring on its own. As in the neural network method which reflects the Bayesian approach, you have to take into account the uncertainties.

Hence I also conclude that the backpropagation neural network method is better than D-S or Fuzzy Reasoning (in the cases studied) because there is no need to know or calculate the uncertainty of your knowledge source, the network indirectly obtains this through the training data. Also by training NN1 and NN2 to classify the target's attributes directly from the sensors (CW and Surveillance radars) into simpler weighted variables (ie. jet/propeller targets; Darwin, Perth and Unknown flightpaths), which correspond to their probabilities, you can directly feed these outputs into the input of NN3 in real time. For D-S and fuzzy reasoning I use the outputs of NN1, NN2 and timetable information to derive the initial mass assignments for the 4 propositions (T1...T4). Otherwise I would have had to calculate or derive curves or data tables for the target attributes coming from the CW and Surveillance radars, which would indicate the probability that any given target type (proposition T1...T4) will have a specified value of the variables (jet/propeller; Darwin,Perth,Unknown flightpaths).

TARGET N0.	SUPPORT FOR THE FOLLOWING PROPOSITIONS USING DEMPSTER-SHAFER (as a percentage)			
	DARWIN JET	PERTH JET	UNKNOWN JET	PROP. AIRC.
1	1.8	.053	95.24	2.8
2	91.85	2.7	2.7	2.7
3	.01167	.025	1.31	98.65
4	.00015	.00011	1.28	98.72
5	.0459	.0379	.056	99.86
6	.12	0.12	.19	99.56
7	.4	0.59	0.4	98.6
8	88.93	3.41	3.81	3.84

Table 17 Indicates the support for the propositions using Dempster-Shafers (Evidential Reasoning) on the normalised outputs of NN1 & NN2 (table 3) and priori information (flight timetables).

TARGET NO.	DECISION ON WHICH TARGET IS LIKELY USING FUZZY REASONING (MINIMUM METHOD)			
	DARWIN JET	PERTH JET	UNKNOWN JET	PROP. AIR
1	0	0	98	0
2	97	0	0	0
3	0	0	1.96	96
4	0	0	1.1	96.9
5	1.19	0	0	95.9
6	0	0	0	97
7	0	0	0	98
8	97	0	0	0

Table 18 Indicates the likelihood for the propositions using Fuzzy Reasoning (Minimum method) on the normalised outputs of NN1 & NN2 & flight timetable priori information.

Appendix A (Linear Prediction)

1.0 Introduction

Linear Prediction is an aspect of time series analysis also known as least squares estimation (prediction), (dates back to 1795). The basic idea of Linear Prediction (LP) [30], [29] analysis, is that a signal sample can be approximated as a linear combination of past signals by minimising the sum of the squared differences (over a finite interval of time) between the actual signal and the linearly predicted data. Once predictor constants are computed then an all pole model can be developed to fit the data (refer to Fig. 37).

Doppler spectra of radar returns (from the CW (continuous wave) radar can be modelled using LP. In my particular application, the LP is basically acting as a smoother rather than a high spectral estimator, since I am using a small model order.

A close relationship exists between a linear prediction filter and an autoregressive process. Consider the linear prediction estimate, of sample *x[n]*, where *a[k]* (square brackets are used for sampled signals) is the linear prediction coefficient at time index *k*.

$$x'[n]=-\sum_{k=1}^{m} a[k].x[n-k]\text{-------}(12)$$

$e[n]$	is the error between predictor and actual sample.
m	is the model (predictor) order
$x'[n]$	is the predictor sample
$a[k]$	is the predictor constants.

The " ′ " is used to denote an estimate, and the prediction is forward in the sense that the estimate at time index *n* is based on *m* samples indexed earlier in time. The complex forward linear prediction error is:

$$e[n]=x[n]-x'[n]\text{---------}(13)$$

$$e[n]=x[n]+\sum_{k=1}^{m} a[k].x[n-k]\text{---}(14)$$

Expressing the error equation (14) in the "z" domain (z is the complex operator) we have:-

$$E(z)=x'(z)[1-\sum_{k=1}^{m} a[k].z^{-k}]------(15)$$

If the predictor order *m* is sufficiently large, then substantially all correlation is removed from the error *e[n]*, and this yields a white (constant spectrum) signal *x'[n]*

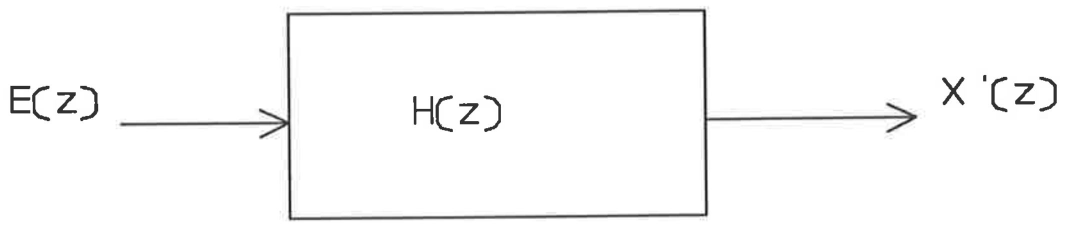


Figure 37 All pole model developed to fit data.

$$H(z)=\frac{x'(z)}{E(z)}$$

$$H(z)=\frac{1}{1-\sum_{k=1}^{m} a[k].z^{-k}}------(16)$$

As *m* becomes large the variance ρ of the error *e[n]* becomes small, so *1/H(z)* approximates the signal spectrum. *H(z)* is also known as the signal model; the denominator of *H(z)*

$$1-\sum_{k=1}^{m} a[k].z^{-k}----(17)$$

is called the inverse filter. The structure of such a filter is given by the following Figure 38.

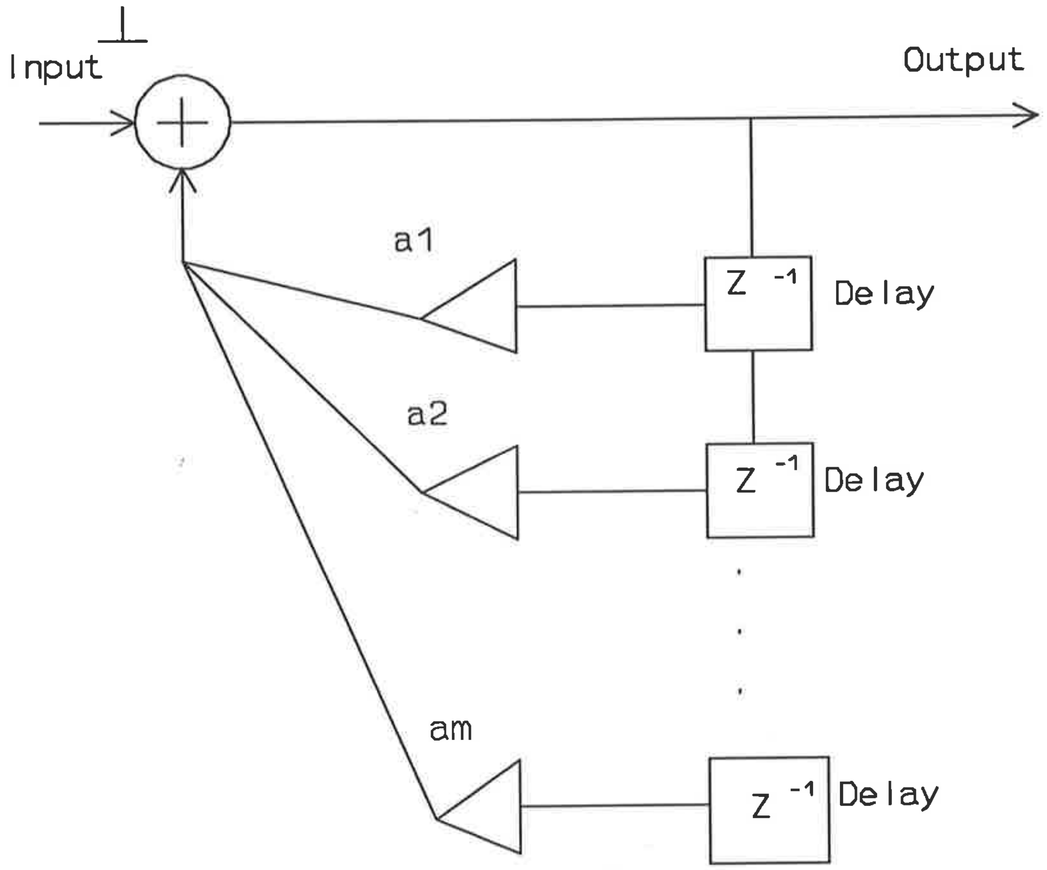


Figure 38 : Digital structure of Generating Filter

2.0 Relationship of the Real Variance (ρ) to LP Analysis

The complex forward linear prediction error show in equation 13 has real variance :

$$\rho = \mathrm{E}\,|e[n]|^2 \text{---------} (18)$$

"E" denotes expectation. Substituting equations 12 and 13 into equation 18 yields expression 19 .

$$\rho = \mathrm{E}\,|e[n]|^2 = \mathrm{E}(x[n] + \sum_{k=1}^{m} a[k].x[n-k])^2$$

$$\rho = \mathrm{E}((x[n] + \sum_{j=1}^{m} a[j].x[n-j].(x[n]^* + \sum_{k=1}^{m} a[k]^*.x[n-k]^*))$$

$$\rho=(x[n].x[n]^*+\sum_{k=1}^{m}x[n].a[k]^*.x[n-k]^*+\sum_{j=1}^{m}a[j].x[n-j].x[n]^*+\sum_{j=1}^{m}a[j].x[n-j].\sum_{k=1}^{m}a[k]^*.x[n-k]^*)$$

$$\rho=r_{xx}[0]+\sum_{k=1}^{m}a[k]^*.r_{xx}[k]+\sum_{j=1}^{m}a[k].r_{xx}[-j]+\sum_{j=1}^{m}\sum_{k=1}^{m}a[j].a[k]^*.r_{xx}[k-j]$$

rearranging

$$\rho=r_{xx}[0]+\sum_{j=1}^{m}a[k].r_{xx}[-j]+\sum_{k=1}^{m}a[k]^*.r_{xx}[k]+\sum_{j=1}^{m}\sum_{k=1}^{m}a[j].a[k]^*.r_{xx}[k-j]$$

assuming *x[n]* is a wide sense stationary process so $r_{xx}[-k]=r_{xx}[k]^*$

$$\rho=r_{xx}[0]+\sum_{j=1}^{m}a[k].r_{xx}[j]^*+\sum_{k=1}^{m}a[k]^*.r_{xx}[k]+\sum_{j=1}^{m}\sum_{k=1}^{m}a[j].a[k]^*r_{xx}[k-j]$$

Rewriting above in a matrix format (getting rid of summation (sigma) terms)

$$\rho=r_{xx}[0]+r_m^a.a+(a)^H.r^m+a^H.R_{m-1}.a\text{-------}(19)$$

Where *a, rm,* and R_{m-1} are as follows (*H* is the complex transpose)

$$a=\begin{bmatrix}a[1]\\a[2]\\ \cdot\\ \cdot\\ \cdot\\a[m]\end{bmatrix},\quad rm=\begin{bmatrix}r_{xx}[1]\\r_{xx}[2]\\ \cdot\\ \cdot\\ \cdot\\r_{xx}[m]\end{bmatrix},\quad R_{m-1}=\begin{bmatrix}r_{xx}[0] & \cdots & r_{xx}[m-1]^*\\ \cdot & & \\ \cdot & & \\ \cdot & & \\ r_{xx}[m-1] & \cdots & r_{xx}[0]\end{bmatrix}$$

The expression (19) is identical to the quadratic equation shown in equation (3.68) of Ref [5] page 69. Therefore the linear prediction coefficient vector *a* that minimises the variance ρ is found as the solution to the normal equation (Refer to Reference [29] chapt. 3.5 pages 69-71, for theory into "least squares normal equations") is given by :-

$$\begin{bmatrix} r_{xx}[0] & r_m^H \\ r_m & R_{m-1} \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a \end{bmatrix} = \begin{bmatrix} \rho \\ 0_m \end{bmatrix}$$

Or

$$\begin{bmatrix} r_{xx}, & r_{xx}[1]^*, & \dots & r_{xx}[m]^* \\ r_{xx}[1], & r_{xx}[0], & \dots & r_{xx}[m-1] \\ r_{xx}[m], & r_{xx}[m-1], & \dots & r_{xx}[0] \end{bmatrix} \cdot \begin{bmatrix} 1 \\ a[1] \\ . \\ . \\ . \\ . \\ a[m] \end{bmatrix} = \begin{bmatrix} \rho \\ 0 \\ 0 \\ . \\ . \\ . \\ 0 \end{bmatrix} \text{-----(20)}$$

Note in equation (20) as shown above *a[0]=1* in the *"a"* vector ie.

$$a = \begin{bmatrix} a[0] \\ a[1] \\ . \\ . \\ . \\ a[m] \end{bmatrix}, \; so \;\; p(z) = \frac{1}{\sum_{i=0}^{m} a_i . z^{-i}}$$

and not as shown in equation (15), ie.

$$p(z) = \frac{1}{1 - \sum_{i=1}^{m} a_i . z^{-i}} \;, \; with \;\; a = \begin{bmatrix} a[1] \\ a[2] \\ . \\ . \\ . \\ a[m] \end{bmatrix}$$

3.0 Steps used to Formulate Matlab Program used to Process Doppler Data

We fit a filter to the Continuous Wave Doppler Radar Data stored in array 0. ie. we null out the noise spikes to find where the frequencies are (refer to Fig. 39).
(Array 0 is a matlab file, obtained from the CW Radar, containing 32k bytes or one tenth of a second of Real data at a certain aspect angle of the target in question).

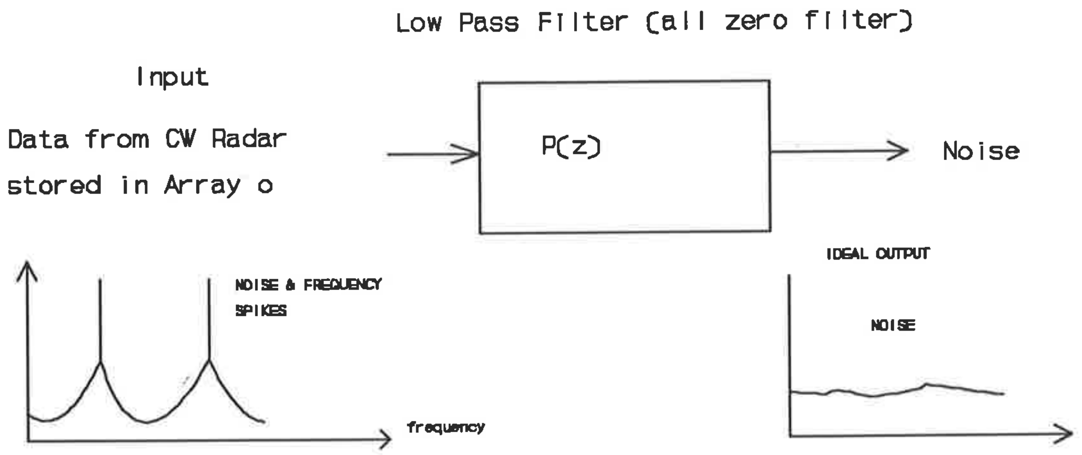


Figure 39 Fit a filter to the CW radar data.

We are calculating the coefficients of the linear prediction coding $a=R^{-1}.rm$

Now that we have a filter we generate a spectrum of the filter
And if we plot $p(z)$ we get the following (nulls or zeroes, get rid of spikes, Fig. 40).

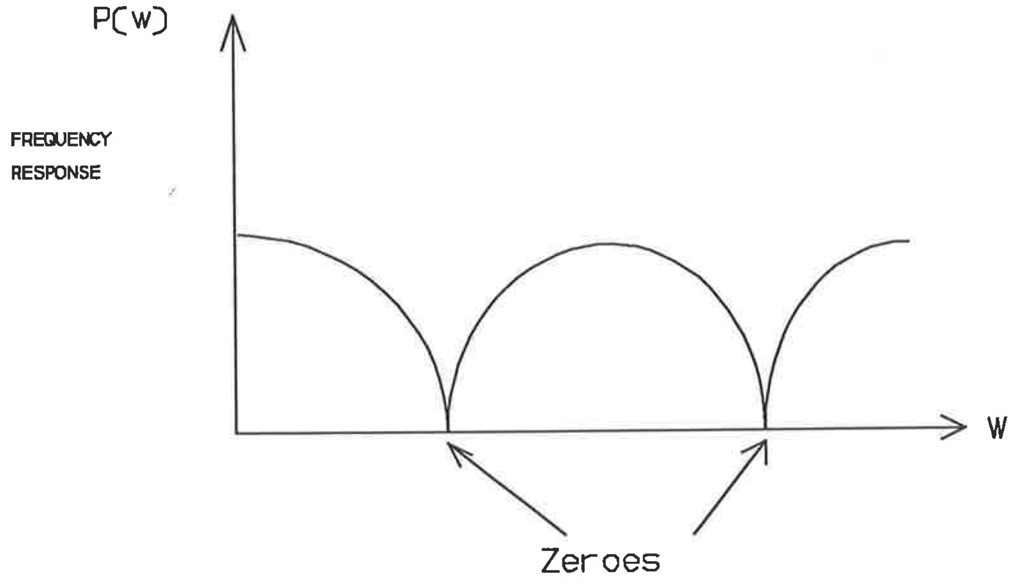


Figure 40

We want the poles where the zeroes are, so the peaks correspond to the sinusoids of the input data, so we take the inverse of *p(z)*, which gives us poles (refer to Fig. 41). The spectrum of the all pole filter (*1/p(z)*) is an estimate of the spectrum of the data. (ie. FFT of the LPC).(The impulse response of *P(z)* in the z domain is the LP coefficients "*a*", so to get the frequency region of *P(z)* we take the fourier transform.)

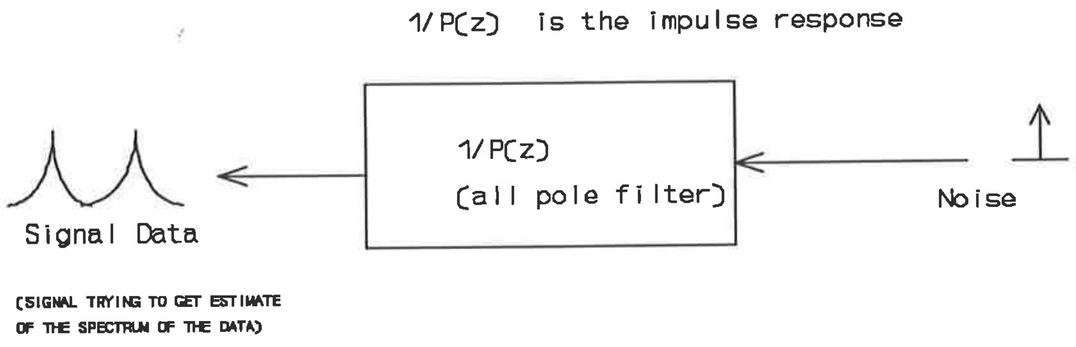

1/P(z) is the impulse response
1/P(z)
(all pole filter)
Noise
Signal Data
(SIGNAL TRYING TO GET ESTIMATE OF THE SPECTRUM OF THE DATA)

Figure 41 Taking the inverse of p(z).

Note, since in my application I am using a small model order the LPC is basically acting as a smoother rather than a high resolution estimator.

4.0 Matlab Programming Steps

Step 1: Array0 is a 4096 x 1 array vector containing the real part of the doppler modulation data (32k bytes, 1/10 th. of a second) obtained from the CW Radar for the target being tracked at a certain aspect angle. Array0 has to be arranged in the form accepted by the "*COV(X)*" Matlab function. *COV(X)* computes the covariance matrix of "*X*", ie.

$$array0=\begin{bmatrix} X1 \\ X2 \\ . \\ . \\ . \\ X4096 \end{bmatrix}, \quad X=\begin{bmatrix} X1 & X2 & X3 & . . . & X4085 \\ . & & & & \\ . & & & & \\ . & & & & \\ X12 & X13 & X14 & . . . & X4096 \end{bmatrix}$$

```
for i=1:12
x(i,:)=array0(i:4096-12+i);
end
```

Step 2: Calculate the covariance (where x' , is the transpose of x), and define *rm* (where *rm* is the variance vector as shown below). The number of zeroes in *rm* is proportional to the number of points you want to use as the history of your input data (to predict the next point, ie. to get 5 output peaks in the output spectrum you need 10 zeroes). Since a small model order of ten was used (to produce the smoothing effect) to suit my application the LPC is basically acting as a smoother rather than a high resolution estimator (the results of the processing can be seen in the plots Appendix H).

```
r1=cov(x');
rm=[1;0;0;0;0;0;0;0;0;0;0;0];
```

Step 3: Fitting a filter to the data by calculating the linear prediction coefficients in array, "*a*". Where "*inv(r1)* " is the inverse of vector "*r1*".

```
a=inv(r1).rm;
```

Step 4: The spectrum of the filter is an estimate of the spectrum of the data (ie. Plotting FFT of LP coefficients).

```
p=plot(-log10(abs(fft(a,4096))));
```

Note the reason why the equation above is "*(fft(a,4096)...* " and not "*(1-fft(a,4096)...*" is shown in the last few lines of paragraph 2 (it depends on how you define your "a" vector, in our case a0=1).

Step 5: Shift the LPC data by subtracting by its minimum , so that all that all plots can be compared on a common relative scale.

```
q=p-min(p);
```

Appendix B (Simulated track data tables)

		TIME (SECONDS)									
		1	2	3	4	5	6	7	8	9	10
T1 R1	R	1.21	22.67	34.9	50.2	66.86	68.47	77.5	81.6	85.8	95.7
	AZ	1.57	30.07	38.02	55.2	80.82	108.1	129.7	146.8	169.4	177.
	EL	1.489	12.67	24.6	36.19	41.91	52.86	68.89	79.5	80.14	88.4
T1 R2	R	3.3	26.75	37.16	52.19	70.12	71.4	80.5	85.9	89.77	99.2
	AZ	1.71	29.8	37.7	55.7	80.01	107.8	128.5	146.4	168.0	176.
	EL	2.97	12.43	24.89	36.6	42.02	53.76	68.8	79.9	81.6	88.7
T1 R1	R	1.219	22.04	34.67	50.67	66.38	68.38	77.52	81.83	85.03	95.0
	AZ	2.05	29.18	38.3	55.83	81.4	108.8	129.5	146.5	160.1	177.
	EL	1.07	12.6	24.88	36.27	41.43	52.76	68.17	79.23	80.22	88.3
T2 R2	R	14.3	28.7	32.25	39.8	48.85	57.14	67.63	60.07	53.44	82.3
	AZ	15.8	18.41	22.84	43.27	51.41	64.54	75.47	81.29	91.18	155.
	EL	26.68	29.31	30.42	38.43	37.26	39.18	50.55	59.0	60.02	61.0
T1 R1	R	1.21	22.04	34.68	50.67	66.38	68.38	77.5	81.8	85.02	95.0
	AZ	2.05	29.18	38.31	55.83	81.4	108.8	129.5	146.5	168.1	177.
	EL	1.07	12.63	24.88	36.27	41.43	52.76	68.17	79.24	80.3	88.3
T3 R2	R	89.32	82.7	71.25	68.8	63.85	56.14	48.62	39.66	25.4	13.3
	AZ	175.8	166.4	154.8	148.2	139.4	130.5	122.4	116.2	70.17	38.1
	EL	89.68	77.32	56.42	45.42	55.26	67.18	51.55	48.0	30.8	12.0
T2 R1	R	12.49	24.26	32.09	36.95	45.07	53.5	67.38	60.27	50.9	78.5
	AZ	16.25	19.47	23.45	44.99	52.77	64.46	75.61	81.7	92.02	156.
	EL	25.57	29.8	30.03	37.53	37.5	39.96	50.75	59.55	60.89	61.6
T1 R2	R	3.31	26.75	37.16	52.19	70.12	71.45	80.5	85.93	89.77	99.2
	AZ	1.71	29.8	37.7	55.7	80.02	107.8	128.5	140.4	168.1	176.
	EL	2.89	12.43	24.89	36.63	42.03	53.76	68.88	79.9	81.6	88.7

Table 19 Simulated track data results (in range, azimuth & elevation) for the track pairs (T1R1, T1R2), (T1R1, T2R2), (T1R1, T3R2), and (T2R1, T1R2) over the 10 second time period.

		TIME (SECONDS)									
		1	2	3	4	5	6	7	8	9	10
T2 R1	R	12.2	24.04	32.67	36.67	45.93	53.3	67.5	60.03	50.03	78.0
	AZ	16.05	18.18	23.31	43.83	52.4	65.82	76.52	81.52	91.09	156.
	EL	25.07	29.63	30.88	37.27	32.43	39.76	50.47	59.24	60.27	61.3
T2 R2	R	14.65	27.35	32.04	37.92	45.33	55.08	70.43	62.94	54.65	82.2
	AZ	15.33	18.63	22.75	43.99	51.36	64.25	75.98	81.72	91.75	155.
	EL	25.33	29.97	31.79	30.82	37.12	40.81	51.0	60.03	60.64	62.9
T2 R1	R	12.49	24.26	32.09	36.94	45.07	53.5	67.38	60.22	50.91	78.5
	AZ	16.26	14.47	23.45	44.99	52.77	64.46	75.61	81.7	92.03	156.
	EL	25.57	29.8	30.03	37.53	37.49	39.95	50.74	59.55	60.07	61.6
T3 R2	R	89.32	82.7	71.25	68.8	63.86	56.13	48.62	39.07	25.44	13.3
	AZ	175.8	166.4	154.8	148.2	139.4	130.5	122.4	116.2	70.18	38.1
	EL	89.68	77.32	56.42	45.43	55.2	67.18	51.54	48.0	30.82	12.0
T3 R1	R	18.22	38.05	46.68	55.68	67.94	82.38	59.52	44.83	33.03	21.0
	AZ	49.05	82.18	86.31	98.83	124.4	146.8	159.5	166.5	171.1	178.
	EL	3.07	7.6	12.88	16.27	24.43	28.76	30.47	31.24	27.17	26.3
T1 R2	R	3.31	26.76	37.16	52.19	70.12	71.4	80.5	85.9	89.77	99.2
	AZ	1.72	29.8	37.7	55.7	80.02	107.8	128.5	146.4	168.1	176.
	EL	2.98	12.43	24.89	36.63	42.03	53.76	68.88	79.93	81.61	80.7
T3 R1	R	18.2	38.06	46.69	55.68	67.94	82.38	59.53	44.8	33.0	21.1
	AZ	49.05	82.18	86.31	98.83	124.4	146.8	159.5	166.5	171.1	178.
	EL	3.07	7.63	12.88	16.27	24.43	28.76	30.47	31.24	27.17	26.3
T2 R2	R	14.32	28.7	32.25	39.81	48.85	57.14	67.63	60.08	53.44	82.3
	AZ	15.85	18.41	22.84	43.27	51.41	64.55	75.47	81.29	91.18	155.
	EL	26.68	29.32	30.42	30.43	37.26	39.18	50.55	59.0	60.8	61.0

Table 20 Simulated track data results (in range, azimuth & elevation) for the track pairs (T2R1, T2R2), (T2R1, T3R2), (T3R1, T1R2), and (T3R1, T2R2) over the 10 second time period.

		TIME (SECONDS)									
		1	2	3	4	5	6	7	8	9	10
T3 R1	R	18.22	38.05	46.67	55.68	67.94	82.38	59.53	44.83	33.03	21.0
	AZ	49.05	82.18	86.31	98.83	124.4	146.8	159.5	166.5	171.0	178.
	EL	3.07	7.63	12.88	16.27	24.46	28.76	30.47	31.23	27.17	26.3
T3 R2	R	89.32	82.7	71.2	68.8	63.85	56.14	48.62	34.08	25.44	13.3
	AZ	175.8	166.4	154.8	148.3	139.4	130.5	122.5	116.2	70.17	38.1
	EL	89.68	77.3	56.42	45.4	55.26	67.18	51.54	48.0	30.8	12.0

Table 21 Simulated track data results (in range, azimuth and elevation) for track pairs (T3R1, T3R2) over the ten second time period.

Appendix C (Program used to simulate tracks T1R1,T1R2 & T2R1,T2R2)

1.0 Matlab Program

The following matlab program was used to simulate track pairs from the two radars (radar1 R1, and radar2 R2), with the noise variances indicated in range, azimuth and elevation . Assume the radars are independent and are tracking the same targets in an overlapping surveillance volume for a time period of ten seconds. Assume simulated tracks from both radars have a common space/time co-ordinate graph system as shown in Fig. 24.

Radar 1 has the following variances:

standard deviation in Range for Radar 1	= 1 metre
standard deviation in Azimuth for Radar 1	= 2 degrees
standard deviation in Elevation for Radar 1	= 3 degrees

and Radar 2 has the following variances:

standard deviation in Range for Radar 2	= 3 metres
standard deviation in Azimuth for Radar 2	= 4 degrees
standard deviation in Elevation for Radar 2	= 1 degree.

Corresponding covariance matrices for Radar 1 & Radar 2 respectively :-

$$P1=\begin{bmatrix}1 & 0 & 0\\ 0 & 4 & 0\\ 0 & 0 & 9\end{bmatrix},\quad P2=\begin{bmatrix}9 & 0 & 0\\ 0 & 16 & 0\\ 0 & 0 & 1\end{bmatrix}$$

Matlab Program is as follows:-

```
r=[1 23 34 50 66 68 77 81 85 95]       % Range   track data for 10 time intervals.
a=[1 29 37 55 80 107 128 146 168 176] % Azimuth  "   "   "   "   "      "
e=[1 12 24 36 41 52 68 79 80 88]       % Elevation "   "   "   "   "      "

rand('normal')             % Normal distribution

% Assuming normal distribution (using random number generator "rand")
% We are creating noise with standard deviations shown below.

noise1=1*rand(1,10)        % noise1 used for range, radar1
noise2=3*rand(1,10)        % noise2 used for range, radar2
noise3=2*rand(1,10)        % noise3 used for azimuth, radar1
noise4=4*rand(1,10)        % noise4 used for azimuth, radar2
noise5=3*rand(1,10)        % noise5 used for elevation, radar1
noise6=1*rand(1,10)        % noise6 used for elevation, radar2

% Adding noise to Range, Azimuth & Elevation, creating new tracks from radar 1 (r1, a1,
% el1) and  radar 2 (r2, a2, el2).

r1= r+noise1          % Adding noise to Range, creating a new track from radar 1.
r2= r+noise2          % Adding noise to Range, creating a new track from radar 2.
a1= a+noise3          % Adding noise to Azimuth, creating a new track from radar1.
a2= a+noise4          % Adding noise to Azimuth, creating a new track from radar2.
el1=e+noise5          % Adding noise to Elevation, creating a new track from radar1.
el2=e+noise6          % Adding noise to Elevation, creating a new track from radar2.

p1=[1 0 0;0 4 0;0 0 9]    % Covariance matrices for radar1 and radar2 (for simulated
                          % scenario),
p2=[9 0 0;0 16 0;0 0 1]   % assuming independent radars, with different accuracies in range
                          % azimuth and elevation.

s12=p1+p2              % Covariances can be added if radars independent.
is12=inv(s12)          % Get the inverse of the sum of the covariances.

pr1=10
eold=0

for i=1:pr1              % We are summing the "e" quadratic term (Mahalanobis distance)
   r=r1(1,i)-r2(1,i)     % for the 10 time instances. The Mahalanobis distance is a
   da=a1(1,i)-a2(1,i)    % measure of similarity between two tracks ( ie. the more
   de=el1(1,i)-el2(1,i)  % similar the tracks are , the smaller the "e" value.
   g=[dr;da;de]
   e1=(g)'*is12*(g)        % Converting the "g" vector into a scalar
   eold=eold+e1
end


for j= 1:pr1             % If the null hypothesis H0 is true (two tracks are the same)
                         % Tracks fused together using equation below [37].
                         % Yc=P2.(P1+P2)^-1.Y1+P1(P1+P2)^-1.Y2
```

```
                % Yc below, is a 3x1 vector containing fused range, azimuth & elevation.

 Yc=p2*is12*[r1(1,j);a1(1,j);el1(1,j)]+p1*is12*[r2(1,j);a2(1,j);el2(1,j)]   % Combined track.

r1c(1,j)=Yc(1,1)          % fused range vector over 10 second period
a1c(1,j)=Yc(2,1)          % fused azimuth vector over 10 second period
end

% Plotting range tracks r1 and r2 from radars 1 & 2, and the fused range track also
% plotting azimuth tracks a1 and a2 from radars 1 & 2, and the fused azimuth track
% over 10 second time period.

t=[1 2 3 4 5 6 7 8 9 10]
subplot(211), plot(t,r1,':',t,r2,'-',t,r1c,'*')
ylabel('RANGE')
subplot(212), plot(t,a1,':',t,a2,'-',t,a1c,'*')
ylabel('AZIMUTH')
```

Appendix D ("C" program used to process raw data from the FPS-16 Radar)

The following software written in "C" reads three raw data files containing :-
Slant Range (in 10's metres), Azimuth (as a binary fraction of 360 degrees, where 800 HEX= 180 Degrees with respect to true north) and similarly for elevation (8000 Hex =180 Degrees) with a 45 degree offset due to the position of the DSTO FPS-16 Tracking Radar. We then perform the polar to cartesian conversions necessary to align the tracks from both radars to bring them into a common space/time co-ordinate system, so they can be correlated and eventually fused, if found to be the same track from both radars. Note, since each kinematic range,azimuth and elevation value of the track is logged into the raw files every second, we need only know the "start logging time" to calculate the local time of each track value (this aids in time alignment for tracks from both radars).

```
#include <stdio.h>
#include <io.h>
#include <string.h>
#include <dos.h>
#include <math.h>
#include <stdlib.h>

#define hex_float_range   10.0        /* Range  conversion =10 metres */
#define hex_float_az      180.0/32768.0  /* Azimuth float conversion from hex, 180.0/2^15
                                         degrees) */
#define hex_float_el 180.0/32768.0      /* Elevation float conversion from hex, 180/2^15
                                         degrees) */

#define pion180          3.14159265/180

#define r1xp   7464.17     /* Position of DSTO Radar in X & Y wrt Adelaide Airport
                              Radar */
#define r1yp  25248.13   /* position in metres. */
#define r1zp  0.0

main()
{
unsigned ac,rc,ec,i;
long int rcf,acf,ecf;
float range,az,el;
float r1azrd,r1elrd,r1x,r1y,r1z,r1xout,r1yout,r1zout,r1gndra,r1tgtht;
float r1rg;
int hr=0,min=0,sec=0;
FILE *rinfp,*fopen(),*ainfp,*einfp,*stream;

rinfp=fopen("b:16range.033","rb");   /*Slant Range input binary file */
ainfp=fopen("b:16az.033","rb");      /*Azimuth     "     "    " */
einfp=fopen("b:16el.033","rb");      /*Elevation   "     "    " */

stream=fopen("b:razel4.dat","w");   /* Output file */

for(i=0;i<256;++i)                  /* Skip range, azimuth & elevation file headers */
        {
        rc=getc(rinfp);
        printf("%c",rc);
        ac=getc(ainfp);
        printf("%c",ac);
        ec=getc(einfp);
        printf("%c",ec);
        }

fprintf(stream, "X CO-ORD(m)  Y CO-ORD(m)  Z CO-ORD(m)   HEIGHT   DATA
TIME ")

 /* Initialise the time in hours, minutes & seconds */

        hr=13;
        min=51;
        sec=00;

for(i=0;i<400;++i)
        {
        /* Get Range & do conversion to float. */

        rc=getw(rinfp);
        rcf=rc;
        range=rc*hex_float_range;

        /* Get Azimuth & do conversion to float */

        ac=getw(ainfp);
        acf=ac;
        az=ac*hex_float_az;
        /* Get Elevation & do conversion to float */

        ec=getw(einfp);
        ecf=ec;
        el=(ec*hex_float_el)+45.0-360;    /* Add 45 degree offset */

        /*********************************************/
        /* Polar to Cartesian Conversions            */
        /*********************************************/

        /* Convert Azimuth and Elevation from degrees to radians */

        r1azrd=(az-1.4)*pion180;    /* Azimuth is converted to radians,           */
                                    /* and the angle is corrected by 1.4 degrees */
                                    /* for true north to grid north correction.  */

        r1elrd=el*pion180;          /* Elevation is converted to radians     */

        /* Polar to Cartesian Conversion */

        r1z=range*sin(r1elrd);      /* Z co-ordinate */
        r1rg=range*cos(r1elrd);     /* Ground range   */
        r1x=r1rg*sin(r1azrd);       /* X co-ordinate  */
        r1y=r1rg*cos(r1azrd);       /* Y co-ordinate  */

        /* Correct positions wrt origin point (Adelaide Airport) */

        r1xout=r1x+r1xp;        /* X, Y, Z positions wrt origin point (Adelaide Airport) */
        r1yout=r1y+r1yp;
        r1zout=r1z+r1zp;


fprintf(stream,"\n%f,  %f,  %f,  %f,  %d",r1xout,r1yout,r1zout,r1tgtht,i);/* Print X,Y,Z */
                                                          /* & height value to output file */


        /************************************/
        /*     Time calculations            */
        /************************************/

        ++sec;
        if(sec>59)
        {
```

```
++min;
sec=0;
}
        if(min>59)
        {
        ++hr;
        min=0;
        }
fprintf(stream,"  %d :  %d  :%d" hr,min,sec); /* Print local time to output file */

}
fclose(stream);

}
```

Appendix E (Real track data in Cartesian Co-ordinates)

		Time (seconds)									
		1	2	3	4	5	6	7	8	9	10
T1 R1	-X	14.78	14.87	14.96	15.07	15.16	15.26	15.3	15.35	15.45	15.5
	Y	5.44	5.59	5.73	5.88	6.03	6.176	6.256	6.337	6.49	6.63
	Z	.936	.944	.952	.963	.970	.987	.988	.989	.996	1.01
T1 R2	-X	13.87	13.97	14.07	14.17	14.27	14.37	14.47	14.57	14.67	14.7
	Y	5.271	5.428	5.58	5.74	5.9	6.05	6.216	6.373	6.531	6.68
	Z	.936	.944	.952	.96	.969	.977	.986	.994	1.002	1.01
T2 R2	-X	8.059	8.08	8.1	8.12	8.14	8.16	8.19	8.21	8.23	8.25
	Y	39.53	39.5	39.47	39.44	39.40	39.27	39.34	39.31	39.28	39.2
	Z	.512	.515	.518	.521	.524	.527	.53	.533	.536	.539
T3 R2	-X	58.48	58.49	58.5	58.52	58.53	58.54	58.55	58.56	58.57	58.5
	Y	65.36	65.4	65.46	65.51	65.56	65.61	65.66	65.7	65.75	65.8
	Z	1.01	1.007	1.005	.996	.988	.98	.972	.964	.955	.947
T4 R2	X	25.75	25.71	25.68	25.64	25.6	25.57	25.53	25.49	25.45	25.4
	-Y	22.07	22.04	22.01	21.97	21.94	21.91	21.88	21.85	21.82	21.8
	Z	1.493	1.493	1.49	1.49	1.49	1.49	1.49	1.49	1.49	1.49
T5 R2	X	13.09	13.14	13.20	13.25	13.3	13.36	13.41	13.47	13.52	13.5
	Y	22.29	22.33	22.37	22.41	22.45	22.49	22.53	22.57	22.61	22.6
	Z	.914	.914	.914	.914	.914	.914	.914	.914	.914	.914

Table 22 Real track data from the DSTO Radar (T1R1), and from Adelaide Airports Surveillance Radar (T1R2, T2R2, T3R2, T4R2, T5R2). T1R1 and T1R2 are tracks , are the same target being tracked by both radars.

Appendix F (CW Doppler Radar and Spectra)

1.0 CW Radar [35]

When a radar signal is reflected off a moving target the frequency is changed. This is called the Doppler effect which allows the velocity to be estimated but, more significantly, it means that returns from unwanted stationary objects such as ground, vegetation, buildings can be filtered out. This process of clutter rejection leads to the radar's strong capability to detect moving targets. The very different characteristic Doppler signature of various types of targets such as marching men, jet engines, propeller driven aircraft, helicopters etc., allows target classification to be carried out.

High resolution spectrum analysis of received doppler signals can reveal a family of modulation sidebands around the airframe line with significant components as far as 10 kHz from the airframe line. The modulation components, which are symmetric about the airframe line in frequency but asymmetric in amplitude, are caused by rotating machinery and are dependent upon engine rpm, and the number of propeller, compressor or turbine blades.

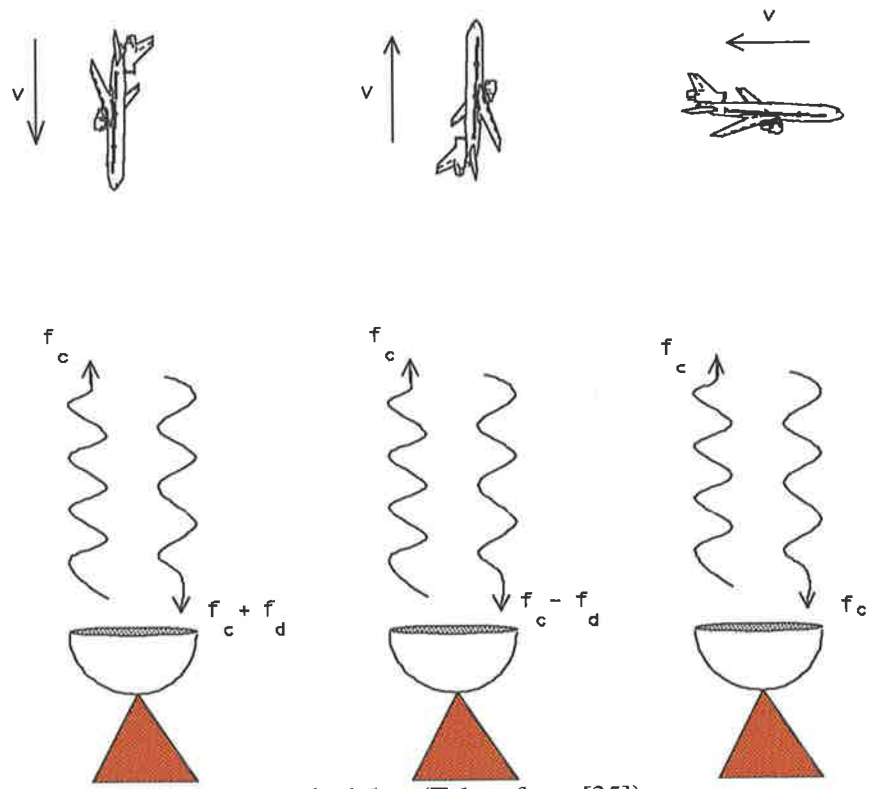


Figure 42 Doppler frequency principle . (Taken from [35]).

Figure 42 shows that an approaching target will increase the radar frequency f_c by the Doppler frequency f_d whilst a target receding at the same velocity will reduce it by the same amount. For a target moving circumferentially around the radar with no radial component the Doppler frequency is zero. The Doppler frequency is given by:

$$f_d=\frac{2.v}{\lambda}$$

Where v is the target velocity radially inwards and λ is the radar wavelength. For a radar with λ=30mm (corresponding to f_c=10 Hz), a vehicle moving towards the radar at 13.4 m/s gives a Doppler frequency of about 900 Hz. An aircraft flying at 660 m/s gives a Doppler shift of 44 kHz on the same radar.

To extract the moving target Doppler frequency from the stationary clutter returns, the radar echo must be mixed with a signal at the original radar frequency. Figure 43 shows how this is done in a continuous wave radar. The radar transmits at f_c continuously by use of a circulator. The circulator passes radar energy from one connection to that on its right whilst leaving that on its left isolated. Thus the energy from the oscillator goes direct to the antenna and not to the receiver where its high power levels would damage the receiver. The radar return is passed from the antenna direct to the receiver. The radar return consists of the incoming target echo at f_c+f_d together with energy reflected from the stationary ground or rain clutter at f_c.

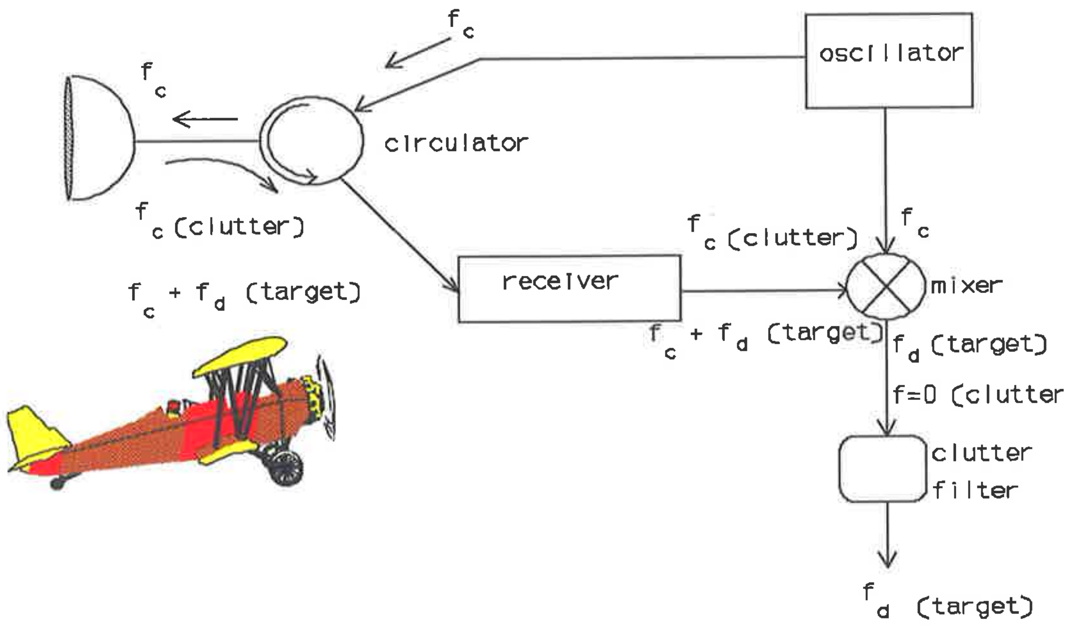


Figure 43 CW Radar.(Taken from [35]).

These signals are received and pass to the mixer. The mixer effectively subtracts the frequencies of the two signals. The output thus consists of $(f_c+f_d)-f_c=f_d$ for the target signal and $f_c-f_c=0$. Thus the clutter is suppressed whilst the target signal can be measured to establish the speed or to classify the target type. Figure 44 shows the transmission characteristic of a typical high pass filter. It is clear that it only passes the high Doppler frequencies and has a gradual cut off down to zero Doppler. In fact clutter will posses a range of Doppler frequencies due to, for example, a wind blown tree movement or the motion of rain. The filter suppresses these whilst passing target Doppler.

The rich content of harmonics in Doppler spectrum from moving targets makes identification possible. Figures 30 & 32, (obtained in trials from an X Band CW Radar) illustrate this by comparing the spectra from a propeller aircraft and a jet. The Doppler frequency due to the airframe motion is clearly seen in both cases. However, the Doppler tones due to the propellers and compressor blades are clearly different. The automation of this classification (after processing) has been illustrated using neural networks.

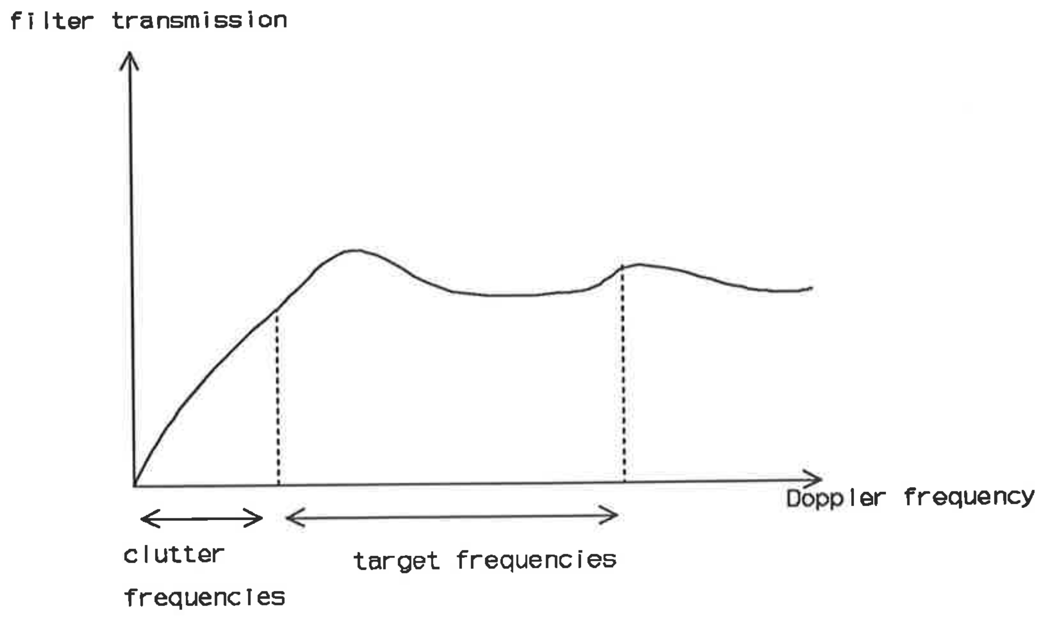


Figure 44 Transmission characteristics of a typical HP Filter.(Taken from [35]).

2.0 Doppler Spectra from Jet and Propeller Aircraft [36,28]

The motion of an aircraft target's propellers, in addition to causing some amplitude modulation of the airframe doppler signal, produces doppler signals of its own. Since the return from a propeller is periodic at the blade rotational frequency (speed at which the blade was rotating), and the blade tip velocity approaches Mach 1.0, the spectrum of the propeller doppler signal at most aspect angles is quite complex and is separate from that of the airframe doppler signal.

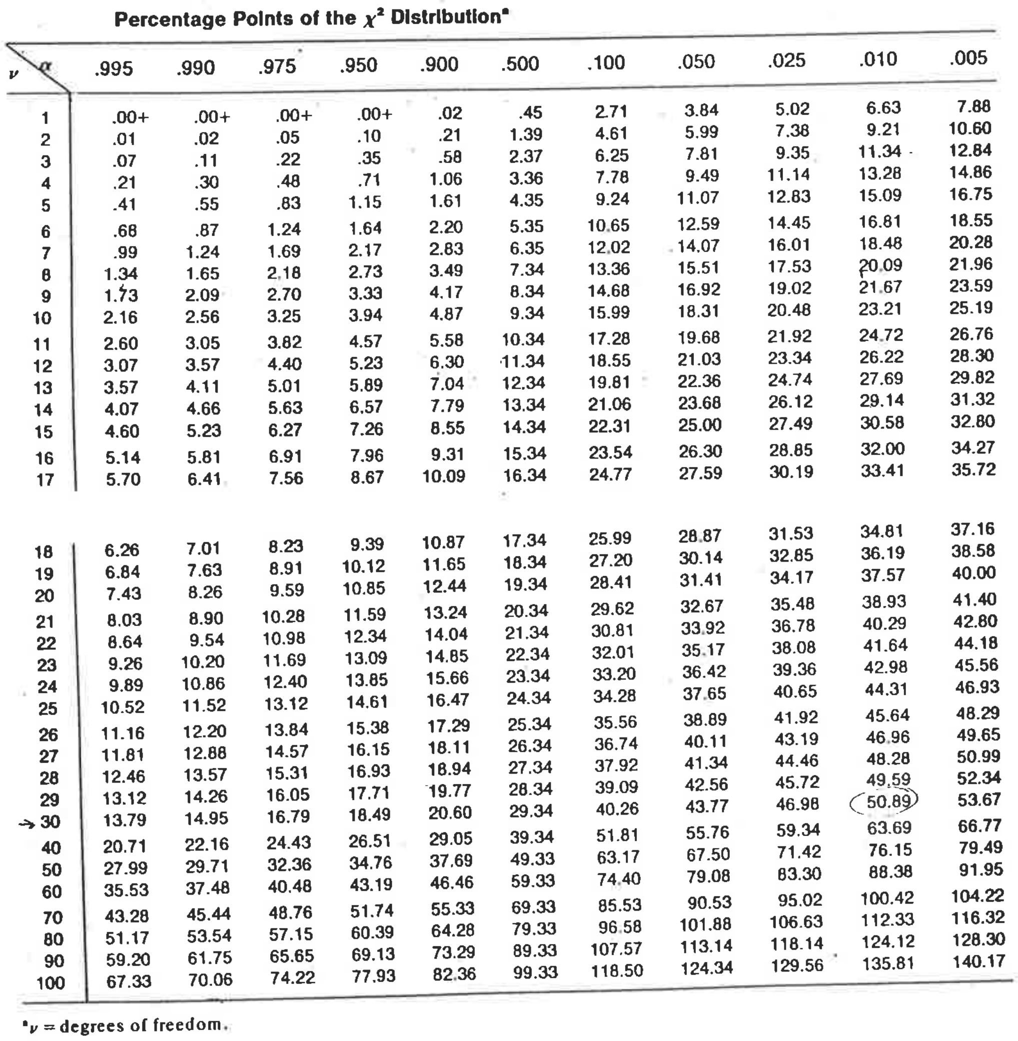
In the case of a jet aircraft, modulation is produced by the compressor or turbine blades of the engine. Even though the engines are usually totally enclosed, except for intake and exhaust ducts some times 16 feet in length, there is sufficient propagation down the ducts at microwave frequencies to allow ample amounts of rf energy to be modulated by the blades. Since the compressors and turbines contain relatively large numbers of blades rotating at high angular velocities, the modulation frequencies will be much higher than those of a propeller-driven aircraft (refer to Fig 30 & 32 experimental spectra results of jet and propeller driven aircraft). The modulation sidebands produced by the blades are easily distinguished, and, depending upon aspect angle and transmitter frequency, can be quite strong compared to the aircraft doppler return.

A typical doppler frequency spectrum of a propeller-driven aircraft, is shown in Figure 32. The most prevalent spectral line, of course, is the doppler return from the airframe, denoting its radial velocity. Some amplitude modulation sidebands at the propeller blade frequency appear around the airframe. This amplitude modulation can be caused by the propeller blade chopping a portion of the reflected radar energy from the airframe, thereby periodically modulating the received echo.

Also in the propeller driven aircraft case another group of spectral lines can be found at a lower frequency than that of the airframe line (refer to Figure 32). These are the result of reflected energy from the rotating propeller blades themselves, thus creating doppler frequencies proportional to the vector sum of the radial components of the airframe velocity and the propeller tangential velocity at the radius of the reflected surface. Since a propeller has a varying blade angle along its length, the position of the reflecting area on the blade depends upon the viewing angle and the blade angular position. Thus, the propeller doppler return is modulated at the blade frequency, and the centre frequency of its spectrum is dependent upon aspect angle.

For approaching targets with small aspect angles the major portion the propeller doppler spectrum is confined mostly to the region around the airframe (refer to Fig. 32) line, and at larger angles it has a much wider spread into the region lower in frequency than the airframe line [36].

Appendix G (Chi-Squared Distribution Tables, taken from [8])



Percentage Points of the χ^2 Distribution*

ν \ α	.995	.990	.975	.950	.900	.500	.100	.050	.025	.010	.005
1	.00+	.00+	.00+	.00+	.02	.45	2.71	3.84	5.02	6.63	7.88
2	.01	.02	.05	.10	.21	1.39	4.61	5.99	7.38	9.21	10.60
3	.07	.11	.22	.35	.58	2.37	6.25	7.81	9.35	11.34	12.84
4	.21	.30	.48	.71	1.06	3.36	7.78	9.49	11.14	13.28	14.86
5	.41	.55	.83	1.15	1.61	4.35	9.24	11.07	12.83	15.09	16.75
6	.68	.87	1.24	1.64	2.20	5.35	10.65	12.59	14.45	16.81	18.55
7	.99	1.24	1.69	2.17	2.83	6.35	12.02	14.07	16.01	18.48	20.28
8	1.34	1.65	2.18	2.73	3.49	7.34	13.36	15.51	17.53	20.09	21.96
9	1.73	2.09	2.70	3.33	4.17	8.34	14.68	16.92	19.02	21.67	23.59
10	2.16	2.56	3.25	3.94	4.87	9.34	15.99	18.31	20.48	23.21	25.19
11	2.60	3.05	3.82	4.57	5.58	10.34	17.28	19.68	21.92	24.72	26.76
12	3.07	3.57	4.40	5.23	6.30	11.34	18.55	21.03	23.34	26.22	28.30
13	3.57	4.11	5.01	5.89	7.04	12.34	19.81	22.36	24.74	27.69	29.82
14	4.07	4.66	5.63	6.57	7.79	13.34	21.06	23.68	26.12	29.14	31.32
15	4.60	5.23	6.27	7.26	8.55	14.34	22.31	25.00	27.49	30.58	32.80
16	5.14	5.81	6.91	7.96	9.31	15.34	23.54	26.30	28.85	32.00	34.27
17	5.70	6.41	7.56	8.67	10.09	16.34	24.77	27.59	30.19	33.41	35.72
18	6.26	7.01	8.23	9.39	10.87	17.34	25.99	28.87	31.53	34.81	37.16
19	6.84	7.63	8.91	10.12	11.65	18.34	27.20	30.14	32.85	36.19	38.58
20	7.43	8.26	9.59	10.85	12.44	19.34	28.41	31.41	34.17	37.57	40.00
21	8.03	8.90	10.28	11.59	13.24	20.34	29.62	32.67	35.48	38.93	41.40
22	8.64	9.54	10.98	12.34	14.04	21.34	30.81	33.92	36.78	40.29	42.80
23	9.26	10.20	11.69	13.09	14.85	22.34	32.01	35.17	38.08	41.64	44.18
24	9.89	10.86	12.40	13.85	15.66	23.34	33.20	36.42	39.36	42.98	45.56
25	10.52	11.52	13.12	14.61	16.47	24.34	34.28	37.65	40.65	44.31	46.93
26	11.16	12.20	13.84	15.38	17.29	25.34	35.56	38.89	41.92	45.64	48.29
27	11.81	12.88	14.57	16.15	18.11	26.34	36.74	40.11	43.19	46.96	49.65
28	12.46	13.57	15.31	16.93	18.94	27.34	37.92	41.34	44.46	48.28	50.99
29	13.12	14.26	16.05	17.71	19.77	28.34	39.09	42.56	45.72	49.59	52.34
→ 30	13.79	14.95	16.79	18.49	20.60	29.34	40.26	43.77	46.98	50.89	53.67
40	20.71	22.16	24.43	26.51	29.05	39.34	51.81	55.76	59.34	63.69	66.77
50	27.99	29.71	32.36	34.76	37.69	49.33	63.17	67.50	71.42	76.15	79.49
60	35.53	37.48	40.48	43.19	46.46	59.33	74.40	79.08	83.30	88.38	91.95
70	43.28	45.44	48.76	51.74	55.33	69.33	85.53	90.53	95.02	100.42	104.22
80	51.17	53.54	57.15	60.39	64.28	79.33	96.58	101.88	106.63	112.33	116.32
90	59.20	61.75	65.65	69.13	73.29	89.33	107.57	113.14	118.14	124.12	128.30
100	67.33	70.06	74.22	77.93	82.36	99.33	118.50	124.34	129.56	135.81	140.17

*ν = degrees of freedom.

Appendix H (CW Radar, Test & Training Data for NN1)

FREQ (KHz)	TRIAL 1&2 CW RADAR TARGET DATA (AT ASPECT ANGLE INDICATED) OVER THE 11 FREQUENCIES USED TO TRAIN NN1 (PRE-PROCESSED USING LINEAR PRED. FILTER)						
	Jet A 20 Deg.	**Jet B 55 Deg.**	**Jet C -60 Deg**	**Jet D -58 Deg**	**Prop. A 0 Deg.**	**Prop. B -20 Deg**	**Pro. C 7.7 Deg**
2	0.1231	0.7039	0.779	0.170	1.0348	0.8169	0.844
3	0.2502	0.6528	0.8526	0.2173	1.0332	0.8233	0.65
4	0.3133	0.6925	0.7319	0.2576	0.9736	0.7425	0.411
5	0.2088	0.7406	0.6534	0.18	1.0102	0.6784	0.248
6	0.084	0.6083	0.766	0.168	1.1064	0.78	0.159
7	0.029	0.4618	0.904	0.346	0.6965	0.67	0.1156
8	0.0685	0.4042	0.4991	0.5137	0.43	0.35	0.0829
9	0.2258	0.4137	0.282	0.1557	0.3129	0.197	0.0451
10	0.3696	0.3993	0.202	0.0112	0.3015	0.163	0.0152
11	0.1736	0.3014	0.2019	0.0226	0.3327	0.1979	0.0078
12	0.0321	0.203	0.2025	0.1248	0.2747	0.177	0.0204

Table 23 Trial 1&2 Continuous Wave Radar target data (consisting of 4 jets & 3 propeller driven light aircraft at various aspect angles) used to train neural network NN1, the data has been pre-processed using a Linear Prediction filter (smoother); as shown in the matlab program in Appendix A section 4.

Freq. (KHz)	TRIAL 3 CW RADAR TARGET DATA OVER THE 11 FREQUENCIES, USED TO TEST NN1 (ASPECT ANGLE SHOWN) (Data pre-processed using Linear Prediction Filter)						
	Target 1 Jet (-31 Deg.)	**Target 2 Jet (-32 Deg.)**	**Target 3 Propeller Aircraft (-16 Deg.)**	**Target 4 Propeller Aircraft (43 Deg.)**	**Target 5 Propeller Aircraft (35 Deg.)**	**Target 6 Propeller Aircraft (-33 Deg)**	**Target 7 Propeller Aircraft (-31 Deg)**
2	0.5728	0.7246	0.7233	0.7549	0.6312	0.5449	0.7031
3	0.3295	0.540	0.687	0.699	0.6071	0.4488	0.5003
4	0.2271	0.316	0.7117	0.615	0.5496	0.369	0.327
5	0.2467	0.206	0.504	0.7305	0.346	0.307	0.2519
6	0.3975	0.22	0.277	0.767	0.185	0.218	0.2612
7	0.6496	0.345	0.1697	0.397	0.125	0.1388	0.244
8	0.6754	0.4612	0.142	0.1957	0.1313	0.1038	0.1312
9	0.6713	0.4203	0.1221	0.1164	0.1089	0.101	0.043
10	0.5369	0.3629	0.0832	0.1207	0.0422	0.0949	0.0225
11	0.2509	0.2734	0.0856	0.1759	0.0259	0.0728	0.0422
12	0.0955	0.1719	0.168	0.213	0.102	0.051	0.0616

Table 24 Trial 3 Continuous Wave Radar target data (consisting of 2 jets & 5 propeller aircraft at aspect angles indicated) used to test neural network NN1, the data has been pre-processed using a Linear Prediction Filter (smoother); refer to Appendix A for matlab program.

Appendix I (Surveillance Radar Training Data for NN2)

TRAINING NN2 WITH POSITIONAL X&Y FLIGHT PATH DATA FOR PERTH & DARWIN FLIGHTPATHS				
CARTESIAN CO-ORDS INPUT IN (KM)		DESIRED OUTPUT		
X(KM)	Y(KM)	DARWIN	PERTH	UNKNOWN
0	4	1	1	0
5	7	1	0	0
10	10	1	0	0
15	16	1	0	0
20	23	1	0	0
25	30	1	0	0
30	39	1	0	0
5	4	0	1	0
10	4	0	1	0
15	5	0	1	0
20	7	0	1	0
30	9	0	1	0
35	10	0	1	0

Table 25 Training data for NN2 backpropagation neural network obtained from Adelaide Airport showing positional X & Y flight path data for Perth & Darwin jets.

TRAINING DATA FOR NN2 WITH POSITIONAL X&Y POSSIBLE UNKNOWN FLIGHT PATHS				
CARTESIAN CO-ORDS INPUT (KM)		DESIRED OUTPUT		
X	Y	DARWIN	PERTH	UNKNOWN
0	39	0	0	1
0	25	0	0	1
0	10	0	0	1
5	30	0	0	1
5	15	0	0	1
5	40	0	0	1
10	23	0	0	1
10	20	0	0	1
10	40	0	0	1
15	25	0	0	1
15	35	0	0	1
15	40	0	0	1
20	30	0	0	1
20	40	0	0	1
20	15	0	0	1
25	40	0	0	1
25	20	0	0	1
30	30	0	0	1
30	25	0	0	1
30	15	0	0	1
35	39	0	0	1

Table 26 Training NN2 with positional possible unknown flight-paths in cartesian co-ordinates X & Y.

Appendix J (FFT Spectrum & Linear Prediction Spectral Estimate Plots of Jet & Propeller Aircraft)

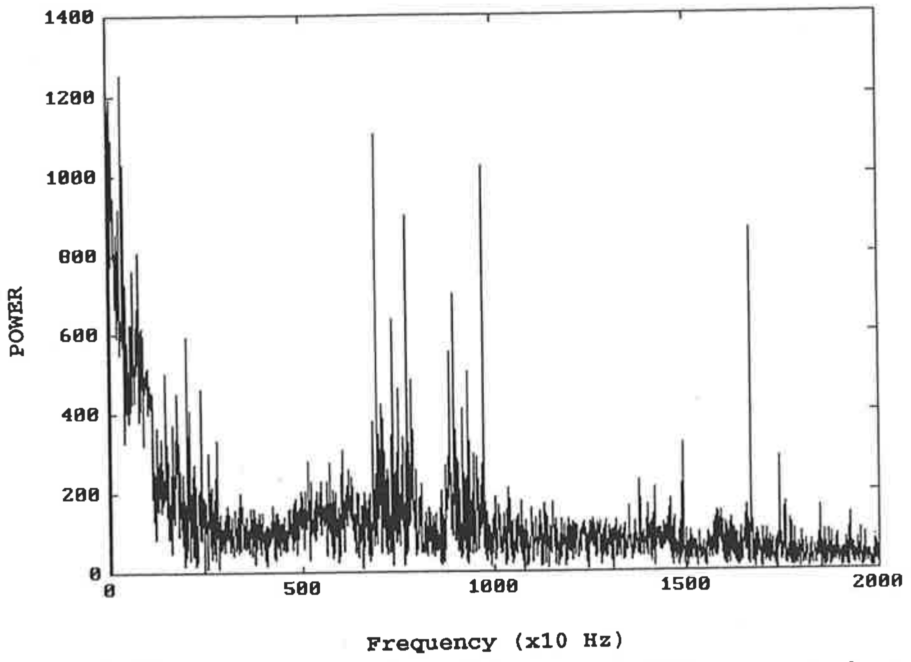


Figure 45 The FFT spectrum (magnitude squared, 4096 samples) of a commercial jet aircraft (Target 1) at an aspect angle of -31 degrees. (Sample rate = 50KHz)

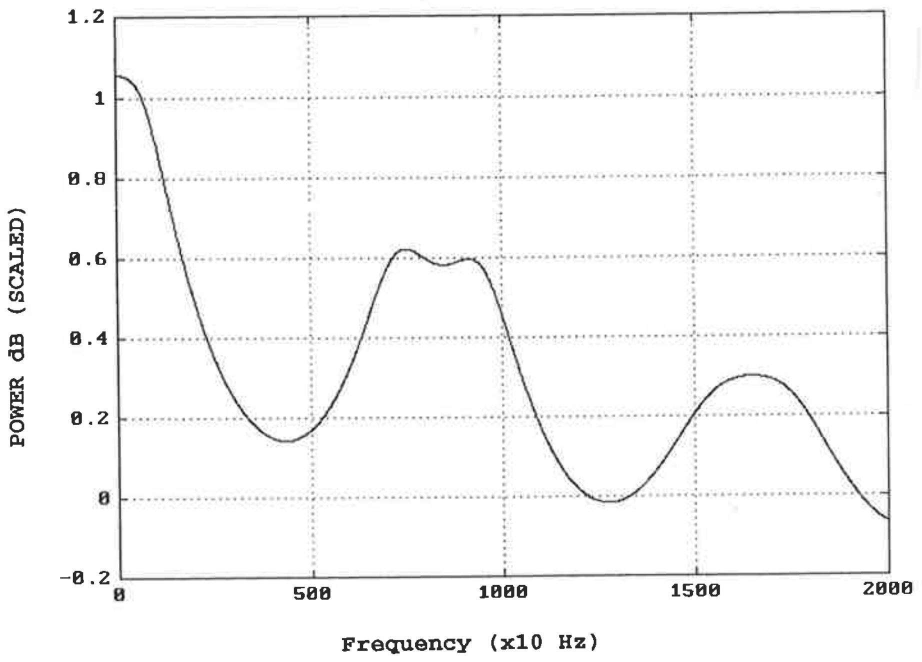


Figure 46 The linear prediction spectral estimate of a commercial jet (Target 1) shown above.

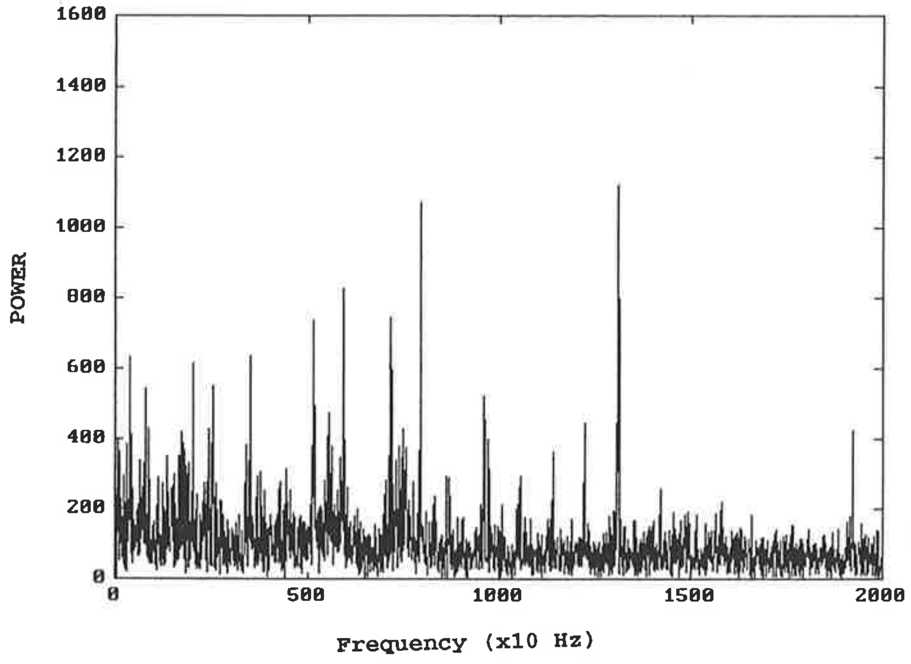


Figure 47 The FFT spectrum (magnitude squared, 4096 samples) of a commercial jet aircraft (Target 2) at an aspect angle of -45 degrees.

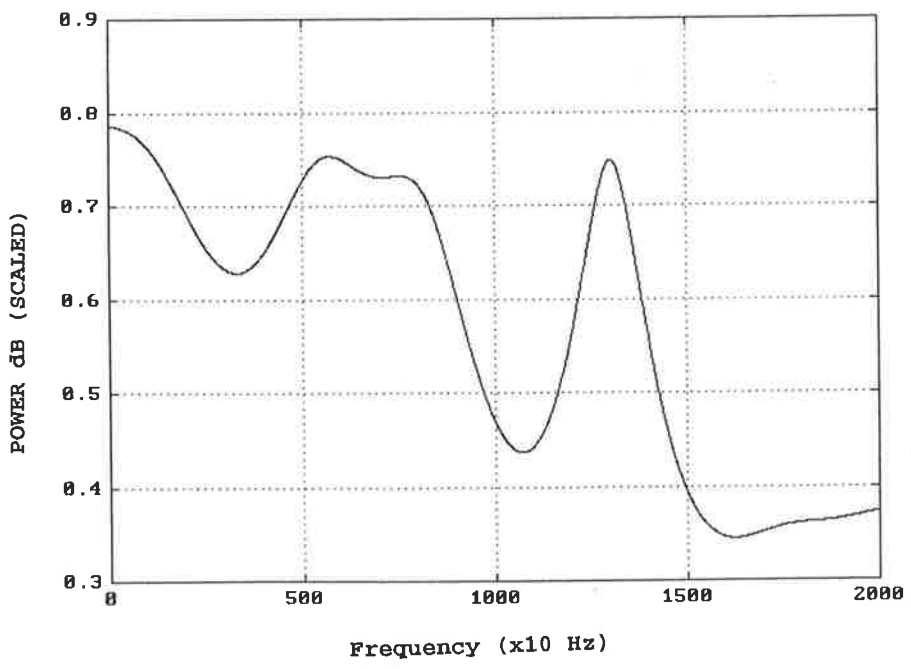


Figure 48 The linear prediction spectral estimate of a commercial jet shown above.

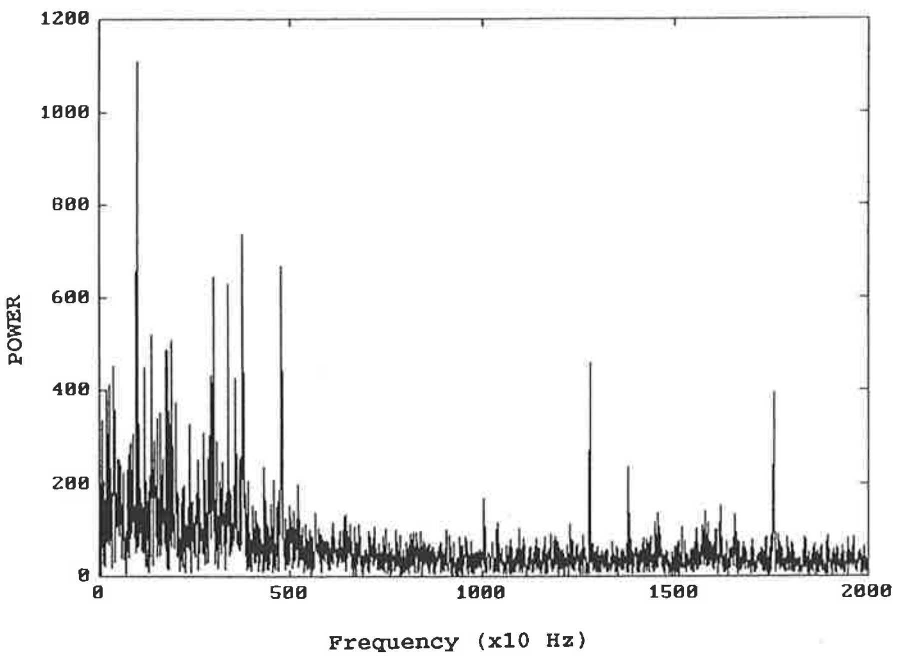


Figure 49 The FFT spectrum (magnitude squared) of a propeller driven aircraft (Target 3) at an aspect angle of -16 degrees.

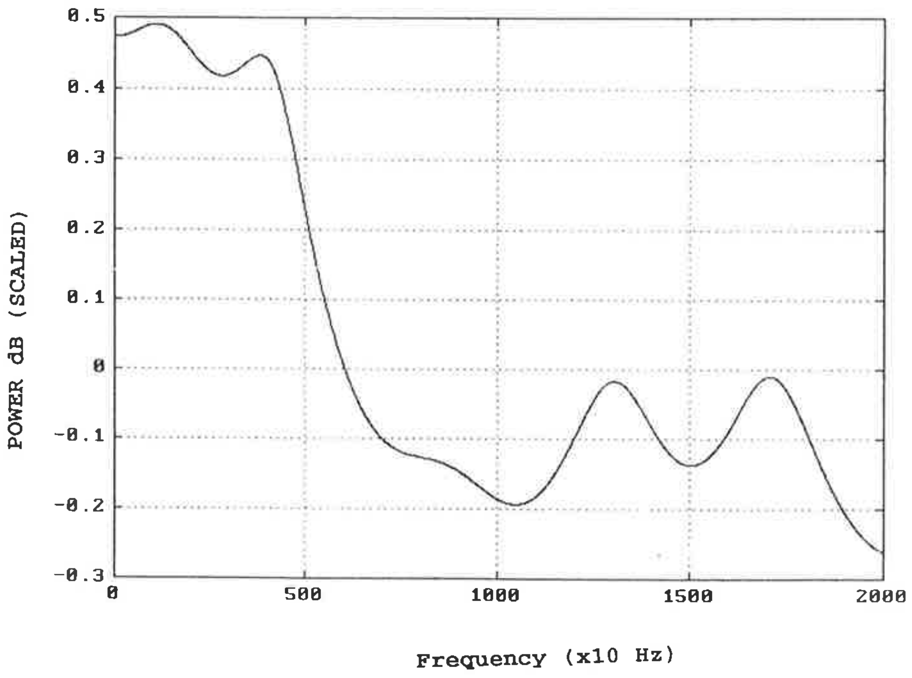


Figure 50 The linear prediction spectral estimate of a propeller driven aircraft shown above.

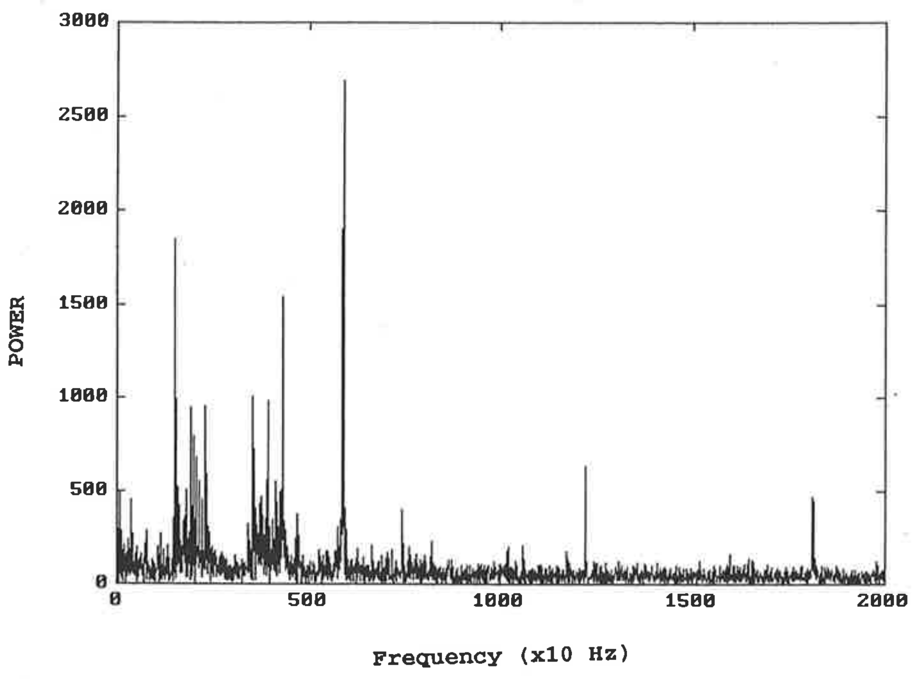


Figure 51 The FFT spectrum (magnitude squared) of a propeller driven aircraft (Target 4) at an aspect angle of 43 degrees.

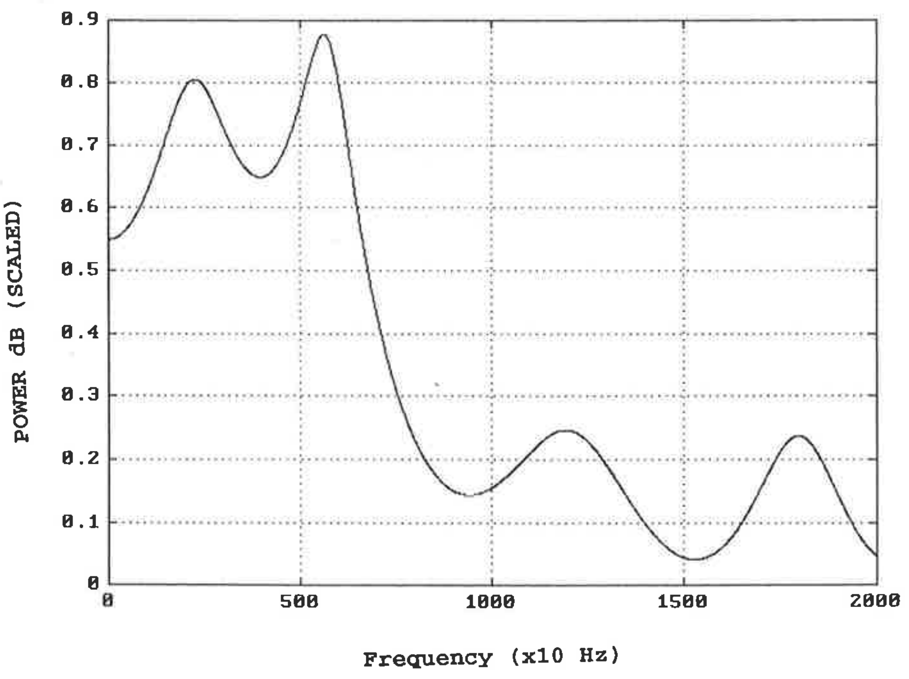


Figure 52 The linear prediction spectral estimate of the propeller driven aircraft shown above.

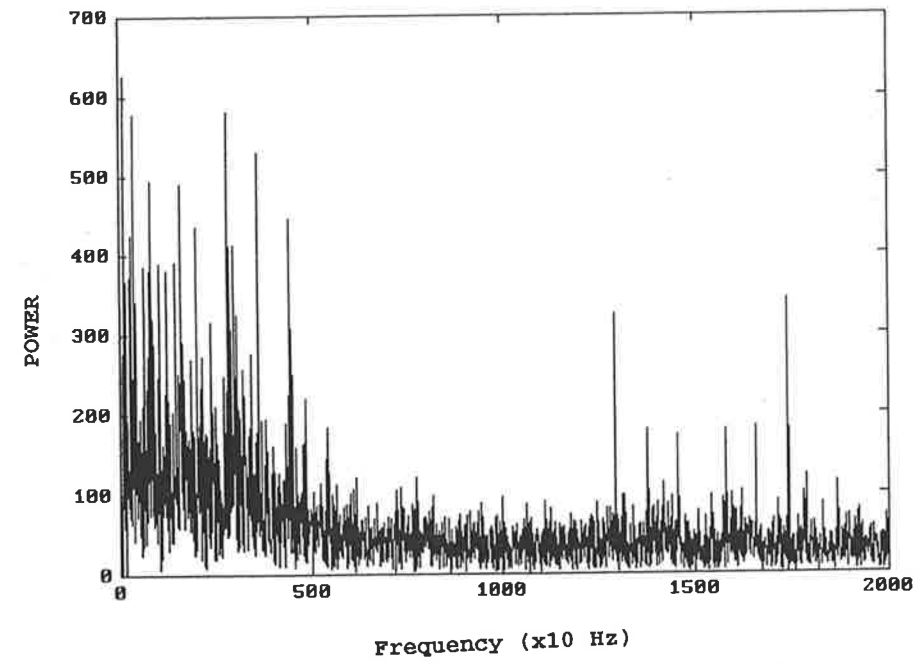


Figure 53 The FFT spectrum (magnitude squared) of a propeller driven aircraft (Target 5) at an aspect angle of -35 degrees.

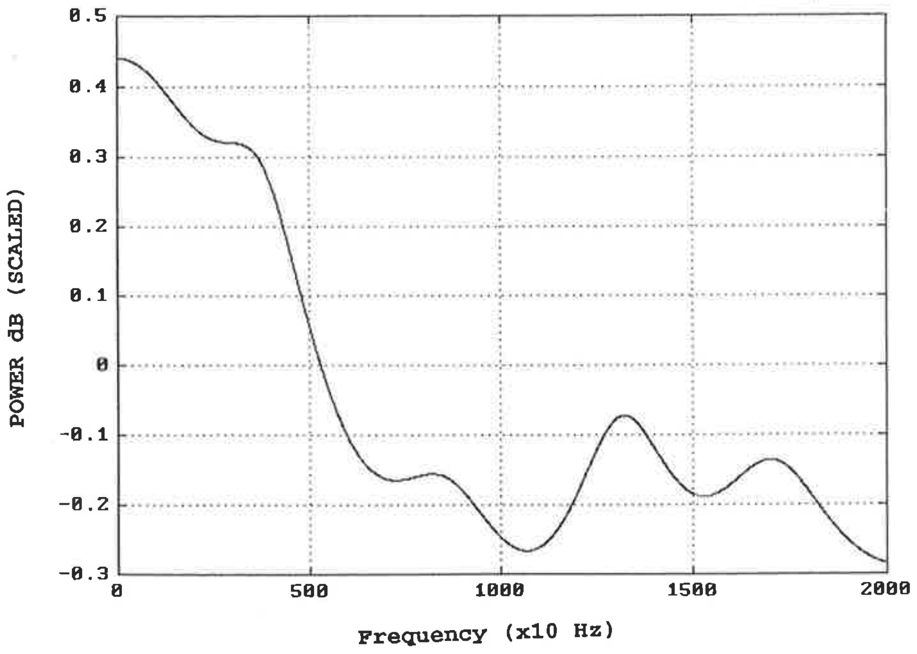


Figure 54 The linear prediction spectral estimate of a propeller driven aircraft shown above.

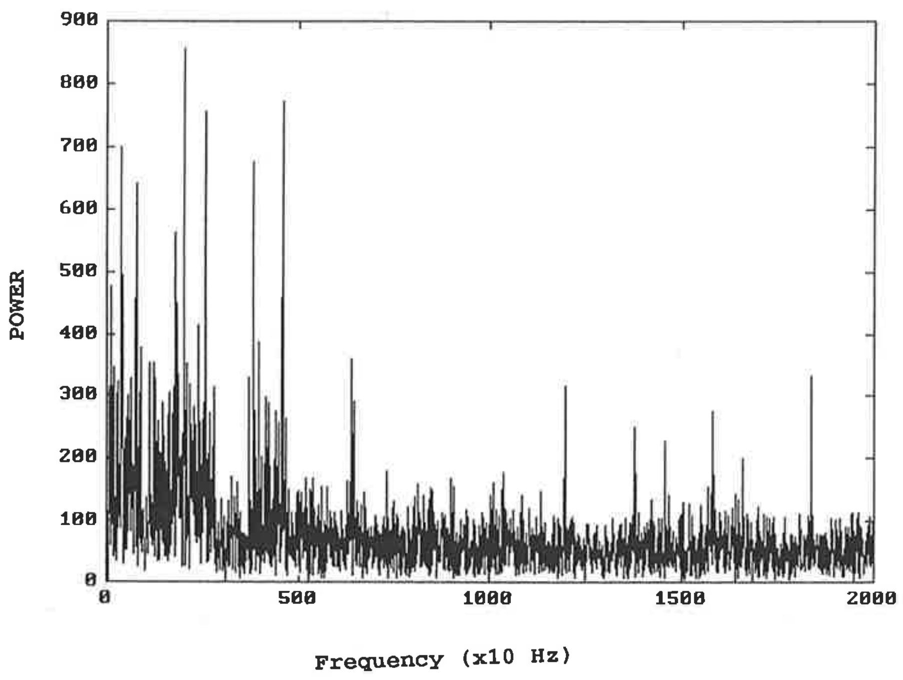


Figure 55 The FFT spectrum (magnitude squared) of a propeller driven aircraft (Target 6) at an aspect angle of -33 degrees.

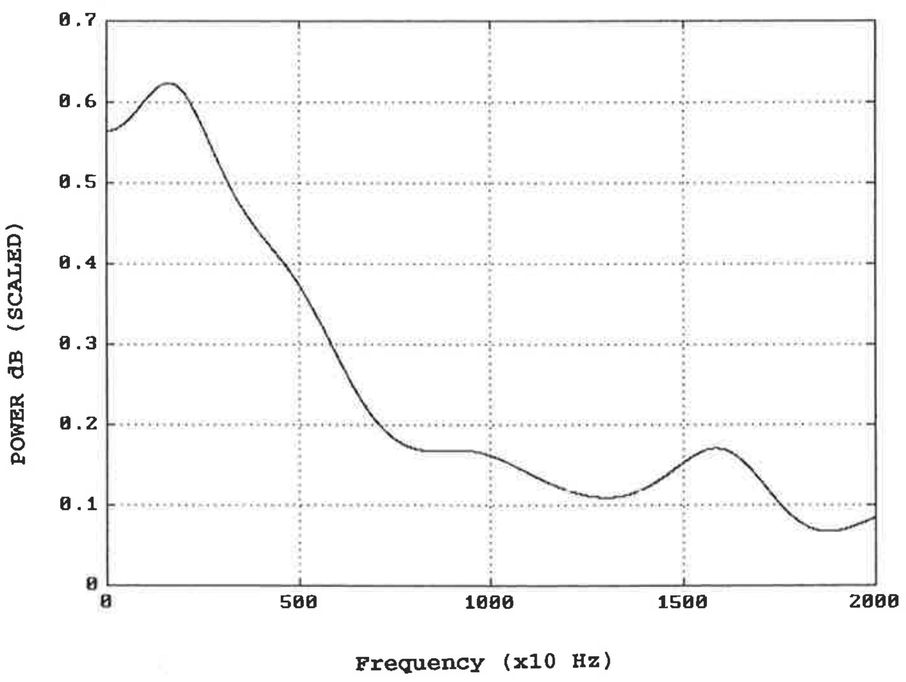


Figure 56 The linear prediction spectral estimate of a propeller driven aircraft shown above.

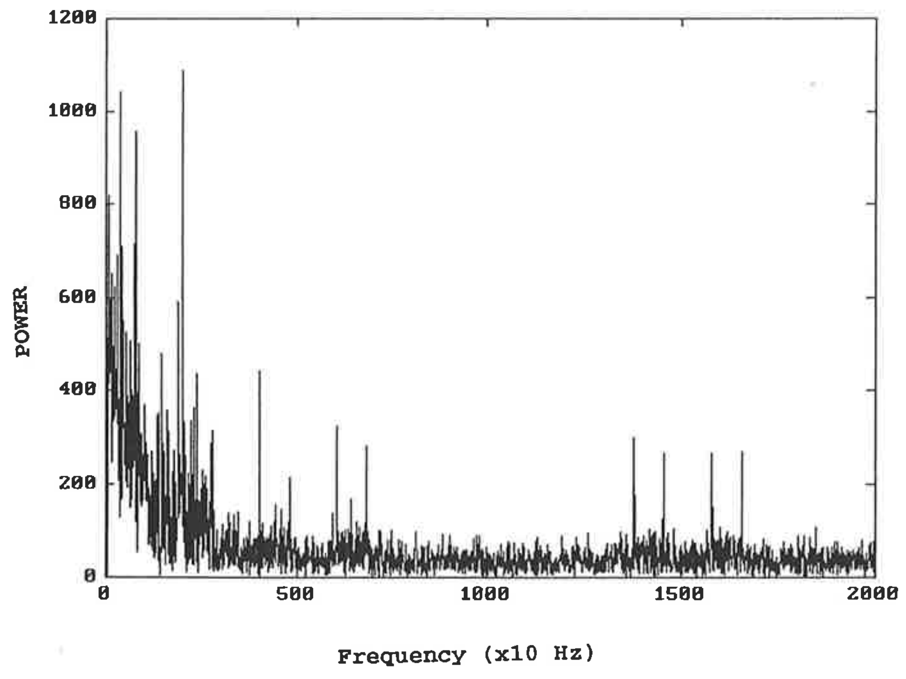


Figure 57 The FFT spectrum (magnitude squared) of a propeller driven aircraft (Target 7) at an aspect angle of -31 degrees).

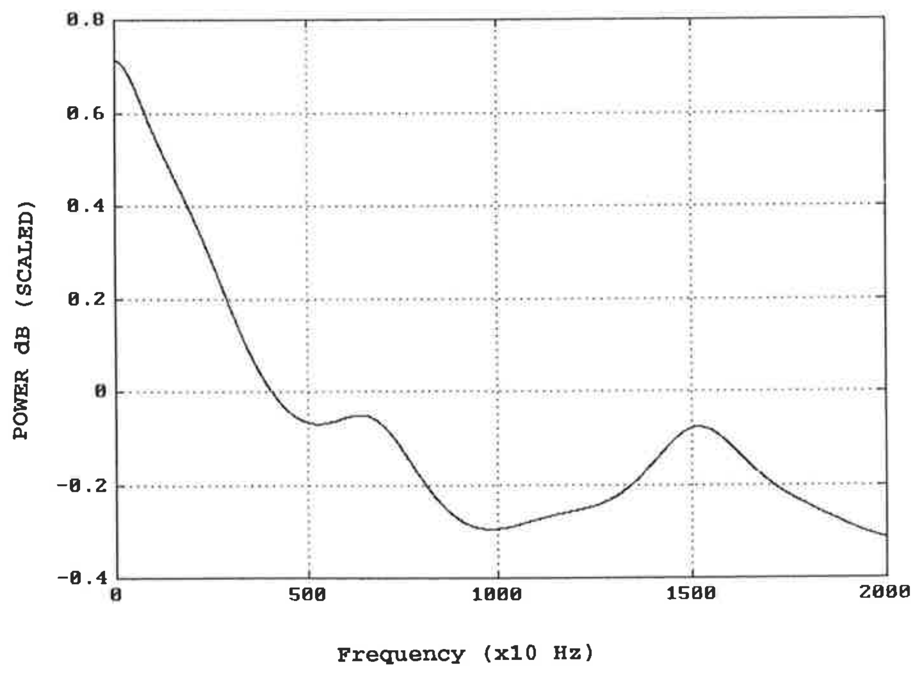


Figure 58 The linear prediction spectral estimate of the propeller driven aircraft shown above.

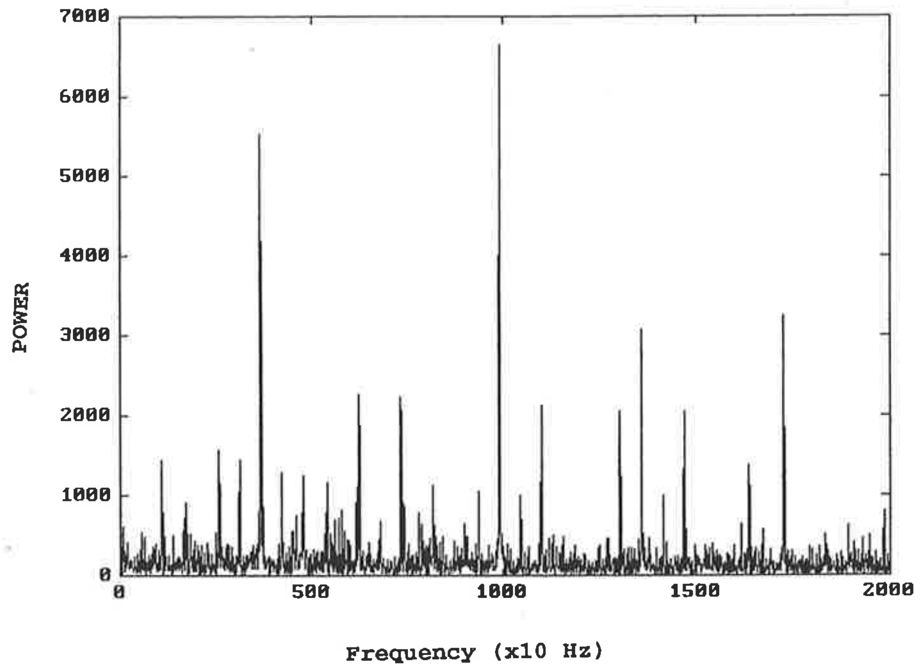


Figure 59 The FFT spectrum (magnitude squared) of a commercial jet aircraft at an aspect angle of 20 degree (training example 1 for NN1).

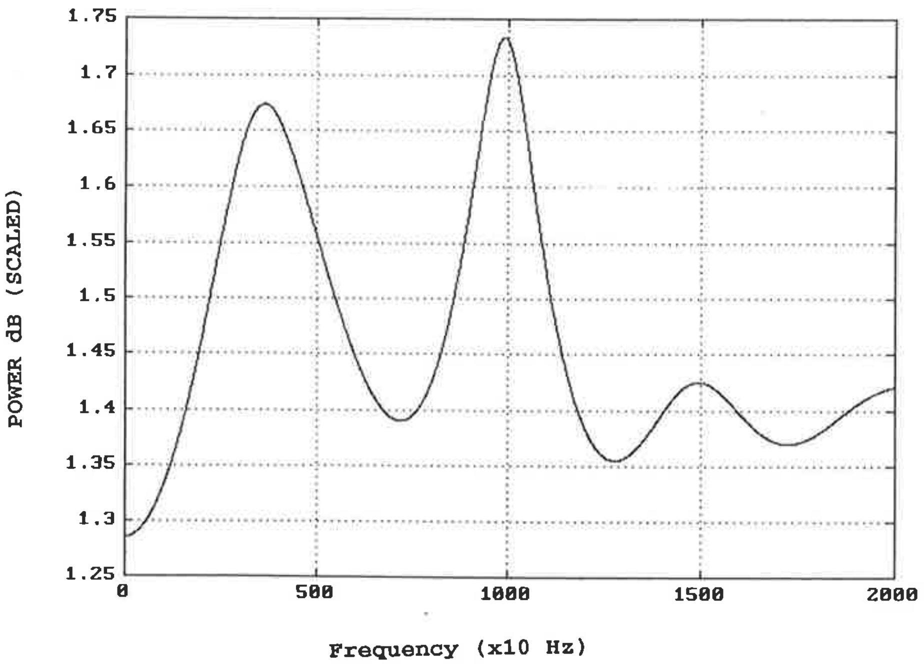


Figure 60 The linear prediction spectral estimate of a commercial jet shown above.

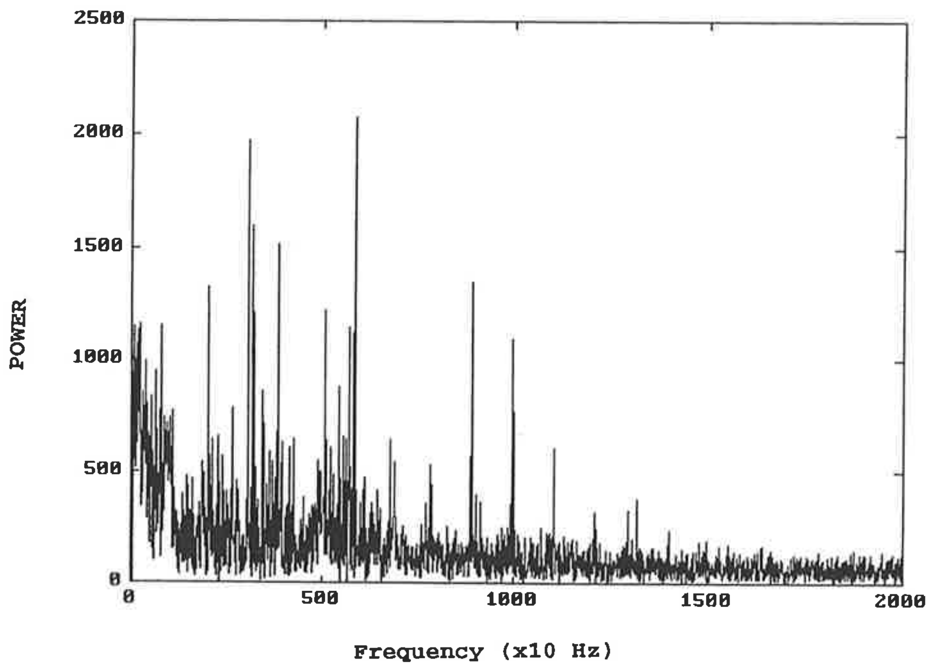


Figure 61 The FFT spectrum (magnitude squared) of a commercial jet aircraft at an aspect angle of 55 degrees (training example 2 for NN1).

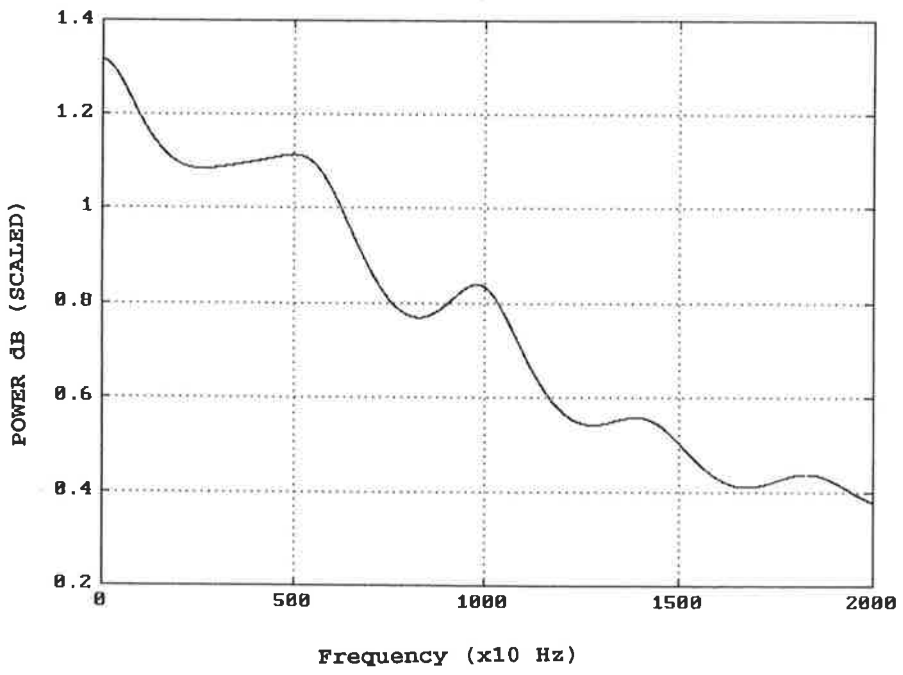


Figure 62 The linear prediction spectral estimate of a commercial jet shown above.

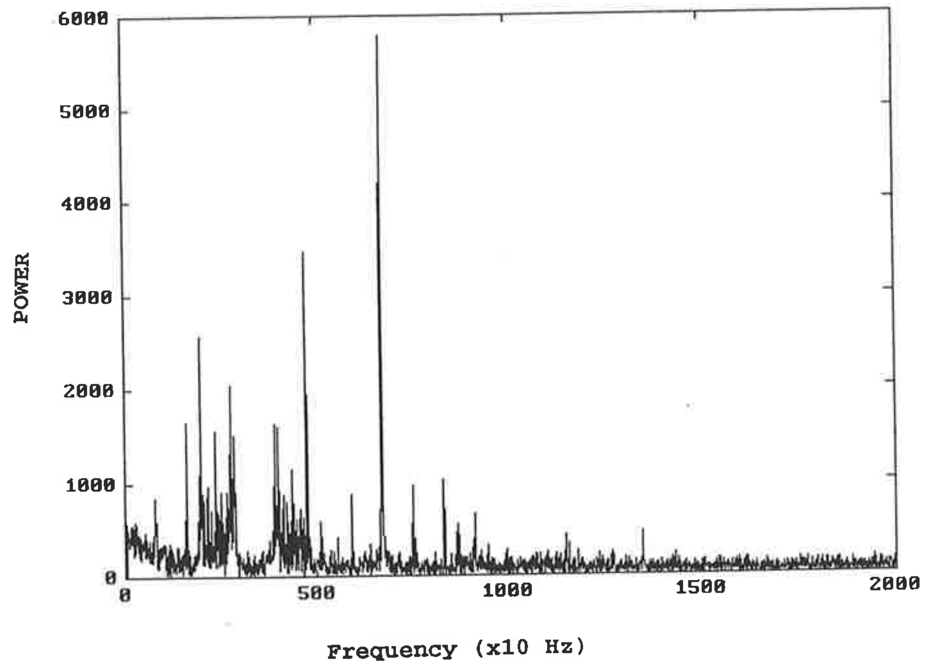


Figure 63 The FFT spectrum (magnitude squared) of a commercial jet aircraft at an aspect angle of 60 degrees (training example 3 for NN1).

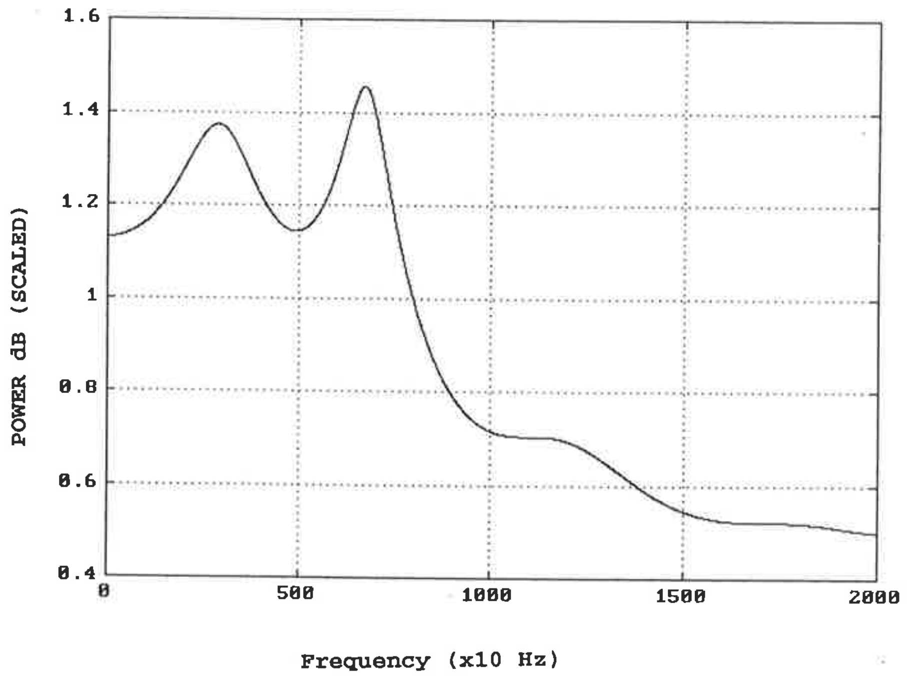


Figure 64 The linear prediction spectral estimate of a commercial jet shown above.

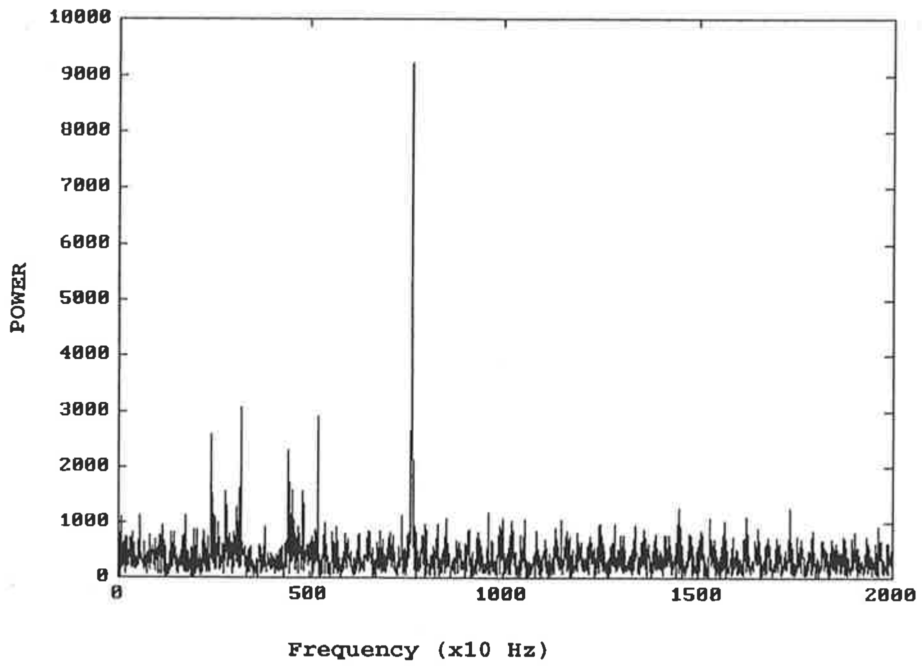


Figure 65 The FFT spectrum (magnitude squared) of a commercial jet aircraft at an aspect angle of -58 degrees (training example 4 for NN1).

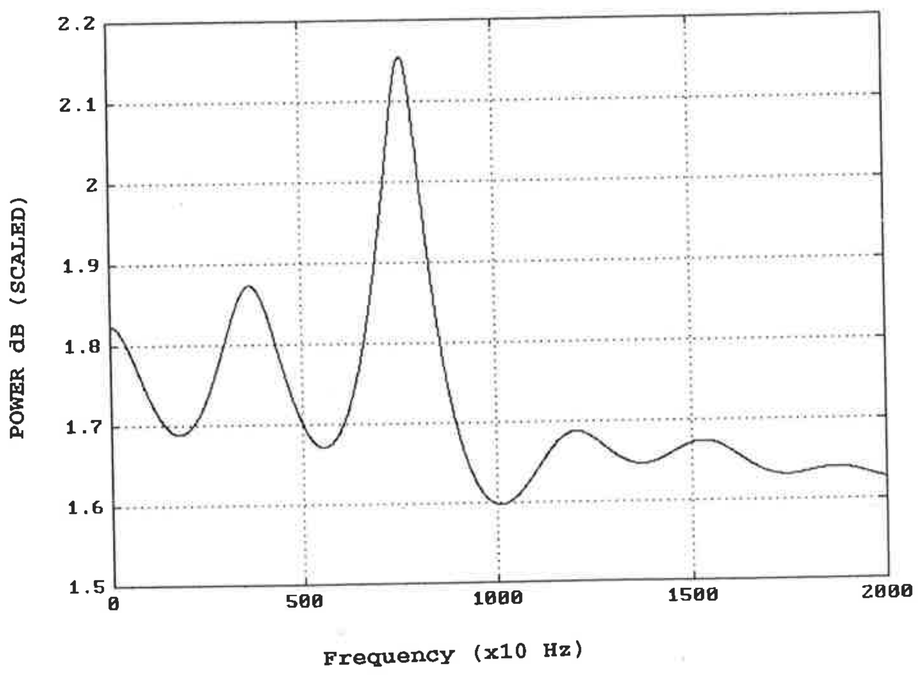


Figure 66 The linear prediction spectral estimate of a commercial jet sown above.

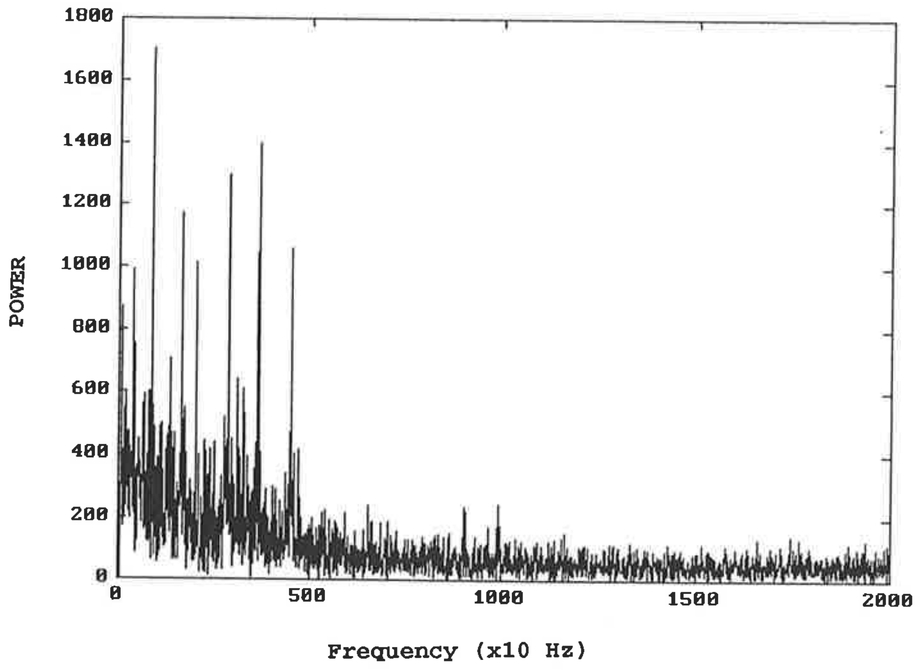


Figure 67 The FFT spectrum (magnitude squared) of a propeller driven aircraft at an aspect angle of 58 degrees (training example 5 for NN1).

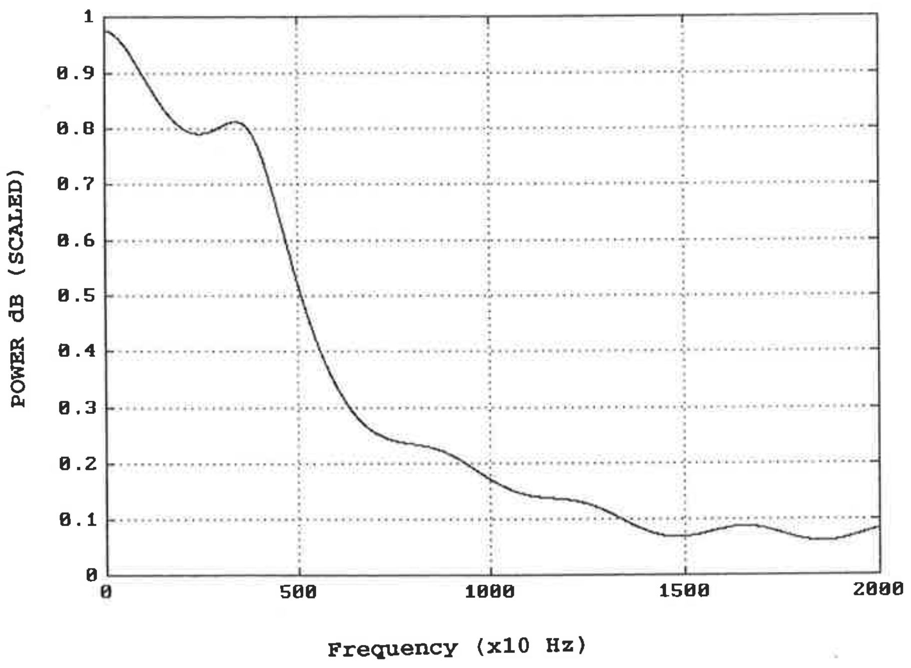


Figure 68 The linear prediction spectral estimate of a propeller driven aircraft sown above.

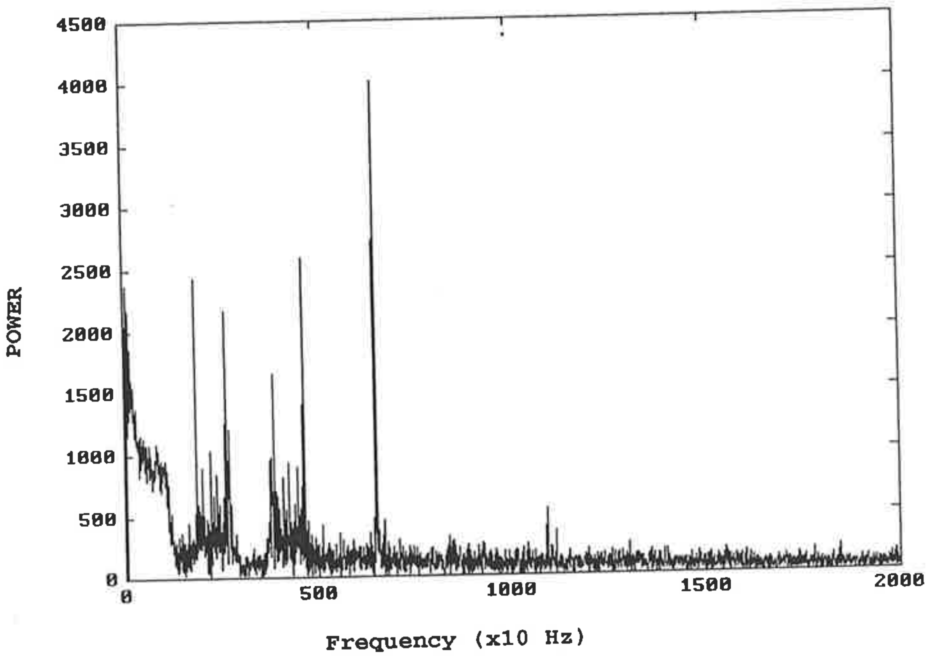


Figure 69 The FFT spectrum (magnitude squared) of a propeller driven aircraft at an aspect angle of -20 degrees (training example 6 for NN1).

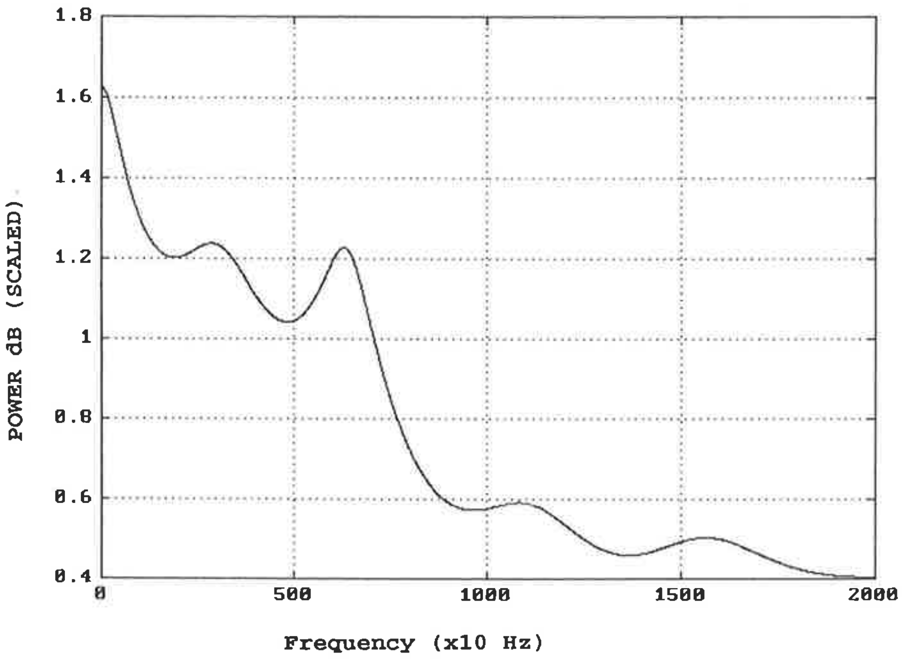


Figure 70 The linear prediction spectral estimate of a propeller driven target shown above.

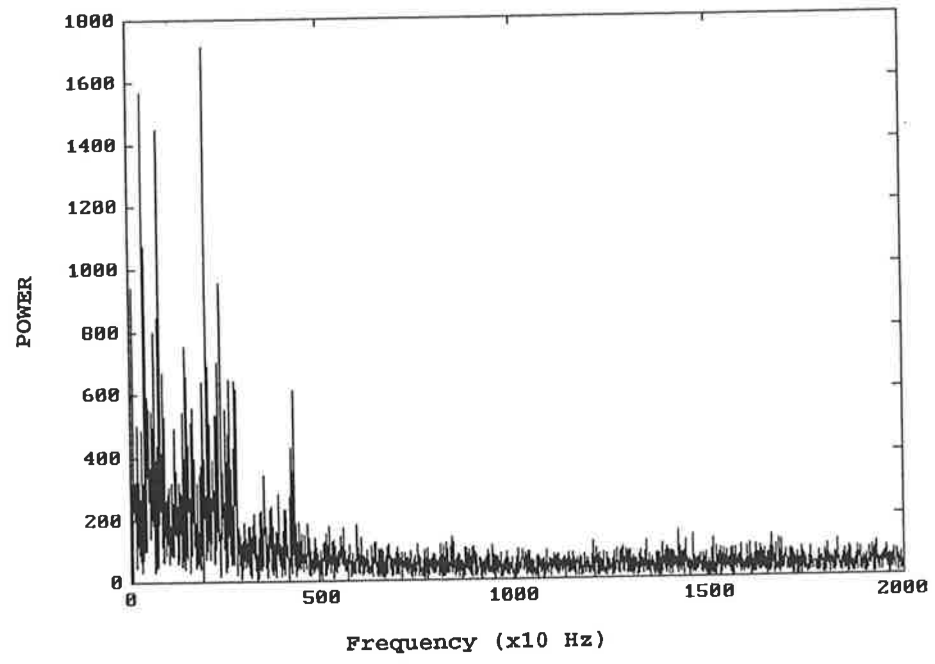


Figure 71 The FFT spectrum (magnitude squared) of a propeller driven target at an aspect angle of 7.7 degrees (training example 7 for NN1).

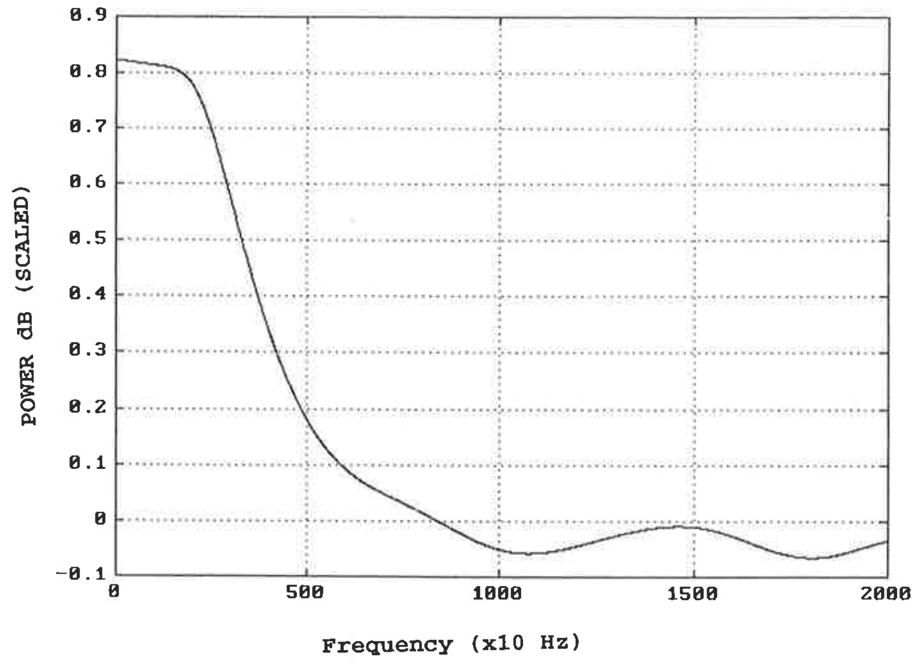
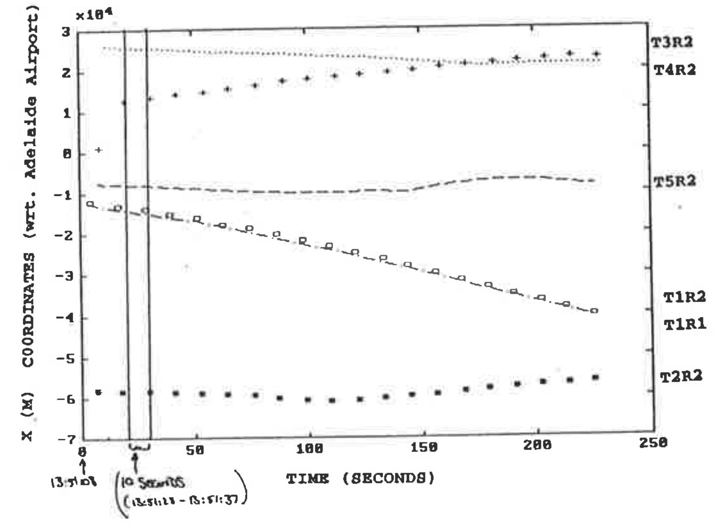
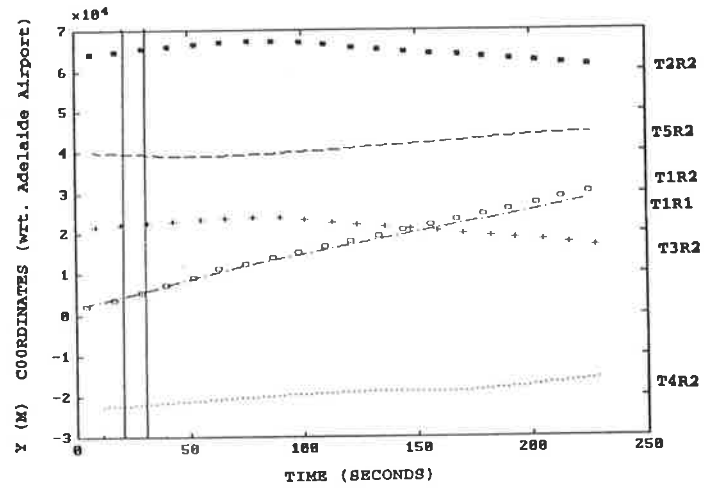


Figure 72 The linear prediction spectral estimate of a propeller driven target shown above.

Appendix K (Adelaide Airport & DSTO Radar X,Y & Z Plots)





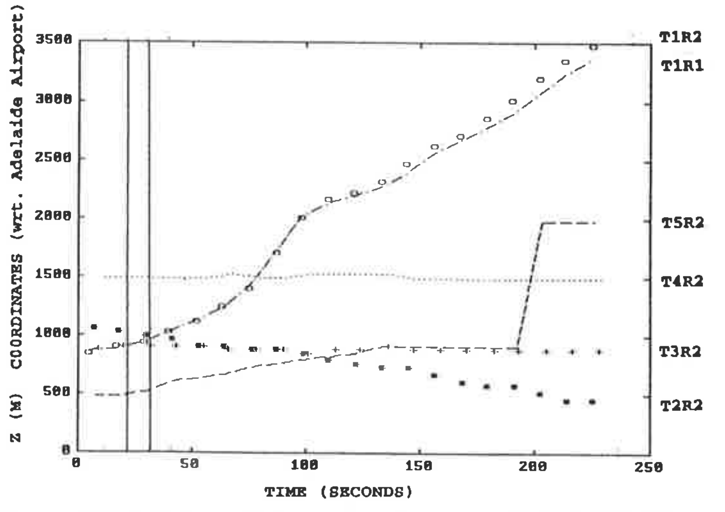


Fig. 73 Displays X,Y & Z (metres) versus time for targets. Radar1 (DSTO) reports 1 track (T1R1) and Radar2 (Adelaide Airport) reports 5 tracks (T1R1,T2R2,T3R2,T4R2 & T5R2). The 5 tracks above (X,Y,Z vs Time) are displayed in a common space/time co-ordinate graph. The vertical lines indicate the 10 second time period, we have performed the correlation tests (ie. 13:51:28-13:51:37).

Appendix L (Entropy & Uncertainty using D-S & Fuzzy Methods)

TARGET NO.	UNCERTAINTY & ENTROPY FROM COMBINED KNOWLEDGE SOURCES 1 & 2		UNCERTAINTY & ENTROPY FROM COMBINED KNOWLEDGE SOURCES 1, 2 & 3	
	D-S METHOD	FUZZY METHOD	D-S METHOD	FUZZY METHOD
1	4.2E-4	0.1976	1.6E-5	0.0899
2	4.2E-4	0.197	2.3E-5	0.089
3	3.3E-4	0.0798	6.6E-6	0.07
4	1.97E-4	0.1425	3.9E-6	0.139
5	1.99E-4	0.1445	5.9E-6	0.0905
6	2.14E-4	0.1561	6.7E-6	0.0899
7	3.2E-4	0.1665	8.99E-6	0.089
8	4.4E-4	0.1679	2.8E-4	0.0668

Table 27 Shows & compares the reduction of uncertainty/entropy using Dempster Shafer (D-S) & Fuzzy Reasoning methods when combining two and then a third knowledge source.

Appendix M (Verification Tests on NN1)

To verify that NN1 was classifying the data correctly the following tests were performed (as shown below) on processed trial data obtained from the CW Radar (ie. Jet or Propeller driven Aircraft). Random noise was added (using matlab software) to the CW radar (real) data stored in Array 0 as discussed in (section 4) Appendix A .(Note the maximum real magnitude value is 12).

The following output results are shown in Table 28 when noise (0.5, 1.0 and 2.0) is added to array0 and processed using the linear predictive filter, and then input to NN1. The neural network NN1 was trained on data shown in Appendix H, Table 23 and tested on data shown in Table 24 (output results with noise shown in Table 28).

Target N0.	Correct Classificat.	Noise Added to Array0			
		None	0.5	1.0	2.0
1	1.0	0.9998	0.9998	0.99969	0.999
2	1.0	0.9994	0.997	0.9901	0.728
3	0.0	0.0032	0.007	0.0047	0.07
4	0.0	0.0024	0.0085	0.0048	0.0059
5	0.0	0.0092	0.01675	0.026	0.23
6	0.0	0.05	0.036	0.045	0.06
7	0.0	0.1608	0.01769	0.035	0.036

Table 28 The output of NN1 when varying amounts of noise is added to the input data in Array0, and processed using LPC.

Test and training data were swapped and the results noted from the output of NN1.
NN1 was initially trained on the data shown in Table 23 and tested on data from Table 24 Appendix H, the output results from NN1 are shown in Table 29 (Fig. 74). Then NN1 was trained on the data shown in Table 24 and tested on data shown in Table 23 Appendix H, the output results from NN1 are shown in Table 30 (Fig. 75).

The overall conclusion is that input noise generated (max. 16% of input signal), didn't make significant changes to the output of NN1 as shown in Table 28. And NN1 classified the input data correctly (jet or propeller targets) even when testing and training data for the network were swapped, (refer to Bar Graphs).

Targets	Desired NN1 Output	Output from NN1 Trained on Data from table 21 and tested on data from table 22
1	1.0	0.9998
2	1.0	0.9994
3	0.0	0.0032
4	0.0	0.0024
5	0.0	0.0092
6	0.0	0.0500
7	0.0	0.1608

Table 29 The output data from neural network NN1 when trained on data shown in table 23 and tested on data shown in table 24 Appendix H.

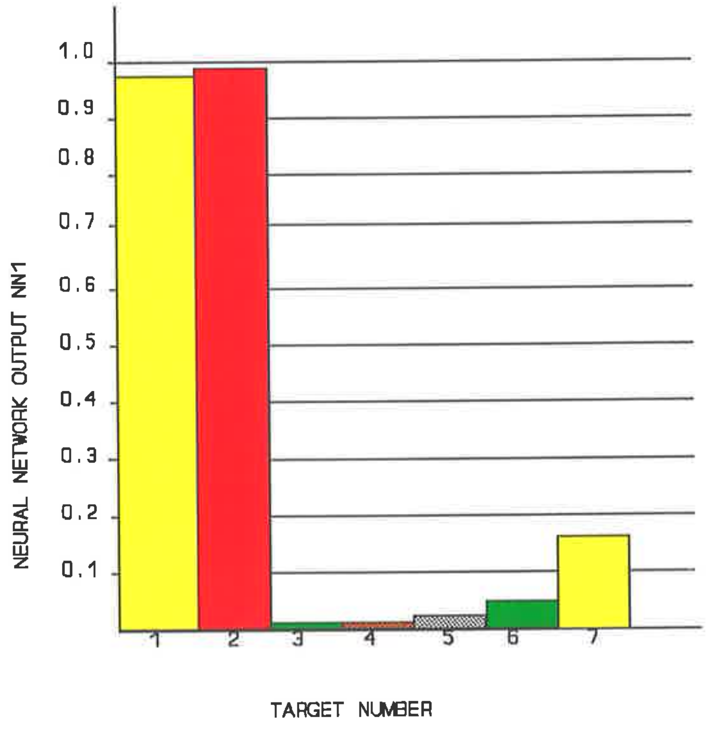


Figure 74 NN1 output when trained on processed data from table 23 and tested on data from table 24 Appendix H.

Target NO.	Desired Output	NN1 output when Trained on Table 22 and Tested on Table 21
1	1.0	0.9676
2	1.0	0.8000
3	1.0	0.7800
4	1.0	0.9960
5	0.0	0.00268
6	0.0	0.010678
7	0.0	0.0028

Table 30 The output of NN1 when trained with data from table 24 and tested with data from table 23 Appendix H (processed CW Radar data).

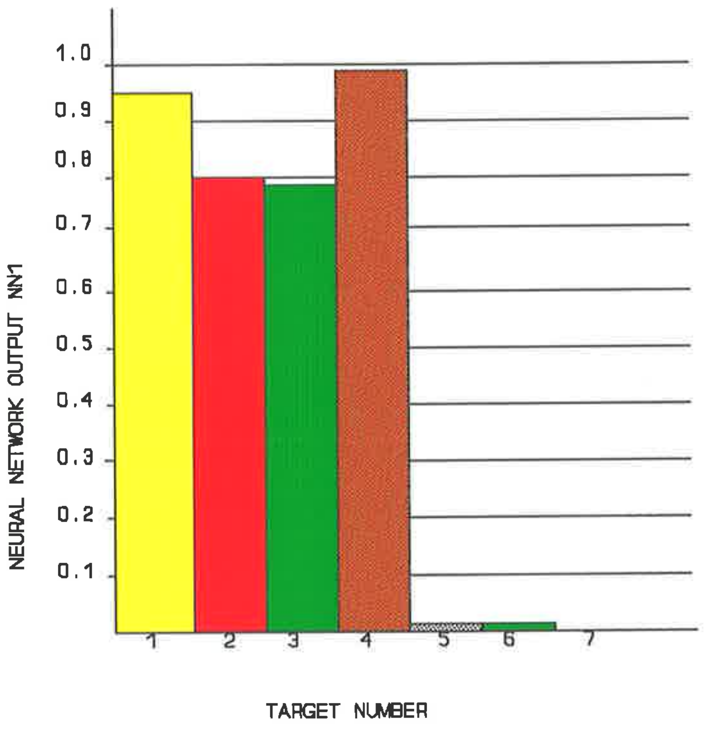


Figure 75 Bar graph showing the results from NN1 when trained with processed CW Radar data from table 24 and tested with data from table 23 Appendix H.

Appendix N (Data Fusion of multiple classifiers [32])

One of the problems in data fusion is the combination of "opinions" of multiple classifiers as to the identity of a target. Such a situation arises for my experimental case, for example, a number of knowledge sources (ie sensors, priori knowledge) outputs can reach a conclusion about the identification of a proposition (ie target type , Darwin jet, Perth jet, Unknown jet, Propeller driven aircraft). It is desirable to combine these knowledge sources to give a better assessment of the identity of the target.

The Bayesian approach to classification of an observation (ie. vector x) of the identity of a target for example is to find the class "*c*" which maximises *P(c/x)* . ie From Bayes rule we have:-

$$P(c/x)=P(x/c).\frac{P(c)}{P(x)}$$

In data fusion sensors (or knowledge sources) can be of different types ie. a Surveillance Radar gives us positional attributes of targets and a CW Radar gives us doppler modulation and flight timetables give us arrival flight times of possible targets. The approach used is to perform initial processing on partial information, and then attempt to fuse the resulting reduced data , ie. make separate classification based on information from each knowledge source and integrate these decisions. For example consider the general case where we can break the classification problem into separate parts as shown in Fig. 76.

Each of the sensors processors Si (refer to Fig. 76) could attempt either "hard" or "soft" classifications, producing decisions on the identity of a target type for example. The fusion centre "F" makes an overall classification on the basis of the information supplied to it.

Adaptive feedforward neural networks can give appropriate probabilistic outputs, and these can be used to estimate the class probabilities [32]. The architecture shown in Fig. 76 is similar to that used for multisensor data fusion (shown in Fig. 35) which consists of a set of independent sensor neural nets (NN1 and NN2) one for each sensor (CW radar and Surveillance radar) coupled to a fusion net NN3.

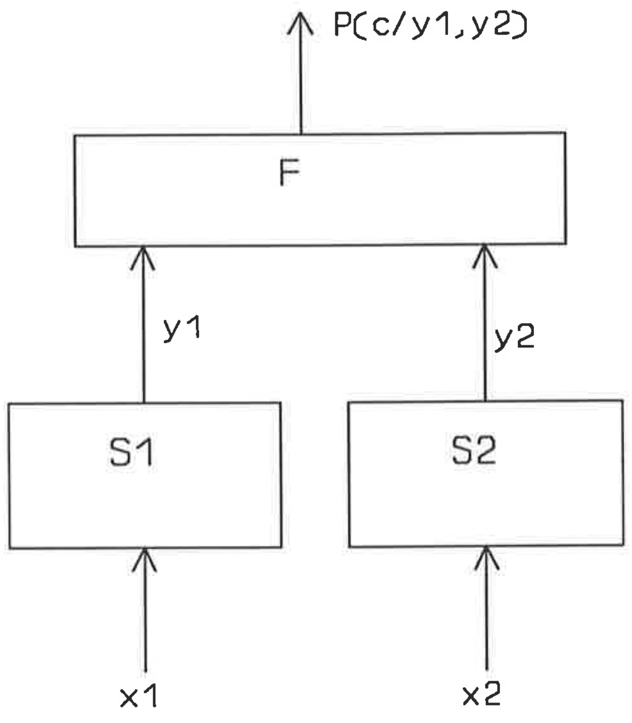


Figure 76 The integration of classifier modules. (Taken from [32]).

Appendix O (Polar to Cartesian Conversion of Radar Data)

The raw data received from the FPS-16 Radar is in the following format (stored in three separate files for range, azimuth and elevation):-
Slant Range (in 10's of metres), Azimuth (as a binary fraction of 360 degrees, where 8000 Hex = 180 Degrees with respect to true north) and similarly for Elevation (8000 Hex =180 Degrees) with a 45 degree offset due to the position of the radar. (Refer to Appendix D for "C" software used to process the data) .

The data from both radars has to be aligned in space (as shown below in steps 1 and 2) and time (local time was recorded every second from both radars) before any fusion can occur . The raw data from the FPS-16 Radar was converted to polar and then to cartesian X(m), Y(m), Z(m) co-ordinate system relative to Adelaide Airports position (origin X=0,Y=0,Z=0) (Refer to Appendix D for software written in "C" used to process the raw radar data to the format mentioned above).

The FPS-16 Radar Polar to Cartesian conversion sums for the program shown in Appendix D, are as follows :-

Step 1 Convert Azimuth and Elevation from Degrees to Radians (correct the angle from true north to grid north (subtract 1.4 Degrees)).

$$AZIMUTH,RADIANS\ ;\ AZRAD=(AZIMUTH-1.4)*\frac{\Pi}{180}$$

$$ELEVATION,RADIANS\ ;\ ELRAD=(ELEVATION)*\frac{\Pi}{180}$$

Step 2 Polar to Cartesian conversion (note, slant range is the distance from the radar to the target in the air, whereas ground range is the ground distance, ie. refer to Fig. 77).

(Z co-ordinate) Z=(SLANT RANGE)∗SIN(ELRAD)

GROUND RANGE=(SLANT RANGE)∗COS(ELRAD)

(X co-ordinate) X=(GROUND RANGE)∗SIN(AZRAD)

Y=(GROUND RANGE)∗COS(AZRAD)

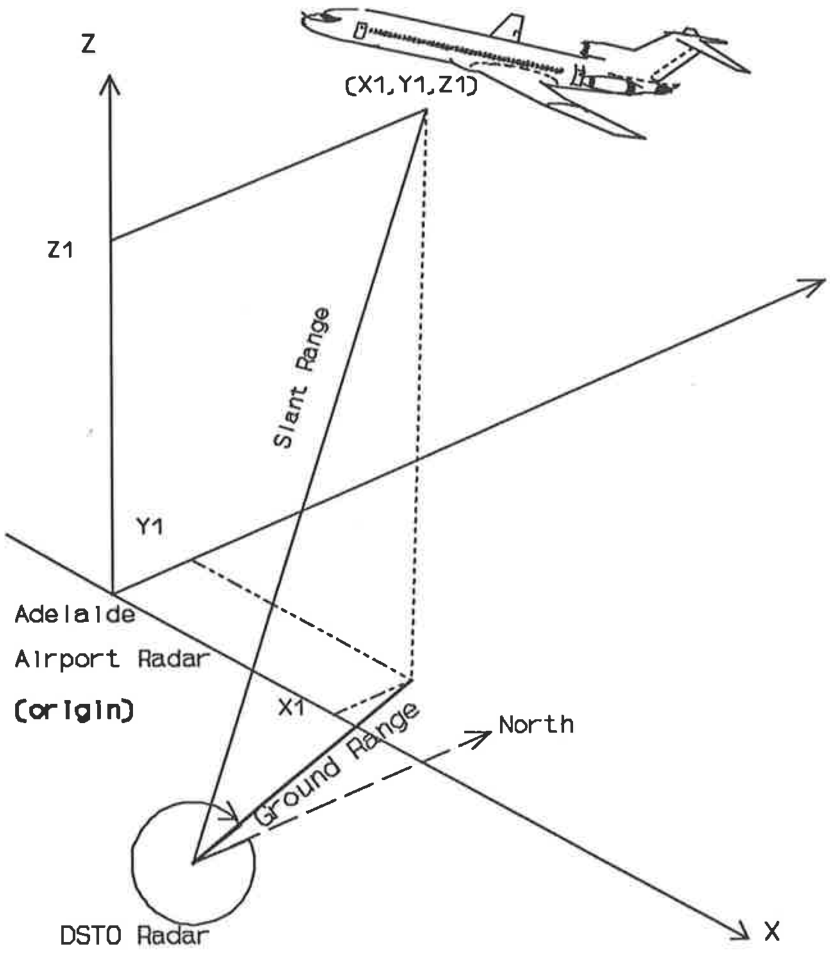


Figure 77 Converting polar coordinates from the radars to cartesian, with respect to Adelaide Airports Grid map position which is used as the origin (ie. x, y, z =0).

DSTO'S Radar grid map position (in Eastings and Northings) are as follows:-

East 282 088.78 metres
North 6 154 616.49 metres.

Adelaide Airport's Radar grid map position (in Eastings and Northings) are:-

East 274 626.1 metres

North 6 129 373.4 metres

Some of the measurement corrections that have to be noted and adjusted for are as follows: Taking into consideration the point scale factor (which is the ratio of an infinitesimal distance at a point on the grid to the corresponding distance on a spheroid) . For example, 1 metre on the ground is equal to 1.0002 metres on a grid map showing Easting and Northings measurements, the difference is negligible for small distances. Also Azimuth (with respect to true north) measurements taken from the radar have to be adjusted to grid north (which is a difference of 1.4 degrees) as shown in the software in Appendix D.

Since I am making Adelaide Airport the cartesian reference point, the new adjusted cartesian co-ordinates are as follows (in metres):-
Adelaide Airport Radar X = 0, Y = 0; DSTO'S FPS-16 Radar position (point scale factor adjusted, with respect to Adelaide Airport) X = 7,4614.17 metres, Y = 25,248.13 metres.

Bibliography

[1] Dr. Robert Popoli "MULTIPLE SENSOR TRACKING" Course Manual provided by Technology Training Corporation, Pty Ltd, Suite 101, 275 Alfred St., North Sydney, NSW.

[2] Edward Waltz, James Llinas, "MULTISENSOR DATA FUSION" Artech House, 1990.

[3] James Llinas "A SURVEY OF TECHNIQUES FOR CIS DATA FUSION" IEE Second International Conference on Command, Control, Communications and Management Information Systems,1987, Publication Number 275.

[4] Edward L. Waltz "COMPUTATIONAL CONSIDERATION FOR FUSION IN TARGET IDENTIFICATION SYSTEMS" 1981 IEEE.

[5] S S Blackman, "MULTIPLE TARGET TRACKING WITH RADAR APPLICATIONS" 1986 380-393.

[6] Bar-Shalom, Y.,"ON THE TRACK-TO-TRACK CORRELATION PROBLEM," IEEE Transactions on Automatic Control, AC-26, April 1981, pp.571-572.

[7] Samuel S. Blackman "MULTIPLE-TARGET TRACKING WITH RADAR APPLICATIONS",pp.363-364.

[8] William W. Hines, Douglas C. Montgomery, "PROBABILITY AND STATISTICS IN ENGINEERING AND MANAGEMENT SCIENCE " Second Edition, John Wiley & Sons, 1980.

[9] Carpenter, Gail A., Grossberg, Stephen, "THE ART OF ADAPTIVE PATTERN RECOGNITION BY A SELF ORGANISING NEURAL NETWORK". Computer, pp. 77-88, March, 1988.

[10] Grossberg, Stephen, "ADAPTIVE PATTERN CLASSIFICATION AND UNIVERSAL RECORDING" Biological Cybernetics, Vol. 23, 1976.

[11] Stork, David. G., "SELF-ORGANISATION, PATTERN RECOGNITION, AND ADAPTIVE RESONANCE NETWORKS", Journal of Neural Network Computing.

[12] Carpenter, Gail A. and Grossberg, Stephen, "ART 2: SELF-ORGANIZATION OF STABLE CATEGORY RECOGNITION CODES FOR ANALOG INPUT PATTERNS", Applied Optics, Vol. 26 No. 232, pp 4919-4930, 1987.

[13] "NEURAL WORKS PROFESSIONAL 2 SOFTWARE MANUAL" 1987, Volume1, pp.179-209 Neural-Ware, Inc. USA.

[14] "HNC EXPLORENET 3000 SOFTWARE MANUAL" April 1991, Chapter 18, ART2, HNC Inc. USA.

[15] G. Shafer, "A MATHEMATICAL THEORY OF EVIDENCE", Princeton Universal Press, 1975, Chapt. 3.

[16] Keith Godfrey (University of Western Australia, Nedlands W.A. 6009), Martin Keye (Unitronics Pty Ltd Technology Park W.A. 6102) "ARTIFICIAL NEURAL NETWORKS WORKSHOP MANUAL".

[17] E. Domany J. L. Van Hemmen, K. Schulten (Eds.) "MODELS OF NEURAL NETWORKS" , Springer-Verlag, 1991.

[18] Jones, William P., Hoskins, Josiah, "BACKPROPAGATION, A GENERALIZED DELTA LEARNING RULE" Byte Magazine, Oct. 1987.

[19] Igor Aleksander and Helen Morton, "AN INTRODUCTION TO NEURAL COMPUTING" Chapman & Hall, 1991.

[20] Richard P. Lippmann, "PATTERN CLASSIFICATION USING NEURAL NETWORKS" November 1989, IEEE Communications Magazine.

[21] Richard P. Lippmann, "AN INTRODUCTION TO COMPUTING WITH NEURAL NETS" IEEE ASSP Magazine April 1987.

[22] Bezdick, Sankar K Pal, editors,"FUZZY MODELS FOR PATTERN RECOGNITION", IEE PRESS, 1991.

[23] Kosko B, "FUZZINESS VS PROBABILITY" International Journal of General Systems, 17, N0. 2-3, 1990.

[24] Zadeh L A, "MAKING COMPUTERS THINK LIKE PEOPLE", IEEE Spectrum, 21, 26-32, 1984.

[25] Zadeh L A "FUZZY SETS", Information & Control, 8, 338-353, 1965.

[26] Bar-Shalom, Y.,"MULTITARGET-MULTISENSOR TRACKING: ADVANCED APPLICATIONS", pp.191-193.

[27] Dr. Martin, Dr. Nandagopal, "THE APPLICATION OF ARTIFICIAL NEURAL NETWORKS TO RADAR SIGNAL PROCESSING" Presented at ISSPA92, Tutorial Program.

[28] R. E. Gardner, "DOPPLER SPECTRA OF AIRBORNE TARGETS " Report from Naval Research Laboratory, Washington DC, March 1962 issue

[29] S. Lawrence Marple, "DIGITAL SPECTRAL ANALYSIS", Chapt. 7 , Prentice-Hall Inc.

[30] John Makhoul; "LINEAR PREDICTION: A TUTORIAL REVIEW" IEE Proceedings Vol. 63, April 1975.

[31] Sam P. Chaundhuri, Som Das, "NEURAL NETWORKS FOR DATA FUSION", IEEE, Sensor Data Integration, Inc. 342 Caterina Heights, Concord, MA 01742.

[32] A.J.R. Heading, M.D. Bedworth, "DATA FUSION FOR OBJECT CLASSIFICATION" IEEE Systems, Man and Cybernetics Conference, Charlottesville VA, October 1991.

[33] S.P. Luttrell, "THE USE OF BAYESIAN AND ENTROPIC METHODS IN NEURAL NETWORK THEORY", in Maximum entropy and Bayesian methods, pp. 363-370, J. Skilling, ed., Kluwe Academic Publishers, 1989.

[34] Dr. D. J. Kewley, HF Radar Division DSTO, "NOTES ON USING DEMPSTER SHAFER & FUZZY REASONING TO FUSE IDENTITY ATTRIBUTE DATA", SRL PO Box 1500, Salisbury SA 5108, Technical Memorandum SRL-0094-TM, Unpublished, August 1992.

[35] P S Hall, T K Garland-Collins, R S Picton, R G Lee, "RADAR" 1990 pp 32-40.

[36] R Hynes, R E Gardner, " DOPPLER SPECTRA OF S-BAND AND X-BAND SIGNALS" 1968.

[37] A Farina, " SIGNAL AND DATA PROCESSING FOR RADAR NETTING", IEEE International Conference Tutorial B-2, Washington, DC May 1990.

[38] D. Nandagopal, D.J Heilbronn, N.M Martin, I.C Potter and R.M Hawkes "CHARACTERISTICS OF DOPPLER MODULATION ASPECTS OF RADAR ECHOS FROM MOVING TARGETS" Radar Con 90, Adelaide, Australia 18-20 April 1990.

[39] Zadeh L A, " A REVIEW OF SHAFER'S ("A MATHEMATICAL THEORY OF EVIDENCE"), AI magazine, 5, 81-83, 1984.

[40] Prade H, " A COMPUTATIONAL APPROACH TO APPROXIMATE AND PLAUSIBLE REASONING WITH APPLICATIONS TO EXPERT SYSTEMS", IEEE Trans. Pattern and Machine Intelligence, 7, 260-283, 1985.