

AFIT/GCS/ENG/99M-10

TARGET POSE ESTIMATION FROM RADAR DATA
USING ADAPTIVE NETWORKS

THESIS
Andrew W. Learn
Captain, USAF

AFIT/GCS/ENG/99M-10

Approved for public release; distribution unlimited

DTIC QUALITY INSPECTED 2

19990409 096

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE March 1999		3. REPORT TYPE AND DATES COVERED Master's Thesis
4. TITLE AND SUBTITLE TARGET POSE ESTIMATION FROM RADAR DATA USING ADAPTIVE NETWORKS			5. FUNDING NUMBERS	
6. AUTHOR(S) Andrew W. Learn, Captain, USAF				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology 2950 P Street Wright-Patterson AFB, OH 45433-7765			8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/GCS/ENG/99M-10	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Dr. Louis Tamburino (937) 255-1115 ext 4389 AFRL/SNAT Bldg 620, Rm C3-K76 2241 Avionics Circle Wright-Patterson AFB, OH 45433-7321			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES Thesis Advisor: Dr. Steven Gustafson (937) 255-3636 ext 4598, sgustafs@afit.af.mil				
12a. DISTRIBUTION AVAILABILITY STATEMENT Approved for public release; distribution unlimited			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This research investigates and extends recent work by J.C. Principe at the University of Florida in target pose estimation using adaptive networks. First, Principe's technique is successfully extended to estimate both azimuth and elevation using SAR images. A network trained and tested using MSTAR data yields mean errors of less than six degrees in azimuth and five degrees in elevation. Second, the technique is applied to high-range resolution radar (HRR) signatures. Ground target (azimuth only) testing yields mean errors of less than 11 degrees for most classes. Air target testing for networks trained and tested on the same aircraft class yields mean errors under five degrees in azimuth and six degrees in elevation. However, cross-class estimation yields poor results. HRR signatures are analyzed to identify error sources and recommend ways to improve accuracy. Finally, Principe's novel mutual information training method is compared against traditional mean-squared-error training. Results show both methods are generally equivalent, but mutual information experiences convergence problems for some complex training sets. In general, adaptive network techniques demonstrate significant potential for improving the state of the art in target pose estimation. Both the estimation of elevation in SAR and the application to HRR are new and noteworthy successes.				
14. SUBJECT TERMS Pose Estimation, Target Recognition, Synthetic Aperture Radar, Radar Signatures, Neural Nets			15. NUMBER OF PAGES 113	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

AFIT/GCS/ENG/99M-10

TARGET POSE ESTIMATION FROM RADAR DATA USING
ADAPTIVE NETWORKS

THESIS

Presented to the Faculty of the Graduate School of Engineering
of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the
Requirements for the Degree of
Master of Science

Andrew W. Learn, B.S.

Captain, USAF

March 1999

Approved for public release; distribution unlimited

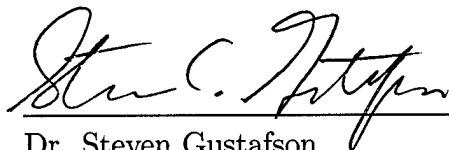
AFIT/GCS/ENG/99M-10

TARGET POSE ESTIMATION FROM RADAR DATA USING
ADAPTIVE NETWORKS

Andrew W. Learn, B.S.

Captain, USAF

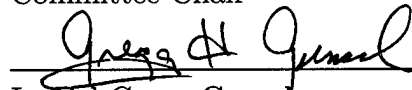
Approved:



Dr. Steven Gustafson
Committee Chair

3 Mar 99

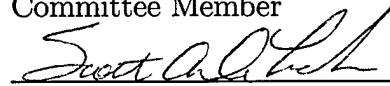
Date



Lt Col Gregg Gunsch
Committee Member

3 MAR 99

Date



Maj Scott DeLoach
Committee Member

2 Mar 99

Date

Acknowledgements

I would like to thank my advisor, Dr. Steven Gustafson, for his constant words of confidence and encouragement. Through the long, brain-twisting experience of trying to understand the mutual information algorithms, he gave me hope that I would actually figure it out in the end. I also thank Dr. Jose Principe and Mr. Dongxin Xu from the University of Florida, whose work is the basis of this thesis. Their insight, explanations, and occasional updates were very helpful.

Many thanks are also due Dr. Lou Tamburino from the Sensors Directorate of the Air Force Research Laboratory. His enthusiasm for my work and his expectations of great results gave me the spark of excitement I needed to stay driven toward the goal. Our discussions stirred up many ideas for approaches to try during this research.

But more than any other person, my thanks, appreciation, and love go to my beautiful wife, Patti. Her continuous support and encouragement were what carried me through many days of frustration and many nights with little sleep. She listened to me complain about partial derivatives with nary a complaint, and she oohed and aahed at my "pretty pictures" when I needed a pat on the back. I also thank my wonderful children, whose understanding and devotion helped me remember what the most important things in life really are.

Thanks also to my parents and all my extended family for your support and for always making the farm a haven where I could recuperate and relax.

Finally I thank God, who gave me strength, peace, and perspective through it all. Long after this thesis is gathering dust on a shelf, I pray that what He has done in my life during this year and a half will plant seeds that make a difference for eternity.

Andrew W. Learn

Table of Contents

	Page
Acknowledgements	iii
List of Figures	v
List of Tables	vi
Abstract	vii
 I. Introduction	 1
1.1 Background	1
1.1.1 Target Recognition Overview	1
1.1.2 Model-based vs. Library-based ATR	2
1.1.3 The Importance of Pose	3
1.1.4 Radar Systems for Target Recognition	4
1.1.5 Difficulties with Pose Estimation for Radar	6
1.2 Adaptive Network Pose Estimation	7
1.3 Problem Statement	8
1.4 Organization	9
 II. Literature Review	 10
2.1 Introduction	10
2.2 Pose Estimation as a Product of Target Classification	10
2.2.1 Persistent Scatterers (Dudgeon)	11
2.2.2 Feature Space Trajectories (Casasent)	12
2.2.3 Information Theoretic Decision Tree (Cyganski and Vaz)	13
2.3 Independent Pose Estimation	14

	Page
2.3.1 Two Degree of Freedom Pose Estimation with Neural Networks (Agarwal and Chaudhuri) . .	15
2.3.2 Pose Estimation Using Information Theoretic Train- ing Criteria (Principe)	16
2.4 Comparison and Analysis	19
2.4.1 Dependence on Classification Method	19
2.4.2 Use of Features vs. Raw Images	20
2.4.3 Common Limitations	20
2.5 Needed Areas of Research	21
2.6 Summary	22
III. Methodology	23
3.1 Training Criteria	23
3.1.1 Network Notation	23
3.1.2 Mutual Information (MI) Training	24
3.1.3 Mean-Squared-Error (MSE) Training	35
3.2 Data Sets Used	36
3.2.1 SAR	36
3.2.2 HRR Ground Targets	36
3.2.3 HRR Air Targets	38
3.3 Description of Experiments	38
3.3.1 Validation of Principe's Results	38
3.3.2 Image Understanding for SAR Azimuth Estima- tion	39
3.3.3 Application to HRR Azimuth Estimation	39
3.3.4 Application to Two Degrees of Freedom in SAR	39
3.3.5 Application to Two Degrees of Freedom in HRR	41
3.4 Equipment and Software	42
3.5 Summary	42

	Page
IV. Results	43
4.1 Validation of Principe's Results	43
4.1.1 180 Degree Azimuth Estimation	43
4.1.2 360 Degree Azimuth Estimation	44
4.2 Image Understanding for SAR Azimuth Estimation . .	44
4.2.1 Weight Characteristics for 180 Degree Estimation	47
4.2.2 Weight Characteristics for 360 Degree Estimation	47
4.3 Application to HRR Azimuth Estimation	50
4.3.1 180 Degree Azimuth Estimation	50
4.3.2 Design Change for 90 Degree Azimuth Estimation	50
4.3.3 Performance Analysis	54
4.4 Application to Two Degrees of Freedom in SAR	57
4.4.1 Concentric Circle Representation	57
4.4.2 Cylinder Representation	61
4.4.3 Performance Analysis	61
4.5 Application to Two Degrees of Freedom in HRR	66
4.5.1 Synthetic Data Testing	66
4.5.2 Performance Analysis	68
4.6 Summary	72
V. Conclusions and Recommendations	73
5.1 Conclusions	73
5.1.1 Advantages of Adaptive Networks for Pose Es- timation	73
5.1.2 SAR Image Understanding	74
5.1.3 HRR Azimuth Estimation (Ground Targets) .	74
5.1.4 SAR Azimuth and Elevation Estimation . . .	74

	Page
5.1.5 HRR Azimuth and Elevation Estimation (Air Targets)	75
5.1.6 Network Architecture	75
5.1.7 Comparison of Training Criteria	75
5.2 Recommendations	76
5.2.1 SAR Ground Targets	76
5.2.2 HRR Ground Targets	76
5.2.3 HRR Air Targets	77
5.2.4 Reducing End Effects	78
5.2.5 Leave-One-Out Testing	78
Appendix A. SAR Two-Degree-of-Freedom Code	79
A.1 Mutual Information Network Training	79
A.2 Mutual Information Network Testing	91
Bibliography	99
Vita	101

List of Figures

Figure		Page
1.	Typical target recognition process	2
2.	Definition of azimuth and elevation angles (Willis, 1991) . . .	4
3.	Creation of HRR imagery	5
4.	Creation of SAR imagery	6
5.	Templates based on persistent scatterers	11
6.	Feature space trajectory for a single target	13
7.	Alternate Network Architectures (Principe)	17
8.	Pose Estimation in Output Space (Principe)	18
9.	Linear network for SAR azimuth estimation	24
10.	Information conveyed by probability distribution functions . .	25
11.	Actual azimuths of 52 target exemplars represented by place- ment on unit circle	26
12.	Gaussian probability estimate for a single exemplar	26
13.	Template pdf representing actual target azimuths for 52 exem- plars	27
14.	Network output generated by a set of exemplars using small initial random weights	29
15.	Pdf for network output using initial random weights	30
16.	MI training (iteration 5)	34
17.	MI training (iteration 15)	34
18.	MI training (iteration 25)	34
19.	MSE training (iteration 100)	37
20.	MSE training (iteration 200)	37
21.	MSE training (iteration 500)	37
22.	Concentric circle representation for 2-DOF pose	40

Figure		Page
23.	Cylinder representation for 2-DOF pose	41
24.	SAR 180 degree results: trained on BMP2/sn-c21, tested on BMP2/sn-c21	45
25.	SAR 180 degree results: trained on BMP2/sn-c21, tested on T72/sn-132	45
26.	SAR 360 degree results: trained on BMP2/sn-c21, tested on BMP2/sn-c21	46
27.	SAR 360 degree results: trained on BMP2/sn-c21, tested on T72/sn-s7	46
28.	Sample BMP2 images at varying azimuth	47
29.	Network weights: MI training for 0-180 degrees on BMP2/sn-c21	48
30.	Network weights: MSE training for 0-180 degrees on BMP2/sn-c21	48
31.	Network weights: MSE training for 0-360 degrees on BMP2/sn-c21	49
32.	Network weights: MSE training for 0-360 degrees on T72/sn-s7	49
33.	HRR 180 degree results: trained on BMP2/sn-c21, tested on BMP2/sn-c21	51
34.	Four similar HRR profiles for BMP-2 caused by vehicle symmetry	51
35.	HRR 90 degree results: trained on 5 vehicles, tested on T72/sn-s7	53
36.	HRR 90 degree results: trained on 5 vehicles, tested on ZSU23/4	54
37.	HRR 90 degree results: Mutual information training/estimation for BMP2	55
38.	Template pdf for mutual information training on BMP2 . . .	55
39.	Network outputs for BMP-2 training signatures	56
40.	Sample target visual width and actual BMP-2 signature width	57
41.	Sample 2S1 images at varying elevation	58
42.	SAR concentric circle output: trained on 2S1, tested on 2S1 .	59

Figure		Page
43.	SAR concentric circle azimuth results: trained on 2S1, tested on 2S1	60
44.	SAR concentric circle elevation results: trained on 2S1, tested on 2S1	60
45.	SAR cylinder output: trained on 2S1, tested on 2S1	62
46.	SAR cylinder output (side view)	63
47.	SAR cylinder output (top view)	63
48.	SAR cylinder azimuth results: trained on 2S1, tested on 2S1	64
49.	SAR cylinder elevation results: trained on 2S1, tested on 2S1	64
50.	Weight visualization for SAR concentric circle network	65
51.	Weight visualization for SAR cylinder network	65
52.	Concentric circle training results using mutual information . .	67
53.	Cylinder training results using mutual information	67
54.	Output space for HRR aircraft pose estimation	68
55.	HRR 2-DOF output: trained on aircraft E, tested on aircraft E	69
56.	HRR 2-DOF azimuth results: trained on aircraft E, tested on aircraft E	69
57.	HRR 2-DOF elevation results: trained on aircraft E, tested on aircraft E	70
58.	HRR 2-DOF azimuth results: trained on aircraft A, tested on aircraft B	70
59.	Sample aircraft HRR signatures at identical pose	71

List of Tables

Table		Page
1.	Validation testing for 0-180 degree training	43
2.	Validation testing for 0-360 degree training	44
3.	HRR testing for MLP network trained with mean-squared-error	53
4.	SAR 2-DOF concentric circle results (MSE training on 2S1) .	58
5.	SAR 2-DOF concentric circle results (MSE training on three vehicles)	59
6.	SAR 2-DOF cylinder results (MSE training on 2S1)	61
7.	Aircraft HRR 2-DOF results (MSE training and testing on same aircraft)	68

Abstract

Pose estimation is used to speed the target recognition process by reducing the number of candidate images that must be searched to find the best match. J.C. Principe at the University of Florida recently developed an adaptive network that estimates target azimuth from synthetic aperture radar (SAR) images. He also developed a novel network training method using mutual information measures. This research investigates and extends Principe's initial work as follows.

First, the adaptive network technique is successfully extended to estimate both azimuth and elevation using SAR images. A network trained and tested using MSTAR data yields mean errors of less than six degrees in azimuth and five degrees in elevation. This ability to predict elevation is a significant step forward in pose estimation, since most current systems predict azimuth only.

Second, the adaptive network technique is applied to high-range resolution radar (HRR) signatures. Ground target (azimuth only) testing yields mean azimuth errors of less than 11 degrees for most classes. Air target testing for networks trained and tested on the same aircraft class yields mean errors of less than five degrees in azimuth and six degrees in elevation. However, cross-class estimation yields poor results. HRR signatures of both ground and air targets are analyzed to identify sources of error, and recommendations are made to improve accuracy.

Finally, the performance of the mutual information training method is compared against traditional mean-squared-error network training. Results show both methods are generally equivalent, but mutual information experiences convergence problems for some complex training sets.

In general, adaptive network techniques demonstrate significant potential for improving the state of the art in target pose estimation. Both the estimation of elevation in SAR and the application to HRR are new and noteworthy successes.

TARGET POSE ESTIMATION FROM RADAR DATA USING ADAPTIVE NETWORKS

I. Introduction

1.1 Background

Automatic target recognition (ATR) is a high priority research area for the United States Air Force. Reliable target recognition systems will be essential in fulfilling the 21st century Air Force vision to "find, fix or track and target anything that moves on the surface of the earth" (USAF, 1996). A critical part of the target recognition process is estimating target pose. This research investigates and extends a recently developed approach to pose estimation that uses information theoretic principles to train a linear network.

1.1.1 Target Recognition Overview. For the purposes of this research, an ATR system is defined as *an automated system that detects and correctly identifies enemy targets in areas of military interest, based on imagery of the area (photographic, radar, infrared, etc.).* The main goals of an ATR system are the following:

1. *Speed* - the ability to process large numbers of images in regions of interest much faster than human analysts.
2. *Accuracy* - the ability to find and identify all the targets present while minimizing false detections and misclassifications.
3. *Reduced cost* - the reduction of both the financial and human cost associated with intelligence and combat missions by identifying targets remotely and striking exactly when and where needed.

The United States military has devoted significant resources over many years to the ATR problem, but progress has been slow and incremental. Pattern recognition in general is a complex problem and has been resistant to any single breakthrough solution. Military target recognition presents additional difficulties such as long ranges, all-weather and nighttime environments, and the partial occlusion (obstructed view) or camouflage of targets.

While there are many varieties of ATR systems, Figure 1 illustrates the typical recognition process (Dudgeon, 1993). Because of the complexity of the problem, ATR is usually approached in multiple stages of increasing refinement. From the raw image data, some simple characteristic (e.g. strength of radar return) is used to identify objects of interest. For each object, the system extracts a set of distinguishing characteristics called *features*. Features may be statistical properties such as moments, geometrical properties such as line and vertex locations, or other ad hoc properties of the image. A classifier then uses these features to determine the identity of the object by comparison with typical samples, or *exemplars*, of potential targets. Classification may be performed through various means such as Bayesian statistics or neural networks.

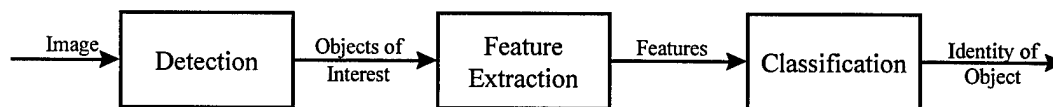


Figure 1 Typical target recognition process

1.1.2 Model-based vs. Library-based ATR. An important characteristic of an ATR system is the *source* of the target exemplars used for classification. There are two competing paradigms: *model-based* and *library-based*. Both methods have advantages and drawbacks.

Model-based ATR systems use computer models to predict the appearance of various targets. The predicted image is then matched against the detected object of interest. Model-based systems are flexible and have lower memory requirements than library-based systems: they can generate an infinite number of target images from every possible angle, yet they need only enough memory to store the target models. However, model-based systems are very sensitive to the accuracy of the computer model predictions.

Library-based ATR systems use libraries of actual measured target images to match against the detected object. This method provides higher confidence that the exemplars truly represent the actual target appearance. However, there are two main disadvantages to library-based systems: large amounts of memory are required to store all the target images, and data collection using real sensors and real targets is expensive. Thus measured data sets are usually smaller than might be desired.

For the above reasons, many systems use a combination of library-based and model-based methods.

1.1.3 The Importance of Pose. In both model-based and library-based ATR systems, target pose is a critical feature. Pose is simply the sensor viewing angle of the target. It is typically represented by two angles, azimuth and elevation, as shown in Figure 2. Elevation is also called the sensor depression angle, because for ground targets the sensor is looking down on the target from above. The terms depression and elevation will be used interchangeably.

Knowing object pose is important for target recognition because objects of interest look different when seen from different points of view. Consequently the characteristics of the feature vector extracted for a specific target also change, often dramatically, with viewing angle.

To achieve acceptable performance, library-based ATR systems must store images taken from many different pose angles for each target type. Hundreds of images

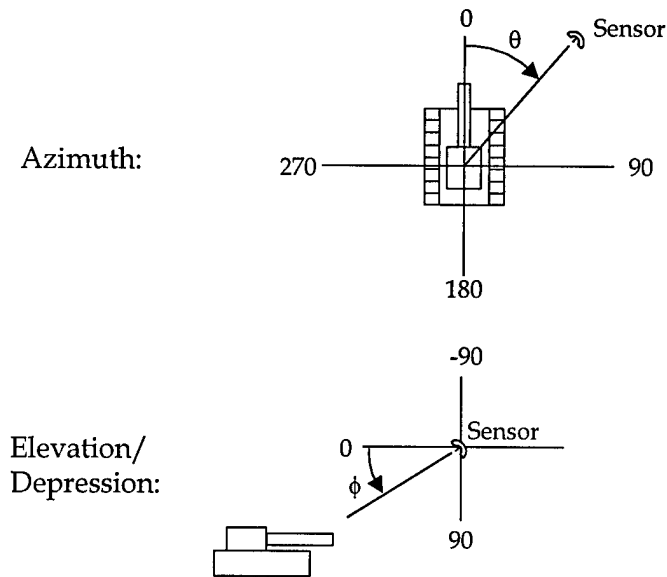


Figure 2 Definition of azimuth and elevation angles (Willis, 1991)

may be required to adequately represent each target, and an ATR system may be required to recognize tens of targets. This presents a formidable search and match problem. If the pose is unknown, the worst case will require searching through exemplars of all possible targets from all pose angles—360 degrees in azimuth and 0 through 90 degrees in elevation for ground targets or -90 to +90 degrees elevation for air targets.

Knowing target pose is a key to efficient search. A reliable pose estimate allows the system to prune the search space significantly. This reduction is valuable, because ATR systems tend to be computationally intensive but operate in a combat environment in which real-time performance is critical. Thus accurate pose estimation is an important capability for any ATR system.

1.1.4 Radar Systems for Target Recognition. ATR systems may use a variety of sensors, including optical, infrared, and radar. Most ATR research, however, is focused heavily on radar systems, which provide the long-range, all-weather, day or

night imaging capability required by most military operational environments. This research considers pose estimation for radar-based systems only.

Two main types of radar are used for target recognition: high range resolution (HRR) radar and synthetic aperture radar (SAR).

HRR may be used for both air targets and ground targets. The target is illuminated with a radar beam, and the energy scattered from the target is measured over time. This creates a one-dimensional target signature with radar return strength as a function of range (Figure 3).

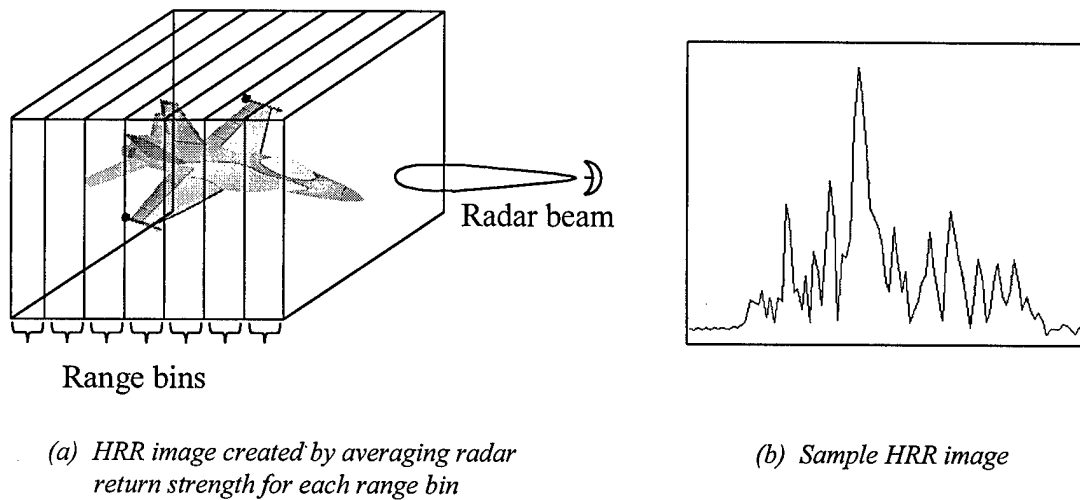


Figure 3 Creation of HRR imagery

In SAR, the same type of range measurement is taken, but the movement of the radar platform is used to create a *synthetic aperture*. A number of range signatures are collected over time and combined to produce a two-dimensional image (Figure 4). Because of the time lapse required to synthesize SAR images, SAR is used primarily for slow-moving ground targets. (The high velocities of air targets make aircraft SAR images blurry and ineffective.) Thus HRR remains the principal radar type used for aircraft recognition.

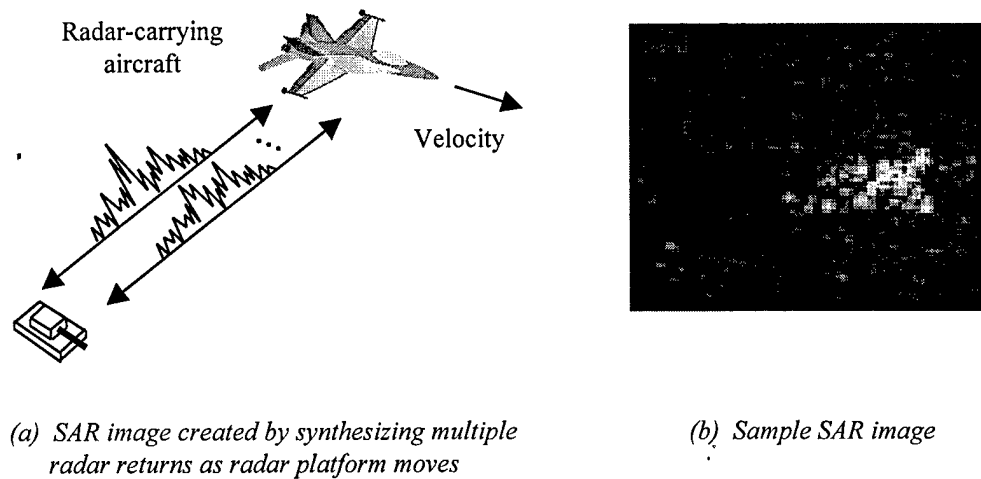


Figure 4 Creation of SAR imagery

1.1.5 Difficulties with Pose Estimation for Radar. Both SAR and HRR present problems for pose estimation due to the complex scattering properties of radar. The energy reflected for any particular range is a combination of returns from different scattering centers, bounced waves, and phase interference effects. A small change in viewing angle can produce a large change in the radar signature for a target. Thus HRR and SAR signatures may change rapidly with changing pose, producing a scintillating or "glinting" effect.

Because of these problems, a common approach to estimating pose for air targets does not use HRR range signatures at all, but rather uses radar tracking to determine the direction of the aircraft. The direction vector provides a rough estimate of the orientation of the aircraft. However, multiple factors such as crosswinds, angle of attack, and aircraft turns make this type of estimate highly uncertain.

The low velocity of ground targets permits pose estimation using SAR imagery. In recent years there has been significant development of pose estimation techniques for two-dimensional images. However, many of the techniques have been developed by the robotics community and are designed to operate on visual images of well-known and well-defined objects. The "glinty" characteristics of SAR images make

it difficult to estimate pose by standard techniques because vertices, lines, and shapes are variable and hard to identify in SAR images. Thus target pose estimation in SAR remains a difficult problem.

1.2 Adaptive Network Pose Estimation

Principe and his colleagues at the University of Florida have recently developed methods for SAR pose estimation using adaptive-weight networks trained with both traditional and innovative techniques (Principe, 1998b). The methods work with either a simple linear network or a multi-layer perceptron architecture. To train the network, raw SAR images are presented as input (one pixel per node), and the output is the target pose.

The traditional training technique uses gradient descent to adapt network weights. The network error measure is the mean-squared-error between the current network output and the desired network output. The derivative of the mean-squared-error is used as the training criterion to update network weights in the direction of minimum error.

The new training technique developed by Principe is based on the information theory principle of maximizing mutual information. Information theory has been used for decades to analyze the data transmission properties of communications systems. For pose estimation, it is used as a network training criterion in place of the mean-squared-error. Mutual information measures are used to maximize the transfer of pose-related information from the network input (raw image) to the network output (pose angle estimate). Performance was found to be equivalent to the mean-squared-error technique for the cases tested.

Work to date has been performed on SAR images to estimate target pose for azimuth only (one degree of freedom). Preliminary results for these techniques are very promising, with a mean error of less than five degrees for a network trained

on a single target (Principe, 1998a). Extensions and applications of these pose estimation methods are the subjects of investigation for this research.

1.3 Problem Statement

The technique of pose estimation using adaptive networks is investigated and extended in three ways.

First, the two training methods are validated and their performance is compared. Weight analysis is used to determine what SAR image characteristics the network uses for pose estimation.

Second, the technique is applied to pose estimation of ground targets using HRR signatures. (Previous work was performed for SAR only.) The accuracy of pose estimation using HRR is determined and the results are analyzed.

Finally, the method is extended to the two degree-of-freedom case. Previous work at the University of Florida demonstrated the success of the method for estimating azimuth; this research uses it for estimating both azimuth and elevation. Two degree-of-freedom tests are performed for both SAR ground targets and HRR air targets.

For all extensions considered, the relative performance of the two training methods (mutual information and mean-squared-error) is compared. In addition, the effectiveness of pose estimation for multiple targets is also investigated. Since the identity of the target is unknown at the time of pose estimation, it is desirable to design a generalized system that can determine the pose for all possible target classes. Single networks are trained and tested on several different target classes to establish the feasibility of a generalized pose estimator.

1.4 Organization

The remainder of the thesis is organized as follows. Chapter 2 reviews current pose estimation techniques and provides an in-depth overview of the initial work done at the University of Florida. Chapter 3 presents the methodology employed, including an overview of the network architecture, the mutual information and mean-squared-error training criteria, and the target data. Chapter 4 discusses the results of system training and testing, and Chapter 5 presents conclusions and recommendations.

II. Literature Review

2.1 Introduction

While many different methods are currently available for estimating object pose, most are highly dependent on the type of image (e.g., optical or radar) and the operating environment (e.g., assembly line or military reconnaissance). For example, the robotics community often deals with optical images of well-known and well-defined objects. In this environment matching object vertex and line locations to a computer model may be an efficient pose estimation technique. However, the same method is not effective for SAR ATR systems, where the target is unknown and scintillation effects prevent vertex and line identification.

This chapter reviews a number of current pose estimation techniques that are specifically applicable to radar-based target recognition. The various algorithms and their accuracies are compared, and common problems and limitations are identified.

2.2 Pose Estimation as a Product of Target Classification

The ultimate goal of a target recognition system is classification. When an ATR system is designed, pose estimation is usually either an intermediate step in the classification process or a by-product of classification. Thus most target pose estimation research is performed and reported in the context of a larger ATR research effort, which results in two typical effects:

1. The pose estimation step is integrated into a larger target recognition system.
2. The pose estimation algorithm chosen is closely linked to the specific method used for classification.

This section reviews three recent pose estimation techniques that exhibit this close tie to the target classification process.

2.2.1 Persistent Scatterers (Dudgeon). Dudgeon and other researchers at the Massachusetts Institute of Technology (MIT) have developed a model-based ATR system intended for use on SAR and inverse SAR (ISAR) images (Dudgeon, 1994). Their system seeks to overcome the difficulties of SAR scintillation by identifying "persistent scatterers."

Scintillation in SAR is caused by points on the target (called scattering centers) which produce high radar returns only over small ranges of pose angle. As the vehicle orientation changes relative to the radar, these locations produce bright spots in the SAR image that persist for one or two degrees and then quickly fade. However, Dudgeon discovered that there are typically about 10 locations on a target that each produce a bright spot which persists over a 20 degree range in azimuth. These persistent scatterers have different locations for each target, and their locations change for a particular target as the vehicle orientation changes.

The MIT system creates a set of target-centered templates that identify the location of the persistent scattering points. A unique set of azimuth-dependent templates is developed for each target (Figure 5). When an unknown SAR image is presented, the system attempts to match it against all templates. The template that generates the highest correlation score is selected as the winner. Thus the system produces a simultaneous hypothesis of the target identity and pose.

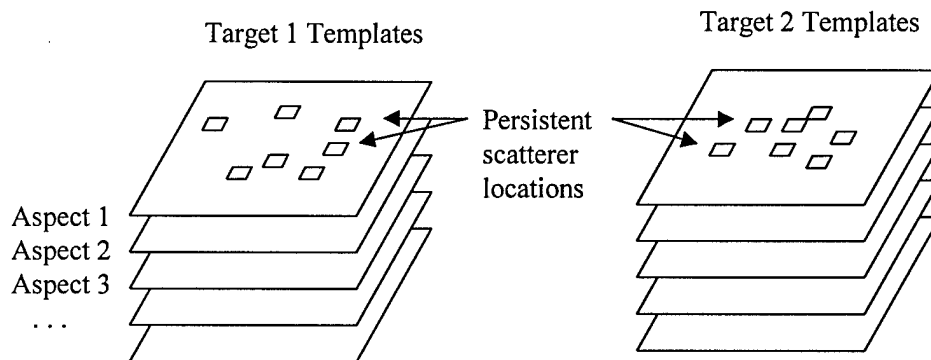


Figure 5 Templates based on persistent scatterers

The template-based system performs pose estimation for azimuth only (one degree of freedom). The templates were developed for images at 19 degrees elevation, but were tested on images from 15 to 32 degrees elevation. The RMS error for the azimuth estimate was reported as six degrees, which seems to indicate that the persistent scatterers were not highly dependent on elevation angle.

2.2.2 Feature Space Trajectories (Casasent). At Carnegie Mellon University, Casasent and his colleagues have developed a unique type of ATR system that uses "feature space trajectories" (FST's). This system has been developed and tested for both SAR (Casasent, 1997) and infrared (Casasent, 1998) sensor data.

The FST approach creates a representation of the target in feature space. Casasent defines features using Karhunen-Loeve eigenvectors for SAR and Fourier transforms for infrared. Images of a target taken at various azimuth angles are projected as points in feature space; adjacent aspect angles are connected by lines (Figure 6). Separate feature space trajectories are generated for each possible target. When the system processes an unknown target image, it extracts the features and plots the corresponding point in feature space. The FST nearest to that point denotes the most likely target class and the line segment on the FST indicates the target pose.

It may be computationally difficult to find the distance between the unclassified point and each FST in n -dimensional space, particularly if n , the number of features, is large. Casasent overcomes this problem by training a neural network to perform the distance measurement.

As in Dudgeon's system, the FST method simultaneously generates target identity and pose. The FST system was able to estimate azimuth to within 8 degrees for SAR and 6.9 degrees for infrared. Casasent highlights the interesting fact that part of the pose error for SAR images was due to 180 degree mistakes. The FST for some targets looked similar to a double circle, causing the system to choose

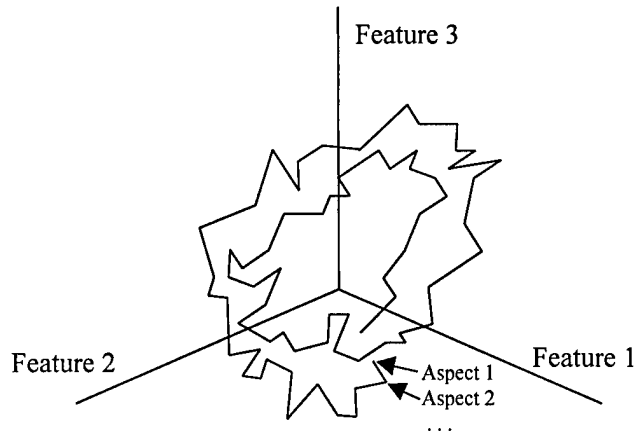


Figure 6 Feature space trajectory for a single target

the wrong segment of the FST as the pose estimate. This effect may be attributed to the symmetrical appearance of many vehicles in SAR imagery. It is also significant that during infrared system testing, the FSTs were created using training aspect views separated by 15 degrees, but the pose estimation yielded an average error of 6.9 degrees. This result demonstrates that the system can interpolate along an FST line segment to obtain a pose estimate of finer resolution than the separation between training images.

2.2.3 Information Theoretic Decision Tree (Cyganski and Vaz). The research of Cyganski and Vaz at the Worcester Polytechnic Institute seeks a computationally efficient ATR system by using a decision tree structure for classification (Cyganski, 1997). In this approach, the target exemplars are the leaves of a binary tree. Given an unknown image, the decision tree attempts to make decisions at each node that lead to the exemplar (leaf) that most closely matches the unknown target. The advantage of the decision tree is its linear increase in processing time for an exponentially increasing number of exemplars. However, Cyganski and Vaz note that implementations of decision trees typically fail for one of the following reasons:

1. Image dimensionality is reduced by feature extraction, which leads to unacceptable loss of discrimination ability.

2. Forced decisions during search can completely eliminate a region of the tree that would yield the best match.

The information theoretic decision tree (ITDT) method seeks to overcome these weaknesses. All calculations are performed on full SAR images to prevent the loss of information caused by projecting the image into a lower dimensional feature space. In addition, target exemplars can be repeated in several branches of the tree in order to avoid forcing arbitrary decisions before enough evidence has been accrued. The decision process is then made at each node using entropy and mutual information measures based on information theory. Each decision seeks to maximize the information gained about how to discriminate between exemplars. As in the previously discussed research, the end result of the ITDT search is simultaneous classification and pose estimation.

Cyganski and Vaz report a mean pose estimation error of less than one degree. This accuracy is better than any of the other estimation methods reviewed. However, a closer examination of the testing procedures indicates these results are preliminary and highly optimistic. The tests were performed on a single target, and the entire set of training images was included in the test set after corruption with simulated SAR speckle noise. In addition, the range of azimuth angles tested was restricted to 120 degrees. (Test results showed, similar to Casasent, that large errors due to 180 degree symmetry occurred when the azimuth range was not restricted.) Thus more thorough testing will likely be required to validate the ITDT method.

2.3 Independent Pose Estimation

While the above research links the pose estimation method closely with the end goal of target classification, pose estimation may also be treated as an independent problem. This section discusses two techniques that fall into this category. Both

of the following research efforts do have an ultimate goal—either aircraft tracking or target recognition. However, the pose estimation algorithm developed is not bound to the technique used to achieve the end goal.

2.3.1 Two Degree of Freedom Pose Estimation with Neural Networks (Agarwal and Chaudhuri). Agarwal and Chaudhuri have developed a neural network approach to pose estimation of aircraft using binary (silhouette) camera images (Agarwal, 1998). The system was developed to aid in aircraft tracking. It employs a multi-layer perceptron (MLP) architecture trained by back-propagation—a configuration that is well-documented and widely used in pattern recognition problems (Bishop, 1995).

The input to the MLP consists of a set of features that characterize the aircraft shape. Agarwal uses as features nine principal axis moments (up to fourth order moments), which provide translation, scale, and rotation invariance. The MLP produces an estimate of the aircraft azimuth and elevation as output.

Because of symmetry considerations for the silhouette images, the network was trained and tested only for a quarter hemisphere of poses: zero through 90 degrees in both azimuth and elevation. A total of 236 training images and 40 test images of the X-29 aircraft were used.

Two architecture variations were tested. In the initial system, a single MLP was trained on the entire set of training images. A more complex architecture was also developed using Kohonen clustering, which divided the training inputs into four groups. A separate MLP was then trained for each group. This procedure allowed each MLP to better learn its functional approximation over a smaller and less complex set of images.

During testing, the single MLP predicted the pose to within 10 degrees in both azimuth and elevation. The Kohonen architecture performed even better with most errors less than four degrees in both azimuth and elevation.

Agarwal and Chaudhuri's work is significant in that it provides a two degree-of-freedom (DOF) estimate: both azimuth and elevation. Prediction accuracy of the Kohonen network architecture ranked among the best of all methods examined. However, the binary imagery and the limited range of angles tested may produce results more optimistic than could be achieved under more realistic scenarios. Nevertheless, the adaptive network pose estimation technique shows significant promise, as will be reinforced by the research of Principe discussed below.

2.3.2 Pose Estimation Using Information Theoretic Training Criteria (Principe).

Principe at the University of Florida has recently developed a method for SAR pose estimation that bears many similarities to Agarwal's neural network estimator; however, it utilizes unusual techniques for network training based on information theory (Principe, 1998a) (Principe, 1998b). Because Principe's methods are the subject of further investigation in this thesis, the system will be reviewed here in more detail than the previously mentioned methods.

2.3.2.1 System Architecture. Principe's network architecture is similar to Agarwal's single MLP system. However, while Agarwal uses just nine image features as input, Principe's system uses the entire SAR image. The image is normalized to a grayscale range from 0 to 255 and clipped to a size of 80 by 80 pixels to include the vehicle and radar shadow. All 6400 pixels are then fed to the network as the input vector. The network produces an azimuth estimate (one DOF) at the output. Principe examined two alternate network architectures to compare results: a simple linear network and a multi-layer perceptron using three hidden nodes. Both architectures are shown in Figure 7.

An interesting feature of these architectures is the representation of the azimuth estimate. While Agarwal's network estimates a limited range of azimuth angles, Principe seeks to provide 360 degree coverage. To do so, he represents the periodic nature of azimuth angle (where 0 degrees equals 360 degrees) with two output nodes,

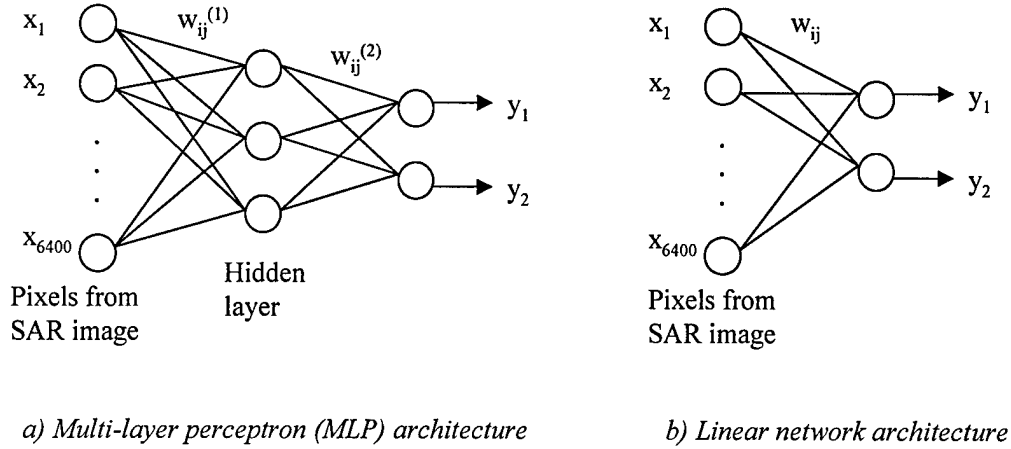


Figure 7 Alternate Network Architectures (Principe)

y_1 and y_2 . Figure 8 illustrates how the network is trained so that a plot of the output space for the training points produces a unit circle. When an unknown image is presented to the network, the output point may fall anywhere within the two-dimensional space. The azimuth is then estimated by the radial position of the point relative to the circle of training points.

2.3.2.2 Training Criteria. The key feature of Principe's technique, however, is the network training method. Both the linear network and MLP are trained by iterative weight adjustment using gradient descent, as described in (Bishop, 1995). But instead of using the typical mean-squared-error (MSE) criterion, the system uses the information theoretic criterion of maximizing mutual information (MI) between the actual network output and the desired network output. The details of the mutual information training method are reviewed in Chapter 3.

In order to compare the mutual information technique with more traditional methods, a similar network was trained using the mean-squared-error criterion and tested on some of the same data sets.

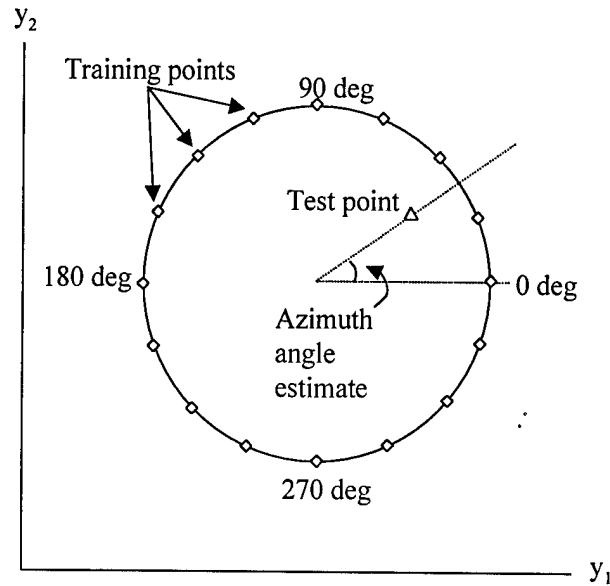


Figure 8 Pose Estimation in Output Space (Principe)

2.3.2.3 Test Results. Principe tested the system on both the MSTAR (DARPA, 1997) and MSTAR II (DARPA, 1998a) data sets. Results were reported in (Principe, 1998c). A network was trained on 0-180 degree images of a single vehicle at a single elevation and tested on five different classes at various elevations; the maximum estimation error was 10 degrees. Another network was trained on two different vehicle classes at three different elevations and again tested on the entire data set. The maximum estimation error for this more thoroughly trained system was less than eight degrees.

Results for 360 degree training and testing were not as accurate as for the 180 degree range. Although exact numbers were not given, Principe reports symmetry problems as experienced by other researchers. The 360 degree system was difficult to train by the mutual information method, and once trained the system would sometimes make estimates 180 degrees from the true azimuth.

The two variants of system architecture made no difference in the test results. Principe reports that the linear network performed just as well as the MLP, presumably because "the input space is so high dimensional that for this problem it is possible to find a linear projection that preserves the pose" (Principe, 1998c).

Finally, a significant result pertains to the comparison of network training criteria. Principe reports that the network trained with mutual information and the network trained with the traditional mean-squared-error "performed at the same level" (Principe, 1998c). This result raises important questions about the relative merits of the two methods, as will be discussed in Section 2.4.

2.4 Comparison and Analysis

The pose estimation techniques discussed above have all shown merit, although some have been more thoroughly tested than others. The estimation accuracy varies but is on the same order of magnitude—within 10 degrees in azimuth. The following sections compare the techniques with respect to their dependence on classification, use of features, and common limitations.

2.4.1 Dependence on Classification Method. One of the main divisions between pose estimation techniques is the degree of dependence on the classification method. As stated previously, the pose estimation algorithms of Dudgeon, Casasent, and Cyganski and Vaz were integrated into a larger target classification system. The ability to estimate pose thus depended on the exemplar representation and the decision mechanism used to classify the unknown object.

Such dependence may be acceptable if a single integrated target recognition technique is sought. However, real ATR systems such as MSTAR are typically modular in nature, where initial indexing by pose may be independent of the final classification method (MSTAR, 1998). Thus treating pose estimation independently

may lead to better estimation techniques than forcing pose estimation to support the requirements of classification.

For these reasons, the search for an independent optimal pose estimation method seems valuable. The methods of Agarwal and Principe have distinct promise in this regard.

2.4.2 Use of Features vs. Raw Images. Another major distinction between techniques is the use of features. The majority of pose estimation methods (and target recognition systems in general) use features rather than raw images. However, features must be carefully chosen to simultaneously minimize the data that must be handled and maximize the amount of information which remains to achieve the objective. Optimal feature selection is difficult; the best results are sometimes discovered only by trial and error.

The work of Willis (Willis, 1991) illustrates the difficulty of finding optimal features. Willis' thesis effort is devoted to finding optimal shape features for infrared pose estimation. However, his results optimize pose-related features for a single target only; extension to other targets or other types of imagery is not guaranteed. Despite difficulties such as these, using features rather than raw images is often viewed as the most practical solution both computationally and algorithmically.

It is not coincidental that of all the recent methods surveyed, only Principe and Cyganski use the entire target image, and both these methods are based on information theory. One of the fundamental premises of information theory is using all the information available in the given data and avoiding artificial reduction of the data through feature extraction. If the complexity can be adequately handled, using complete exemplar images should maximize the performance of either a classifier or pose estimator.

2.4.3 Common Limitations. A number of limitations repeatedly surfaced in the reviewed literature. Symmetry problems were the most common. The symmetrical appearance of ground vehicles makes it difficult to distinguish views separated by 180 degrees. In practice, this problem simply increases the search space. For example, if a target azimuth is estimated at 60 degrees, then the views near to both 60 degrees and 240 degrees must be searched for a best match. While it is desirable to distinguish views 180 degrees apart, searching both symmetrical aspects remains a great improvement over a 360 degree search.

Another common limitation was the lack of elevation estimate. Only Agarwal's system met this challenge—probably due to the importance of distinguishing different elevations for aircraft, which could conceivably be observed at any possible orientation. All other systems were designed for ground targets and sought to provide accurate azimuth estimates over a range of elevations—seeking commonality over elevations rather than distinguishing between them. While the elevation range of interest is much more limited for ground targets—10 to 40 degrees for MSTAR extended operating conditions as described in (Keydel, 1996)—an elevation estimate would still be beneficial in further reducing the search space.

Finally, the literature distinctly lacks pose estimation techniques for HRR radar for either ground or air targets. Some methods have been developed to refine trajectory-based aircraft pose estimates, as in (Libby, 1996), but no methods were found for estimating pose based completely on HRR signatures.

2.5 Needed Areas of Research

The Sensors Directorate of the Air Force Research Laboratory has expressed an interest in improved pose estimation in general and in Principe's methods in particular. As a result of the above literature review and analysis, several research needs were identified for investigation in this research.

Principe's and Agarwal's methods of independent, adaptive network-based pose estimation show potential for further extension and refinement. Principe makes the understated observation in (Principe, 1998c) that networks trained with the mutual information criterion and the mean squared error criterion have similar performance. This raises significant questions: What aspects of the data are the MI and MSE training methods using to discern pose? Since performance is equivalent, are they using the same characteristics of the image?

Application of these methods to HRR data is also of great interest. Can the MSE and MI methods be used for estimating azimuth of ground targets or air targets with HRR?

Finally, extension to more degrees of freedom is desirable. Can SAR or HRR pose estimation with Principe's methods be extended to provide both azimuth and elevation, similar to Agarwal's system? If so, will both the MI and MSE training criterion work? What advantage, if any, does the more complex MI criterion provide over MSE for any of these scenarios?

2.6 Summary

Current pose estimation methods applicable to radar ATR were reviewed and compared in terms of their dependence on classification and use of features versus raw images. Analysis identified as preferable those methods that exhibited independence from classification and used raw images to capture all relevant information. The pose estimation methods of Principe and Agarwal were selected for extension to HRR data and two degrees of freedom.

III. Methodology

This chapter describes the methodology used during this research, including the two network training criteria and the data sets. It also details the series of experiments that were conducted to progressively extend pose estimation to HRR data and multiple degrees of freedom.

3.1 Training Criteria

Principe states in (Principe, 1998c) that networks trained with the mutual information (MI) and mean-squared-error (MSE) criteria have equivalent performance. However, the MI technique is more complex than MSE training (order of n^2 versus order of n , where n is the number of training exemplars). Thus the MSE technique is preferred for the pose estimation scenarios Principe has tested.

However, it is unknown whether the performance of the two methods is also equivalent for proposed extensions to HRR and two degrees of freedom. An important objective of this research is to utilize both training methods for the extended cases and compare their performance. The details of both techniques are reviewed here, following a quick review of network notation.

3.1.1 Network Notation. The linear network for SAR azimuth estimation is illustrated in Figure 9. In general, the network has k input nodes, and $\mathbf{X}$ is the input vector of size $[k \times 1]$. Each of the k elements in $\mathbf{X}$ is one pixel in the SAR image or one range bin in the HRR signature. There are m output nodes, and $\mathbf{Y}$ is the output vector of size $[m \times 1]$. Also, $\mathbf{W}$ is the weight vector of size $[m \times k]$. A single weight w_{ij} is the strength of the connection to output node i from input node j . The output of the network is calculated as $\mathbf{Y} = \mathbf{W}\mathbf{X}$.

The network output for a set of input images is calculated in exactly the same way. Let $\mathbf{X}'$ be a set of n input vectors of size $[k \times n]$. Let $\mathbf{Y}'$ be the corresponding

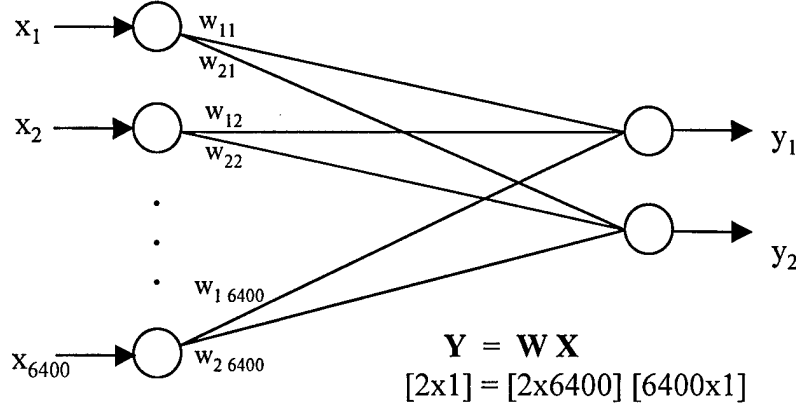


Figure 9 Linear network for SAR azimuth estimation

set of n output vectors of size $[m \times n]$. Then the set of network outputs is calculated as $\mathbf{Y}' = \mathbf{W} \mathbf{X}'$.

3.1.2 Mutual Information (MI) Training. The mutual information training technique is based on information theory. The reader is referred to (Principe, 1998a) for an in-depth development of the information theoretic principles used in Principe's algorithm. This section summarizes the concepts and explains specifically how they are applied to pose estimation.

3.1.2.1 Probability Distributions, Information, and Entropy. The amount of information conveyed by a random variable depends on its probability distribution function (pdf). If the value of a random variable is known with high probability, then the outcome of a particular event conveys little information. If the value of the variable is very uncertain, then the outcome of an event conveys much information. (See Figure 10.)

Entropy is the mathematical measure of the amount of uncertainty (and hence the amount of information) contained in a random variable's pdf. In other words, it measures how *spread out* the distribution is. Different formulas for entropy have

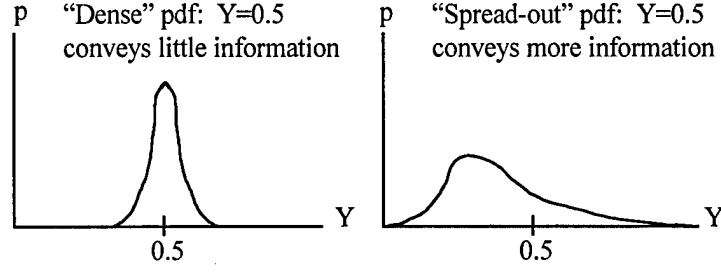


Figure 10 Information conveyed by probability distribution functions

been proposed; Principe uses Renyi's quadratic entropy (Principe, 1998a):

$$H(Y) = -\log \left(\int_{-\infty}^{\infty} f_Y(y)^2 dy \right)$$

To apply information theory to azimuth estimation, we first seek to find the probability distribution and corresponding entropy for a set of target exemplars which span the 0-360 degree azimuth range. Consider T to be the two-dimensional variable representing the *true* azimuth of the set of target exemplars, as shown in Figure 11. Each target image has a point on the unit circle with its angular position determined by azimuth. (Note that the points are not all evenly spaced around the circle, due to dropouts and variations in the SAR data collection.) The pdf of this variable can be estimated using the Parzen windows method (Bishop, 1995) with a Gaussian kernel. Figure 12 shows the form of the Gaussian probability estimate for a single target exemplar; Figure 13 shows the complete pdf estimate for a set of 52 exemplars from the MSTAR data set (DARPA, 1997). Since this pdf is the representation of the true aspect angles, it is used as a pdf *template* (i.e., the desired response of the system) for training.

The pdf and entropy associated with T for this set of training points $\{a_i\}$ is calculated as follows:

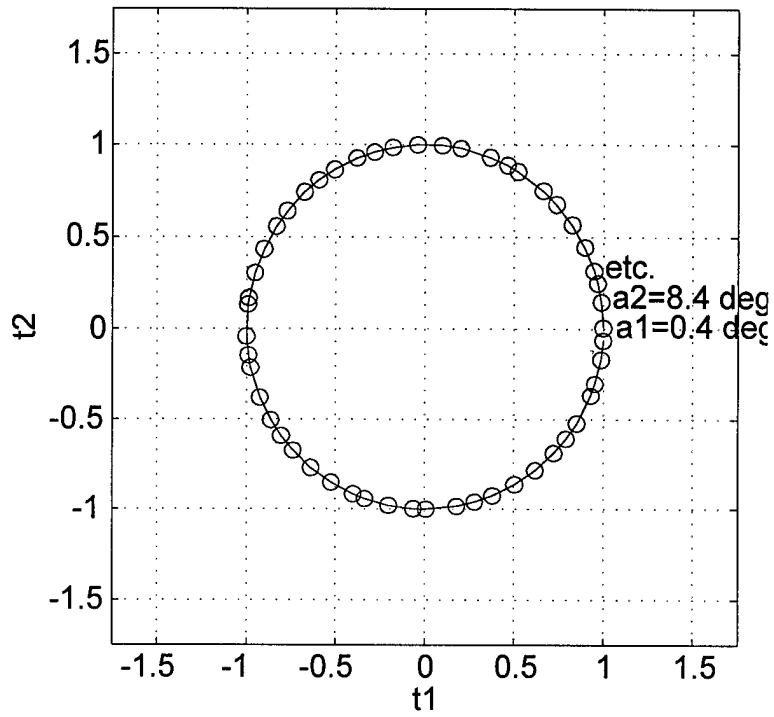


Figure 11 Actual azimuths of 52 target exemplars represented by placement on unit circle

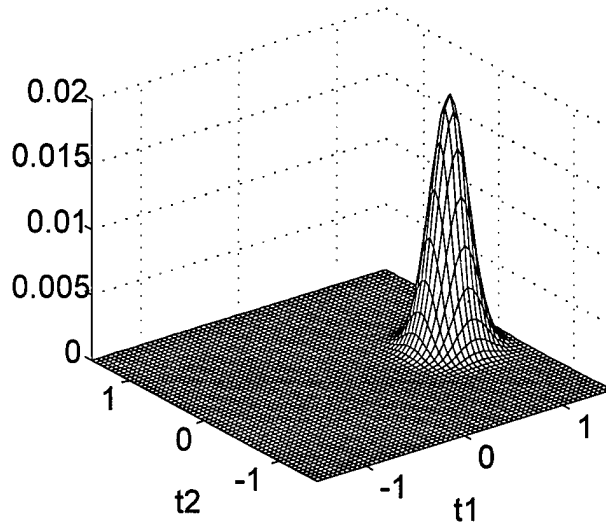


Figure 12 Gaussian probability estimate for a single exemplar

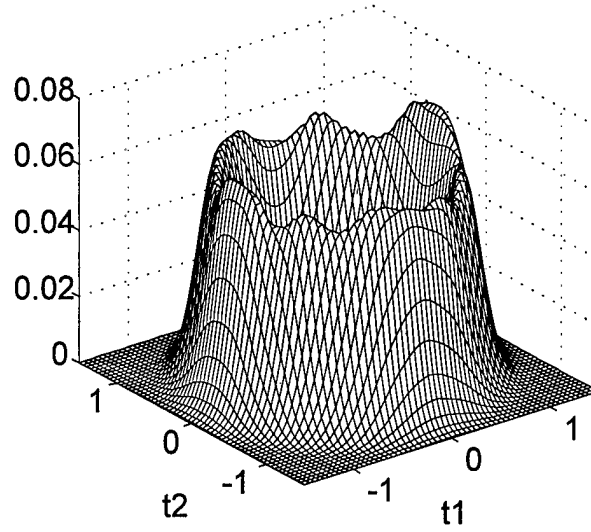


Figure 13 Template pdf representing actual target azimuths for 52 exemplars

$$\begin{aligned}
 pdf &= f_T(\{\mathbf{a}_i\}) = \frac{1}{N} \sum_{i=1}^N G(\mathbf{t} - \mathbf{a}_i, \sigma^2) \\
 Entropy &= H(\{\mathbf{a}_i\}) = -\log \left(\int_{-\infty}^{\infty} f_T(\mathbf{t})^2 d\mathbf{t} \right) = -\log V(\{\mathbf{a}_i\}) \\
 V(\{\mathbf{a}_i\}) &= \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N \int_{-\infty}^{\infty} G(\mathbf{t} - \mathbf{a}_i, \sigma^2) G(\mathbf{t} - \mathbf{a}_j, \sigma^2) d\mathbf{t} \\
 &= \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2)
 \end{aligned}$$

where $G(\boldsymbol{\omega}, \sigma^2)$ is the Gaussian kernel in two-dimensional space with covariance matrix equal to $\sigma^2 \mathbf{I}$:

$$G(\boldsymbol{\omega}, \sigma^2) = \frac{1}{2\pi\sigma^2} e^{-\frac{\boldsymbol{\omega}^T \boldsymbol{\omega}}{2\sigma^2}}.$$

Principe calls the quantity $V(\{\mathbf{a}_i\})$ an "information potential" (Principe, 1998a), making an analogy with particles in physical systems. The potential between two points is high when the points are close together and low when they are far apart.

3.1.2.2 Measuring Mutual Information. The mutual information (also called cross-entropy) between two variables is a measure of the similarity of their pdfs, i.e., the commonality of the information each one conveys. Mutual information is large if the shapes of the pdfs are very similar.

For pose estimation, consider the linear network shown in Figure 9, where Y is the output of the network and has the same dimensionality as T . We wish to measure the mutual information between the pdf of Y (the network output) and the pdf of the template variable T (representing the actual azimuth angles). To measure mutual information, Principe proposes a method based on the joint pdf of the two variables (Xu, 1998b). It is established that two variables Y and T are statistically independent only if their joint distribution is the product of their marginal distributions, i.e., $f_{YT}(\mathbf{y}, \mathbf{t}) = f_Y(\mathbf{y}) f_T(\mathbf{t})$. Using this fact, the Euclidean distance inequality $\|a - b\|^2 = \|a\|^2 + \|b\|^2 - 2a^T b \geq 0$ and the quadratic entropy are used to construct the following distance measure:

$$\begin{aligned} D(Y, T) &= \iint f_{YT}(\mathbf{y}, \mathbf{t})^2 d\mathbf{y} d\mathbf{t} + \iint f_Y(\mathbf{y})^2 f_T(\mathbf{t})^2 d\mathbf{y} d\mathbf{t} \\ &\quad - 2 \iint f_{YT}(\mathbf{y}, \mathbf{t}) f_Y(\mathbf{y}) f_T(\mathbf{t}) d\mathbf{y} d\mathbf{t} \\ D(Y, T) &= V_{joint} + V_{marginal} - 2V_{cross} \end{aligned}$$

This measures the distance between the joint pdf $f_{YT}(\mathbf{y}, \mathbf{t})$ and the product of the marginal pdfs $f_Y(\mathbf{y}) f_T(\mathbf{t})$. Thus, $D(Y, T) \geq 0$ and the distance is zero only when Y and T are statistically *independent*. Principe and Xu state in (Xu, 1998a) that this distance has also been experimentally verified as a valid measure of *dependence*. They call this measure "Euclidean Distance Quadratic Mutual Information (ED-

QMI)” (Xu, 1998b). To maximize mutual information (that is, to make two pdfs as dependent as possible), one maximizes the ED-QMI measure.

3.1.2.3 Adapting Network Weights with Mutual Information. The goal of Principe’s system is to maximize the mutual information between Y (network output) and the template variable T (representing the actual azimuth angles). In practical terms, this maximization makes the pdf of Y similar to the template pdf shown in Figure 13. Accomplishing this transformation requires adjusting the network weights.

Initially, the weights are set to small random values. Presenting the set of training images to the network then produces a set of output points $\{b_i\}$ randomly clustered around the origin, as shown in Figure 14, with the corresponding initial pdf for Y shown in Figure 15.

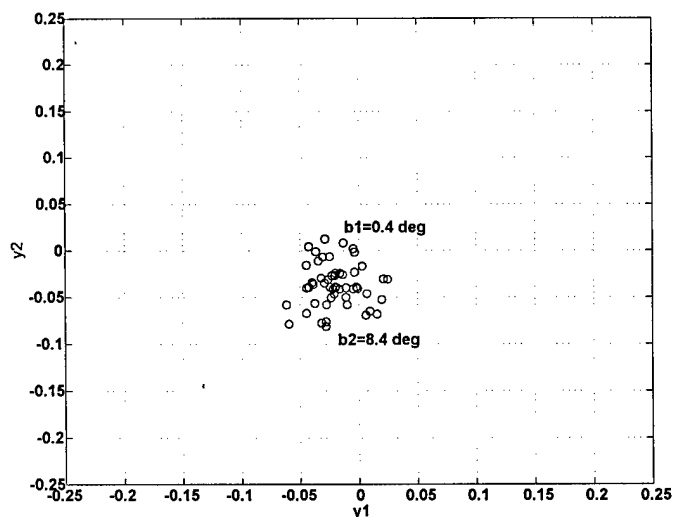


Figure 14 Network output generated by a set of exemplars using small initial random weights

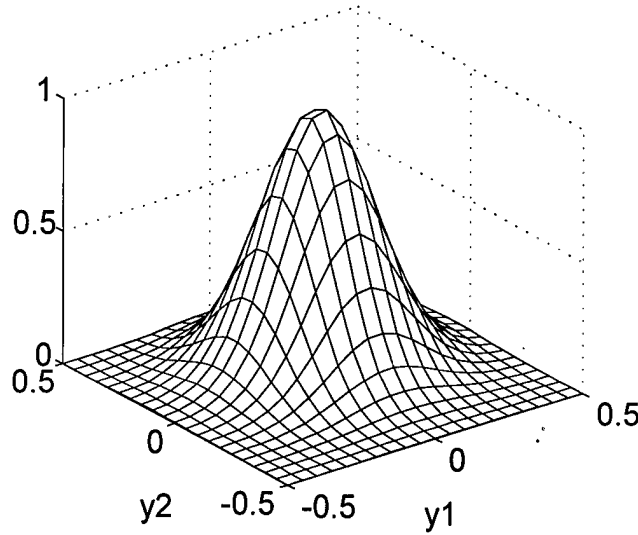


Figure 15 Pdf for network output using initial random weights

Mutual information is maximized by maximizing the distance $D(Y, T)$. This maximization has a direct parallel with traditional network training, where an error function must be minimized. In traditional methods the derivative of the error is used to adjust the weights, moving the network toward an error minimum. In the MI technique the derivative of the ED-QMI measure is used as an "information force" (Principe, 1998a) to adjust the weights, moving the network toward a mutual information maximum.

The information forces are derived from the ED-QMI distance measure as follows. The joint probability and marginal probabilities of Y and T are estimated as:

$$\begin{aligned}
 f_{YT}(\mathbf{y}, \mathbf{t}) &= \frac{1}{N} \sum_{i=1}^N G(\mathbf{y} - \mathbf{b}_i, \sigma^2) G(\mathbf{t} - \mathbf{a}_i, \sigma^2) \\
 f_Y(\mathbf{y}) &= \frac{1}{N} \sum_{i=1}^N G(\mathbf{y} - \mathbf{b}_i, \sigma^2) \\
 f_T(\mathbf{t}) &= \frac{1}{N} \sum_{i=1}^N G(\mathbf{t} - \mathbf{a}_i, \sigma^2).
 \end{aligned}$$

The terms of the distance measure are calculated as:

$$\begin{aligned}
V_{joint} &= \iint f_{YT}(\mathbf{y}, \mathbf{t})^2 d\mathbf{y} d\mathbf{t} \\
&= \iint \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{y} - \mathbf{b}_i, \sigma^2) G(\mathbf{y} - \mathbf{b}_j, \sigma^2) G(\mathbf{t} - \mathbf{a}_i, \sigma^2) G(\mathbf{t} - \mathbf{a}_j, \sigma^2) \right] d\mathbf{y} d\mathbf{t} \\
&= \frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{b}_i - \mathbf{b}_j, 2\sigma^2) G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2).
\end{aligned}$$

$$\begin{aligned}
V_{marginal} &= \iint f_Y(\mathbf{y})^2 f_T(\mathbf{t})^2 d\mathbf{y} d\mathbf{t} \\
&= \iint \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{y} - \mathbf{b}_i, \sigma^2) G(\mathbf{y} - \mathbf{b}_j, \sigma^2) \right] \\
&\quad \times \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{t} - \mathbf{a}_i, \sigma^2) G(\mathbf{t} - \mathbf{a}_j, \sigma^2) \right] d\mathbf{y} d\mathbf{t} \\
&= \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{b}_i - \mathbf{b}_j, 2\sigma^2) \right] \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right].
\end{aligned}$$

$$\begin{aligned}
V_{cross} &= \iint f_{YT}(\mathbf{y}, \mathbf{t}) f_Y(\mathbf{y}) f_T(\mathbf{t}) d\mathbf{y} d\mathbf{t} \\
&= \iint \left[\frac{1}{N} \sum_{i=1}^N G(\mathbf{y} - \mathbf{b}_i, \sigma^2) G(\mathbf{t} - \mathbf{a}_i, \sigma^2) \right] \\
&\quad \times \left[\frac{1}{N} \sum_{i=1}^N G(\mathbf{y} - \mathbf{b}_i, \sigma^2) \right] \left[\frac{1}{N} \sum_{i=1}^N G(\mathbf{t} - \mathbf{a}_i, \sigma^2) \right] d\mathbf{y} d\mathbf{t} \\
&= \frac{1}{N} \sum_{i=1}^N \left[\left(\frac{1}{N} \sum_{j=1}^N G(\mathbf{b}_i - \mathbf{b}_j, 2\sigma^2) \right) \left(\frac{1}{N} \sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right) \right].
\end{aligned}$$

Again using the analogy with physical systems, each of the terms is considered an information potential. The derivative of the potential is an "information force" (Principe, 1998a). The overall information force exerted on network output point $\mathbf{b}_p$ is:

$$\begin{aligned} Force &= \frac{\partial}{\partial \mathbf{b}_p} D(Y, T) \\ Force &= \frac{\partial}{\partial \mathbf{b}_p} V_{joint} + \frac{\partial}{\partial \mathbf{b}_p} V_{marginal} - 2 \frac{\partial}{\partial \mathbf{b}_p} V_{cross}. \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial \mathbf{b}_p} V_{joint} &= \frac{\partial}{\partial \mathbf{b}_p} \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{b}_i - \mathbf{b}_j, 2\sigma^2) G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right] \\ &= \frac{1}{N^2} \sum_{i=1}^N \{ [G(\mathbf{b}_p - \mathbf{b}_i, 2\sigma^2) G(\mathbf{a}_p - \mathbf{a}_i, 2\sigma^2) \\ &\quad + G(\mathbf{b}_i - \mathbf{b}_p, 2\sigma^2) G(\mathbf{a}_i - \mathbf{a}_p, 2\sigma^2)] \left(\frac{\mathbf{b}_i - \mathbf{b}_p}{4\sigma^2} \right) \} \end{aligned}$$

$$\begin{aligned} \frac{\partial}{\partial \mathbf{b}_p} V_{marginal} &= \frac{\partial}{\partial \mathbf{b}_p} \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{b}_i - \mathbf{b}_j, 2\sigma^2) \right] \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right] \\ &= \left[\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right] \\ &\quad \times \frac{1}{N^2} \sum_{i=1}^N \{ [G(\mathbf{b}_p - \mathbf{b}_i, 2\sigma^2) + G(\mathbf{b}_i - \mathbf{b}_p, 2\sigma^2)] \left(\frac{\mathbf{b}_i - \mathbf{b}_p}{4\sigma^2} \right) \} \end{aligned}$$

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{b}_p} V_{cross} &= \frac{\partial}{\partial \mathbf{b}_p} \frac{1}{N} \sum_{i=1}^N \left[\left(\frac{1}{N} \sum_{j=1}^N G(\mathbf{b}_i - \mathbf{b}_j, 2\sigma^2) \right) \left(\frac{1}{N} \sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right) \right] \\
&= \frac{1}{N^3} \sum_{i=1}^N \left\{ \left[\left(\sum_{j=1}^N G(\mathbf{a}_p - \mathbf{a}_j, 2\sigma^2) \right) G(\mathbf{b}_p - \mathbf{b}_i, 2\sigma^2) \right. \right. \\
&\quad \left. \left. + \left(\sum_{j=1}^N G(\mathbf{a}_i - \mathbf{a}_j, 2\sigma^2) \right) G(\mathbf{b}_i - \mathbf{b}_p, 2\sigma^2) \right] \left(\frac{\mathbf{b}_i - \mathbf{b}_p}{4\sigma^2} \right) \right\}
\end{aligned}$$

The force on each point is a $[m \times 1]$ vector, where m is the number of network outputs. Let $\mathbf{F}$ be the $[m \times n]$ matrix that gives the forces on the entire set of output points. The weights can then be adjusted as follows:

$$\begin{aligned}
\mathbf{W}_{new} &= \mathbf{W}_{old} + \alpha \mathbf{F} \mathbf{X}^T \\
[m \times k] &= [m \times k] + [m \times n][n \times k],
\end{aligned}$$

where α is the step size.

Figures 16 to 18 illustrate the network output space during MI training. The lines indicate the direction and relative strength of the information force on each point. As training proceeds, the points in the output space move to make the pdf of $\mathbf{Y}$ match that of the template. Note that the resulting circle is not centered on the origin because the system is tuning the *shape* of the pdf; the actual location of the circle is free to float in the output space during training.

3.1.2.4 Pose Estimation for Unknown Targets. Once the network has been trained, an unknown target may be presented to the network. This presentation generates a single point somewhere in the output space. The azimuth is estimated by finding the maximum of the joint probability density $f_{YT}(\mathbf{y}, \mathbf{t})$ of the single test point $\{\mathbf{b}\}$ with the set of training points $\{\mathbf{a}_i\}$, which can be calculated as previously shown.

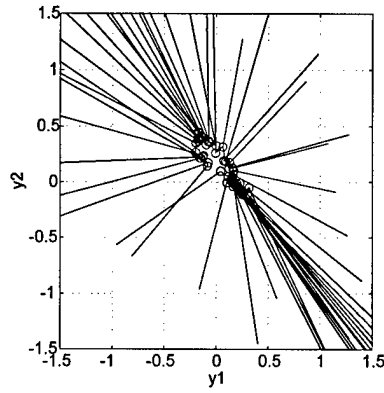


Figure 16 MI training (iteration 5)

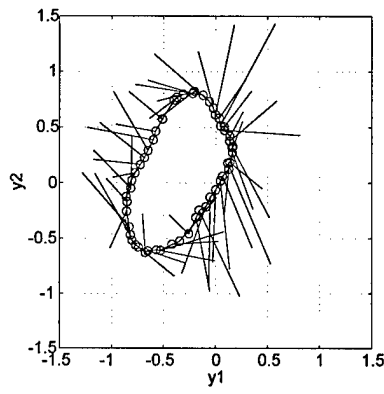


Figure 17 MI training (iteration 15)

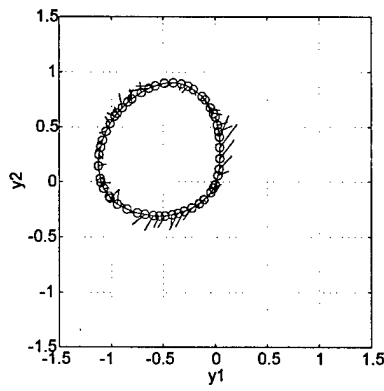


Figure 18 MI training (iteration 25)

3.1.3 Mean-Squared-Error (MSE) Training. The mean-squared-error learning rule (also known as the Widrow-Hoff algorithm) has been used for many years for adaptive training of linear networks (Mathworks, 1998). It is straightforward and avoids the pdf estimation procedures required for the mutual information technique. It is based simply on the squared error between the network outputs and the desired outputs:

$$\begin{aligned} \text{Error} &= \mathbf{e} = \mathbf{a}_i - \mathbf{b}_i \\ \text{MSE} &= \frac{1}{N} \sum_{i=1}^N (\mathbf{a}_i - \mathbf{b}_i)^2, \end{aligned}$$

where $\mathbf{a}_i$ is the desired network output for exemplar number i and $\mathbf{b}_i$ is the two-dimensional network output for the same exemplar.

Because the mean-squared-error is quadratic, gradient descent can be used to adjust network weights to find the minimum error. Taking the partial derivative of the squared error with respect to the weights yields:

$$\begin{aligned} \frac{\partial \mathbf{e}^2}{\partial w_{ij}} &= 2\mathbf{e} \frac{\partial \mathbf{e}}{\partial w_{ij}} \\ &= 2\mathbf{e} \frac{\partial}{\partial w_{ij}} [\mathbf{a}_i - \mathbf{b}_i] \\ &= 2\mathbf{e} \frac{\partial}{\partial w_{ij}} [\mathbf{a}_i - \mathbf{W}\mathbf{X}] \\ &= 2\mathbf{e} (-x_j). \end{aligned}$$

The weights may thus be updated in matrix form as follows:

$$\begin{aligned} \mathbf{W}_{new} &= \mathbf{W}_{old} + \alpha \left[\frac{\partial \mathbf{e}^2}{\partial w_{ij}} \right] \\ &= \mathbf{W}_{old} + \alpha [2\mathbf{e}\mathbf{X}^T] \\ &= \mathbf{W}_{old} + 2\alpha [\mathbf{a}_i - \mathbf{W}_{old}\mathbf{X}] \mathbf{X}^T, \end{aligned}$$

where α is the step size.

An illustration of the training process using the MSE technique is shown in Figures 19 to 21. The network outputs for the set of training exemplars are initially randomly clustered around the origin. The error for each exemplar is shown by a line from the current network output point to the desired location on the unit circle for that exemplar.

3.2 Data Sets Used

3.2.1 SAR. Training and testing for SAR images was performed with the MSTAR public data sets. MSTAR (Public) Targets (DARPA, 1997) included three targets at 15 and 17 degrees elevation. The MSTAR/IU (Public) Mixed Targets data set (DARPA, 1998a) consisted of seven different targets, three of which included elevation coverage at 15, 17, 30, and 45 degrees. Both sets had complete 360 degree azimuth coverage.

Preprocessing was performed by taking the logarithm of the SAR magnitude for each target ($\log_{10}[\text{magnitude}]$) and scaling so that all pixels were in the range 0-255. The image was then clipped to 80×80 pixels to include the target and radar shadow. The 6400-element vector was then used as the input to the pose estimation network.

3.2.2 HRR Ground Targets. HRR signatures were obtained for the same ground targets as the MSTAR data sets. These signatures (DARPA, 1998b) were provided by the Air Force Research Laboratory through the Target Recognition Using Mathematical and Physical Exploitation of Target Signatures (TRUMPETS) program. The signatures were extracted directly from the MSTAR SAR data sets; one HRR signature was created from each SAR image. The data included complete 360 degrees azimuth coverage at 15 and 17 degrees elevation.

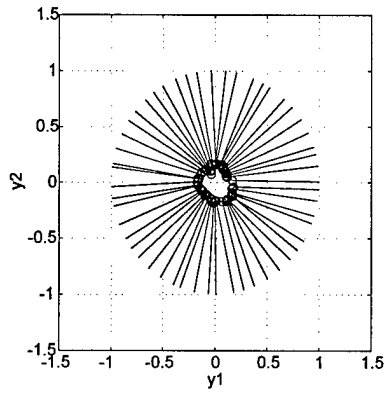


Figure 19 MSE training (iteration 100)

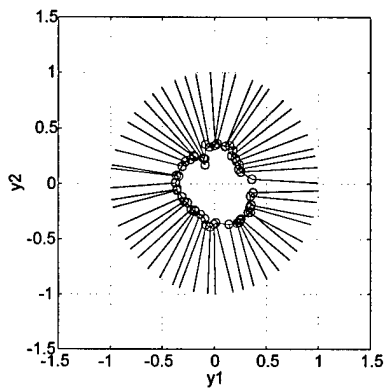


Figure 20 MSE training (iteration 200)

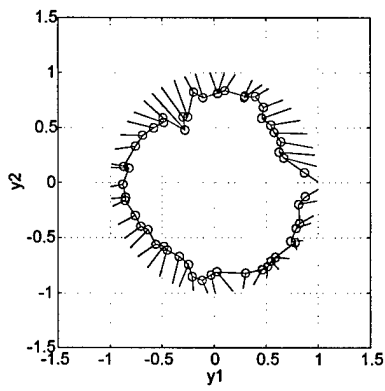


Figure 21 MSE training (iteration 500)

AFRL performed preprocessing when creating the TRUMPETS data, including zeroing clutter and normalizing magnitudes. The only additional processing done was to clip the signatures to include only the center 90 range bins. This 90-element vector was then used as the input to the pose estimation network.

3.2.3 HRR Air Targets. Because of data classification levels, only a limited amount of HRR aircraft data was obtained from AFRL. The data set included both measured and simulated HRR signatures for six targets over a 30 degree azimuth range and a 35 degree elevation range. The actual class identity and pose of the data were sanitized; aircraft classes were simply labeled *A* through *F*, and the pose was labeled 60-90 degrees azimuth and 0-35 degrees elevation.

Preprocessing was performed as follows. The magnitude of the complex-valued HRR signatures was taken: $magnitude = \sqrt{\text{real}^2 + \text{complex}^2}$. The signature magnitudes were scaled to the range 0-255 and clipped to 150 range bins. Due to the complex and highly variable nature of aircraft signatures, the data was also normalized and smoothed (to help prevent slight differences in signature alignment from causing large changes in network output). The signatures were smoothed by averaging the magnitude of each range bin with its two nearest neighbors: $magnitude$ of range bin $i = m_i = \frac{1}{3}(m_{i-1} + m_i + m_{i+1})$. Normalization was performed by making the area under each signature equal. This processed 150-element vector was then used as the input to the pose estimation network.

3.3 Description of Experiments

The following series of experiments was performed in order to progressively investigate and extend the pose estimation techniques described above.

3.3.1 Validation of Principe's Results. The first requirement was to validate the results obtained by Principe for azimuth prediction of SAR images. Principe and Xu implemented the algorithm using PV-Wave software. Because the Air Force

Research Laboratory and the Air Force Institute of Technology both use MATLAB as their standard computation system, the PV-Wave code was translated to MATLAB and extensively commented.

Experiments were performed for 180-degree and 360-degree azimuth estimation using the three targets in (DARPA, 1997) in order to validate Principe's reported results (Principe, 1998a).

3.3.2 Image Understanding for SAR Azimuth Estimation. According to (Principe, 1998c), both the MI and MSE training techniques produced good results for azimuth estimation. However, it is desirable to understand what image characteristics the network is using to recover pose for different training and testing scenarios. These image understanding concepts were explored through weight visualization. The weights for each output node were scaled to the range 0-255, where $0 = w_{\min}$ and $255 = w_{\max}$, and displayed as an 80×80 pixel image. Comparing the weight visualization with the actual target image indicates what weights are being applied to what portions of the target image.

3.3.3 Application to HRR Azimuth Estimation. Principe applied network-based pose estimation only to two-dimensional SAR images. A major objective of this research is to discover whether one-dimensional HRR profiles contain enough information to estimate pose using the same techniques. In order to properly compare HRR versus SAR, HRR tests were performed on the same MSTAR targets as Principe's SAR tests (using TRUMPETS data).

The MATLAB code was rewritten to read and preprocess the HRR data as described in Section 3.2.2. The number of input nodes to the network was changed from 6400 to 90—the size of the HRR signature vector. Experiments were conducted to test the accuracy of azimuth prediction and to check whether HRR distinguished symmetrical views better or worse than SAR.

3.3.4 *Application to Two Degrees of Freedom in SAR.* In order to estimate both azimuth and elevation, the pose angle representation must be modified. The unit circle is no longer adequate to represent two degrees of freedom. Two possible representation surfaces were examined: concentric rings and cylinders. (A spherical surface was considered but not implemented, because it was not sufficiently different from the cylindrical surface for the elevation range considered here.)

The concentric ring representation minimizes changes to the network structure, and thus to the code. The network still has two output nodes, and the exemplars still produce circles in two-dimensional output space. However, the exemplar sets for different elevations are represented by circles of different diameters (Figure 22).

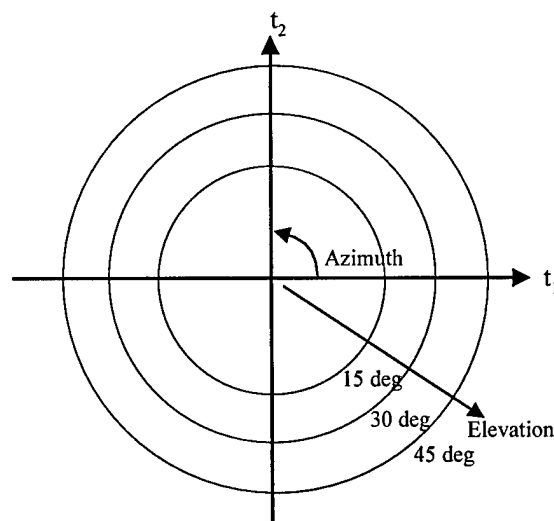


Figure 22 Concentric circle representation for 2-DOF pose

The cylinder representation requires adding a third output node to the network. The azimuth is represented by a circle as before. The elevation is represented by the height on the surface of the cylinder (Figure 23). The cylinder representation required more significant modifications to the code, especially for the MI technique. All pdf and information force calculations had to be extended to use three-dimensional vectors and Gaussians in three-dimensional space.

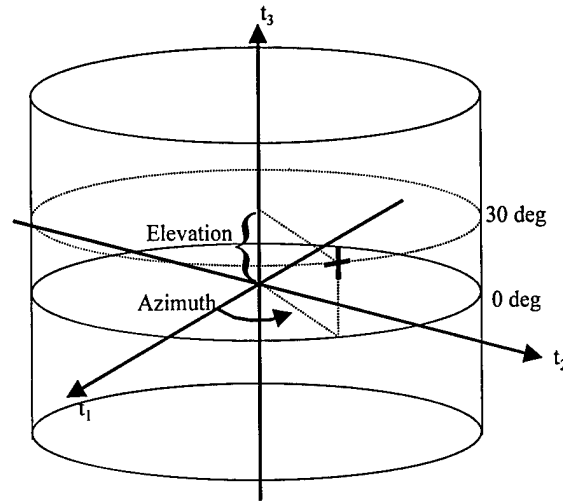


Figure 23 Cylinder representation for 2-DOF pose

All tests were performed using the MSTAR/IU Mixed Targets data set, since it provided images at multiple elevations (15, 30, and 45 degrees were used for training and testing). The accuracy of the pose representations and the training criteria were compared, and weight visualization was also performed to determine what image characteristics were being used to predict elevation.

3.3.5 Application to Two Degrees of Freedom in HRR. The TRUMPETS data set provided HRR signatures only at 15 and 17 degrees elevation for MSTAR ground targets. This small range was not considered adequate to test the ability to discriminate between elevations. Thus only the limited air target data set was used to test two-DOF pose estimation for HRR. Experiments were conducted to train and test for a single aircraft and to train and test on multiple aircraft classes.

Only the mean-squared-error algorithm was used for the HRR two-DOF experiments, for reasons that are discussed in Chapter 4. Much of the code from previous experiments was used directly, with most modifications being made in the data management and preprocessing routines.

3.4 Equipment and Software

All experiments were run on a Sun Ultra 2 workstation using MATLAB version 5.2 with Neural Networks Toolbox version 3.0.

Space constraints prevent the inclusion in this thesis of all MATLAB code generated. However, the code for two degrees of freedom in SAR is included in Appendix A as an example of how the algorithms are implemented.

3.5 Summary

This chapter discussed in detail the methodology used for pose estimation, with emphasis on comparing the two training criteria. Concepts and equations were developed for the mutual information technique, with its information forces used for network adaptation. The traditional mean-squared-error (Widrow-Hoff) algorithm was also reviewed. Descriptions of network configurations, data sets, and experiments were also provided.

IV. Results

This chapter presents results and analysis of the pose estimation experiments.

4.1 Validation of Principe's Results

Several experiments were performed to validate the MATLAB version of the algorithm and to confirm the repeatability of Principe's results given in his report (Principe, 1998a). All validation was performed on the three targets in the MSTAR (Public) Targets dataset (DARPA, 1997).

4.1.1 180 Degree Azimuth Estimation. The system was trained on the BMP-2/sn-c21 vehicle using the mutual information training criterion and tested for the 180 degree azimuth range. One variation from Principe's scenario was used in an attempt to improve the estimate accuracy, as follows. Principe trained only on 0-180 degree azimuths but tested on 0-360 degrees. To improve generalization over the entire testing range, this system was trained on 0-360 degrees *mod 180*. Thus the system provided an estimate in the range 0-180 but also captured the similarities between symmetrical views. A comparison of testing results is shown in Table 1.

It can be seen from Table 1 that the standard deviation is slightly higher than Principe's, but the error mean for some targets is lowered by almost two degrees.

Class/type	# test images	Error mean (deg)	Principe error mean	Error std dev (deg)	Principe error std dev
BMP2/sn-c21	131	2.26	3.45	1.67	1.61
BMP2/sn-9563	233	3.24	4.99	2.47	1.97
BMP2/sn-9566	232	3.26	4.99	2.81	2.58
T72/sn-s7	228	7.13	6.98	5.18	2.28
T72/sn-182	232	7.35	not reported	6.09	not reported
T72/sn-812	231	7.03	not reported	4.81	not reported
BTR70/sn-c71	233	3.50	not reported	2.72	not reported

Table 1 Validation testing for 0-180 degree training

Class/type	# test images	Error mean (deg)	Principe error mean	Error std dev (deg)	Principe error std dev
BMP2/sn-c21	131	8.32	12.4	7.78	4.53
BMP2/sn-9563	233	16.44	17.56	29.35	5.47
BMP2/sn-9566	232	23.47	26.15	29.25	5.58
T72/sn-s7	228	66.13	66.16	53.26	7.33
BTR70/sn-c71	233	64.74	not reported	45.47	not reported

Table 2 Validation testing for 0-360 degree training

The output space is illustrated for the best and worst test cases in Figures 24 and 25.

The system was also trained and tested using the MSE training criterion; error means were equivalent to those of MI training to within 0.5 degrees.

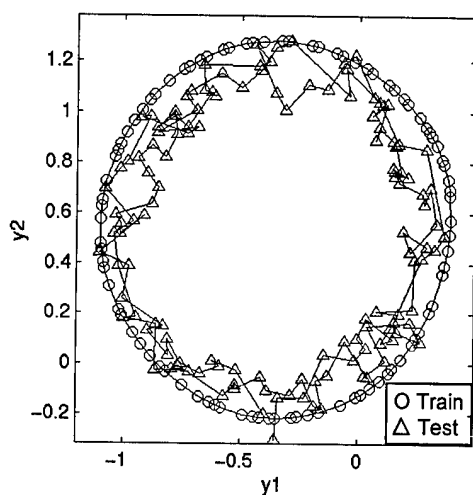
4.1.2 360 Degree Azimuth Estimation. The system was again trained on the BMP-2/sn-c21, this time for the 360 degree azimuth range. When using the mutual information training criterion, the system was unable to converge on a solution. Thus Principe's mutual information results were not confirmed for 360 degree training.

However, the system *was* able to converge when using the MSE criterion. A comparison of MSE results obtained here and Principe's MI results is shown in Table 2.

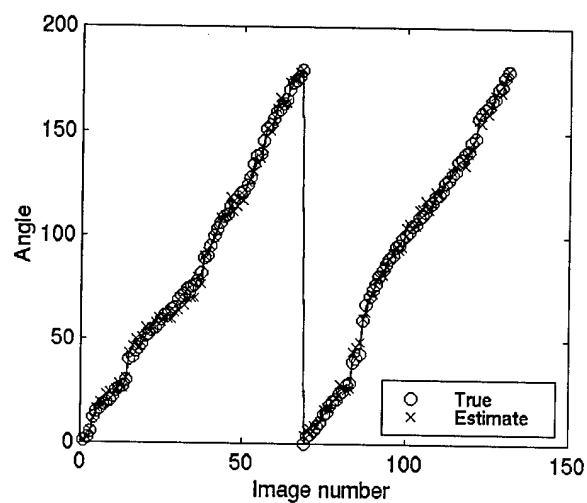
It is easily confirmed from Table 2 that 360 degree estimation does not generalize well to other vehicles or even to variations within the same vehicle class. The output space is illustrated for the best and worst test cases in Figures 26 and 27.

4.2 Image Understanding for SAR Azimuth Estimation

An important goal was to discover what image characteristics the network uses to determine pose. Weight visualization was used to analyze the differences between 180 and 360 degree azimuth estimation and to compare the two training criteria.

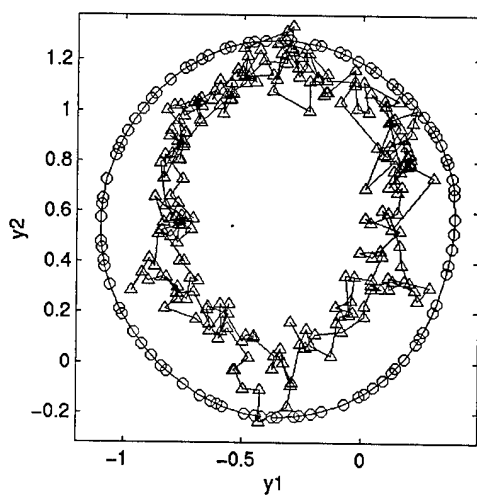


(a) Network output space

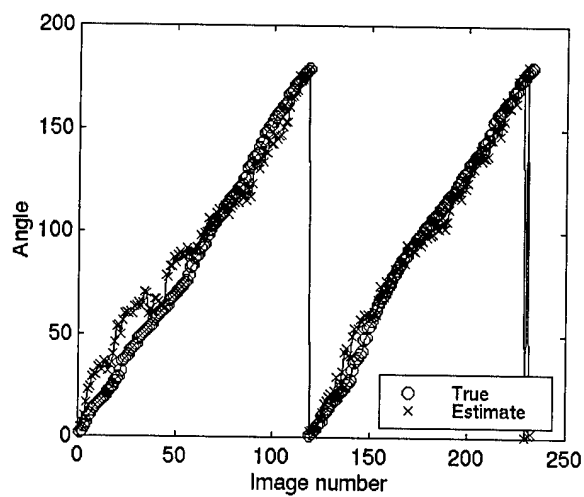


(b) True vs. estimated azimuth

Figure 24 SAR 180 degree results: trained on BMP2/sn-c21, tested on BMP2/sn-c21

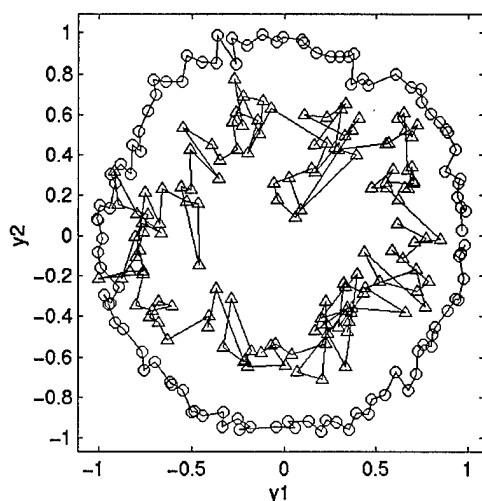


(a) Network output space

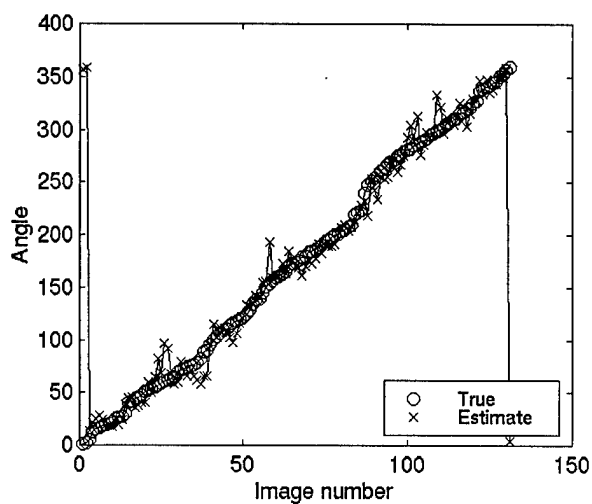


(b) True vs. estimated azimuth

Figure 25 SAR 180 degree results: trained on BMP2/sn-c21, tested on T72/sn-132

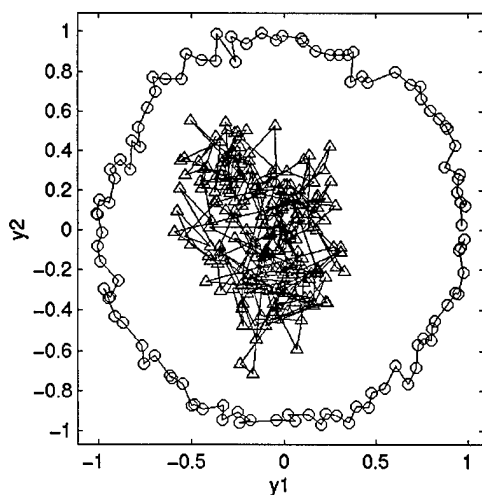


(a) Network output space

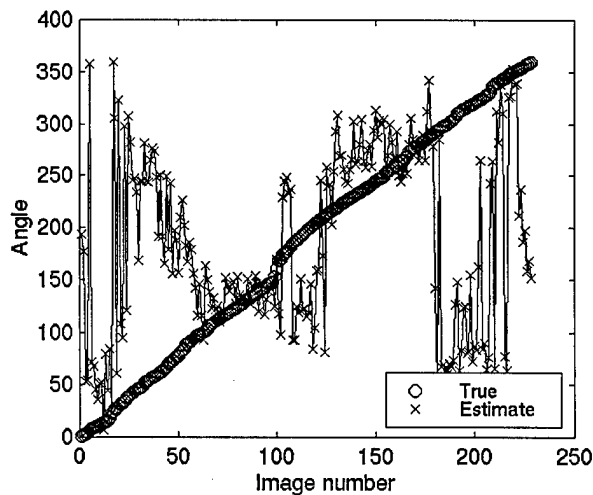


(b) True vs. estimated azimuth

Figure 26 SAR 360 degree results: trained on BMP2/sn-c21, tested on BMP2/sn-c21



(a) Network output space



(b) True vs. estimated azimuth

Figure 27 SAR 360 degree results: trained on BMP2/sn-c21, tested on T72/sn-s7

4.2.1 Weight Characteristics for 180 Degree Estimation. Some sample BMP-2 images are shown in Figure 28. Figures 29 and 30 show the weight visualization for the 180 degree estimation networks. White areas indicate the largest positive weights and black areas indicate the largest negative weights. Middle gray represents weights of approximately zero.

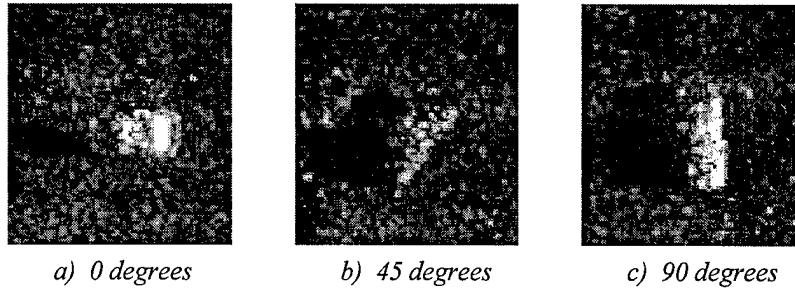


Figure 28 Sample BMP2 images at varying azimuth

From these weight visualizations it is apparent that the network is performing edge analysis of the vehicle image. The significant weights are located in bands around the vehicle edges, with positive or negative areas depending on the angular position. The network seems to obtain its estimate from the rectangularity of the target. The size of the bands indicates the network is tolerant of slightly off-center target positioning.

Another important result is obtained by comparing the network weights for MI and MSE training. The two different training criteria produce network weights with the same general appearance; thus it can be concluded that they use the same general image characteristics for azimuth estimation.

4.2.2 Weight Characteristics for 360 Degree Estimation. Figure 31 shows the weight visualization for the 360 degree estimation network. This figure illustrates the difference in image characteristics required to distinguish symmetrical views. While the 180 degree network used general target shape characteristics, the 360 degree network uses small, specific regions of the target. These small black and white

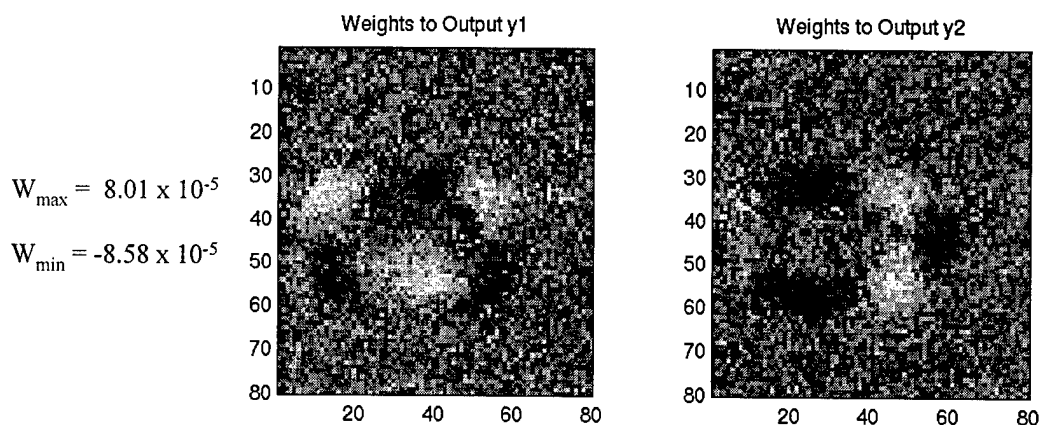


Figure 29 Network weights: MI training for 0-180 degrees on BMP2/sn-c21

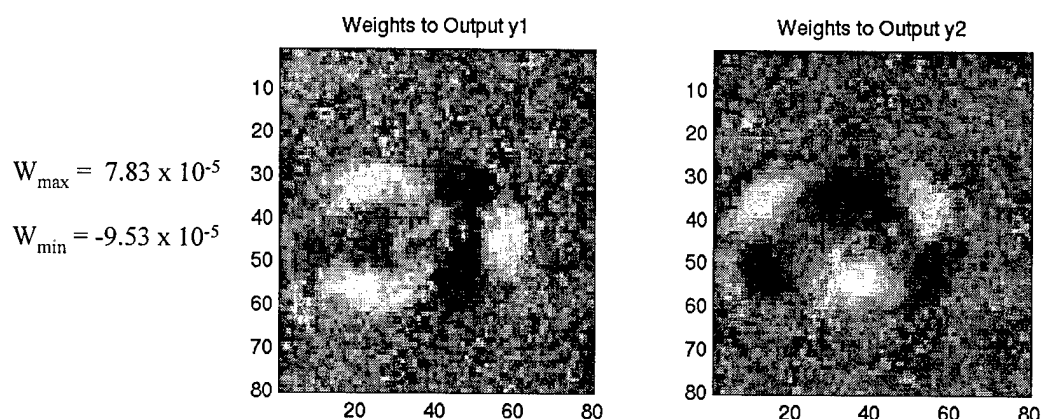


Figure 30 Network weights: MSE training for 0-180 degrees on BMP2/sn-c21

regions emphasize the parts of the BMP-2 that look different at angles separated by 180 degrees.

The above results explain why 360 degree estimation is so difficult. First, a smaller area of interest will produce greater errors for target variations due to vehicle configuration, clutter, and off-center positioning. In addition, the parts of a target useful for distinguishing symmetrical views may vary from target to target. For example, Figure 32 shows the weights for a network trained for 360 degree estimation using a T-72. For the T-72, the locations of regions with large weights are different from those of the BMP-2.

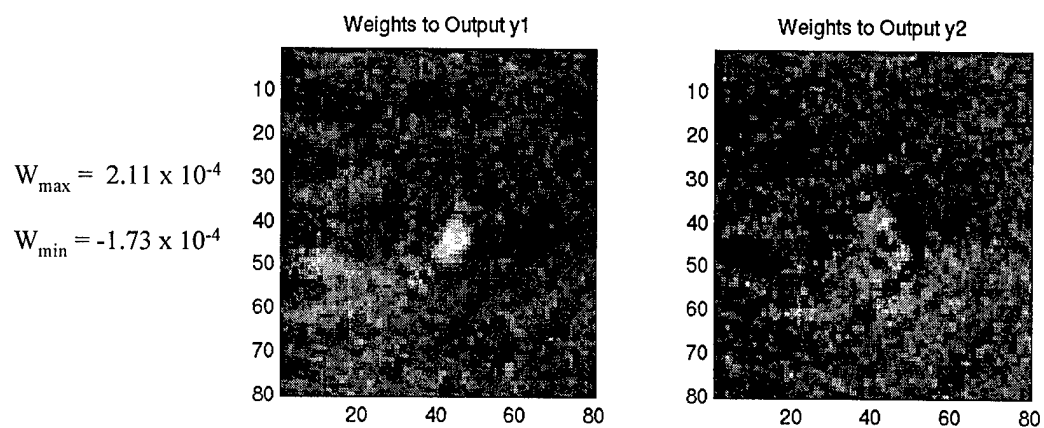


Figure 31 Network weights: MSE training for 0-360 degrees on BMP2/sn-c21

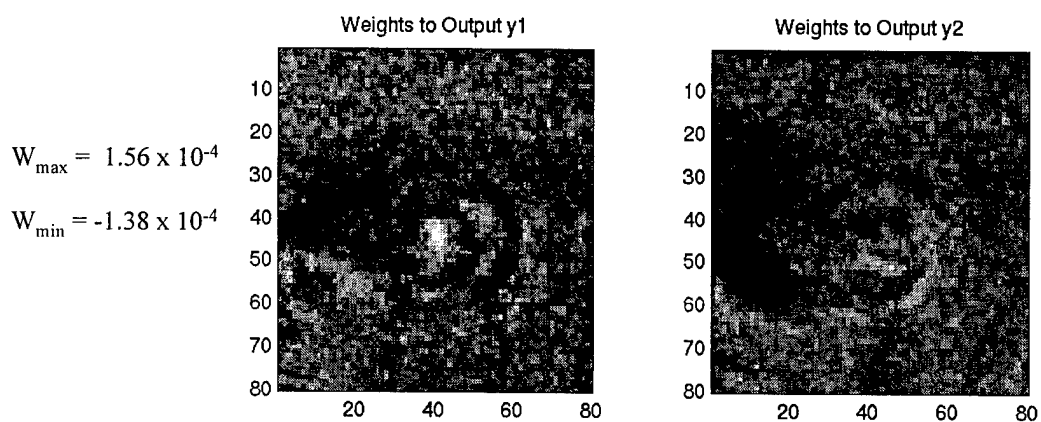


Figure 32 Network weights: MSE training for 0-360 degrees on T72/sn-s7

It can thus be concluded that it is not possible to form a generalized network that will produce accurate 360 degree estimates across vehicle classes. It is more productive to train a network to estimate 180 degree azimuth in an accurate and robust way, and then search both symmetrical views during the recognition process.

4.3 Application to HRR Azimuth Estimation

The next major goal was to test the feasibility of HRR pose estimation. The network was trained and tested with HRR signatures of MSTAR ground targets using the TRUMPETS data set (DARPA, 1998b). The 17 degree signatures were used as the training set, and the 15 degree signatures were used as the test set.

4.3.1 180 Degree Azimuth Estimation. The network was trained for the 0-180 degree range on the BMP-2/sn-c21 data. Training progressed more slowly than SAR training; both training criteria required an order of magnitude more iterations than SAR to converge to a solution. Results for network testing are shown in Figure 33 and reveal an important result: while SAR images produced a 180 degree output symmetry, the HRR signatures produced additional symmetries about the 90 degree azimuth angle (i.e., the network confused signatures at $azimuth = \theta$ with signatures at $azimuth = 180 - \theta$).

This result can be explained in terms of the range profile of the vehicle. Figure 34 shows that (for a rectangular vehicle) the summing of signal returns in each range bin can produce this symmetry along the radar line of sight. Thus the 180 degree and 90 degree symmetries together create four angles that produce very similar HRR profiles, in terms of both width and shape.

4.3.2 Design Change for 90 Degree Azimuth Estimation. Because the trained system could not distinguish symmetries even within the same class it was trained on, the network design was modified to perform 0-90 degree estimation. Since the angle range is no longer periodic (i.e., 0 and 90 degree signatures are not

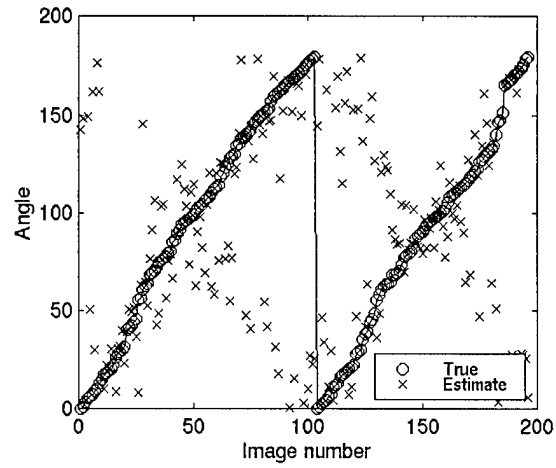


Figure 33 HRR 180 degree results: trained on BMP2/sn-c21, tested on BMP2/sn-c21

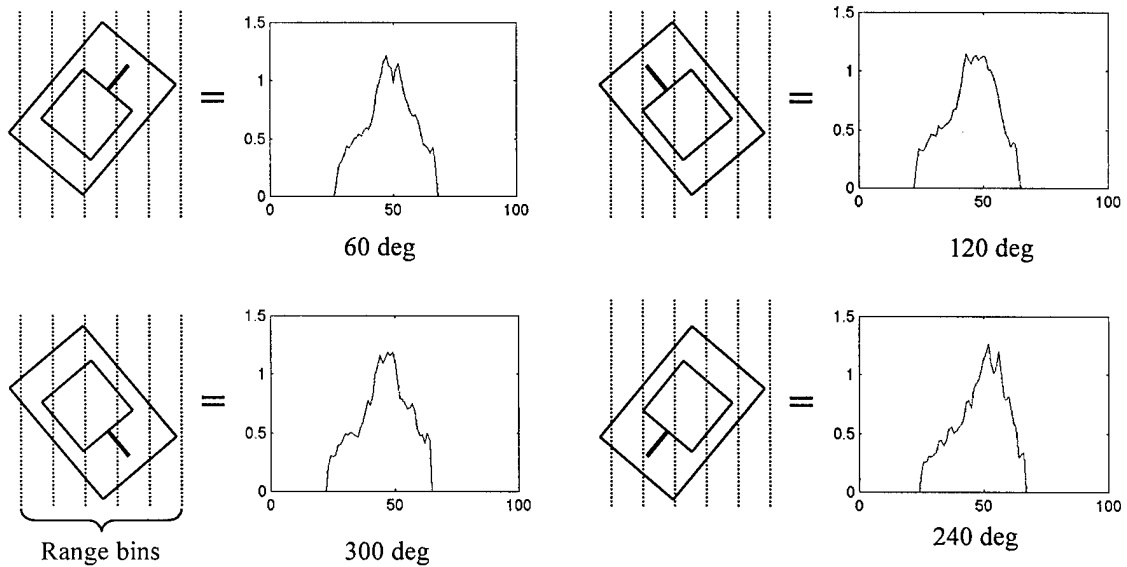


Figure 34 Four similar HRR profiles for BMP-2 caused by vehicle symmetry

equivalent), the two output nodes were reduced to one output node. The actual azimuths of the training set were modified as shown below to produce four sets of signatures in the 0-90 degree range.

Actual azimuth	Modified azimuth
$\theta = 0...90 \text{ deg}$	$\theta' = \theta$
$\theta = 90...180 \text{ deg}$	$\theta' = 180 - \theta$
$\theta = 180...270 \text{ deg}$	$\theta' = \theta - 180$
$\theta = 270...360 \text{ deg}$	$\theta' = 360 - \theta$

All four sets of signatures were used for training to give the system maximum ability to generalize.

Training and testing were performed using both the mutual information and the mean-squared-error methods. As was not the case for SAR pose estimation, the MSE method performed significantly better than the MI method; MSE consistently provided a mean error 3 degrees smaller than MI. In addition, a multi-layer perceptron architecture with three hidden nodes (as described in Chapter 2) provided slightly better performance than a linear network. Test results for an MLP network trained with the mean-squared-error criterion are shown in Table 3. When trained on a single vehicle (BMP-2) the network generalized well for some vehicles but not for others.

In order to attempt better generalization, the network was trained again with the signatures from five different vehicles (BMP-2, T-72, BTR-70, BRDM-2, ZSU-23/4). Test results are again shown in Table 3; accuracy improved in most cases. True versus estimated azimuth for the best and worst cases (except for the D7) are illustrated in Figures 35 and 36. The D7 bulldozer had unusually high errors, probably because its shape and size are much different from the other targets.

Class/type	# test signatures	One	Vehicle	Five	Vehicles
		Error mean (deg)	Error std dev (deg)	Error mean (deg)	Error std dev (deg)
BMP2/sn-c21	196	6.88	6.21	8.89	7.52
BMP2/sn-9563	195	8.19	7.88	8.67	7.86
BMP2/sn-9566	196	6.35	6.26	8.52	6.55
T72/sn-s7	191	12.25	10.05	6.62	5.41
BTR70/sn-c71	196	11.28	10.46	8.50	5.95
2S1	274	10.73	9.02	9.07	7.25
BRDM2	274	17.20	12.11	9.20	7.53
BTR60	195	10.13	8.43	10.31	8.79
D7	103	24.37	12.42	22.63	12.94
T62	273	12.23	10.56	8.50	6.70
ZIL131	274	11.27	8.33	11.10	7.84
ZSU23/4	274	13.52	10.98	14.73	10.09

Table 3 HRR testing for MLP network trained with mean-squared-error

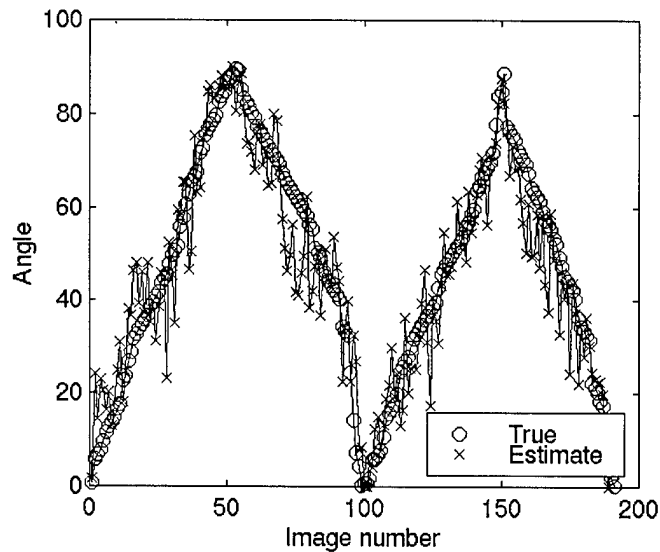


Figure 35 HRR 90 degree results: trained on 5 vehicles, tested on T72/sn-s7

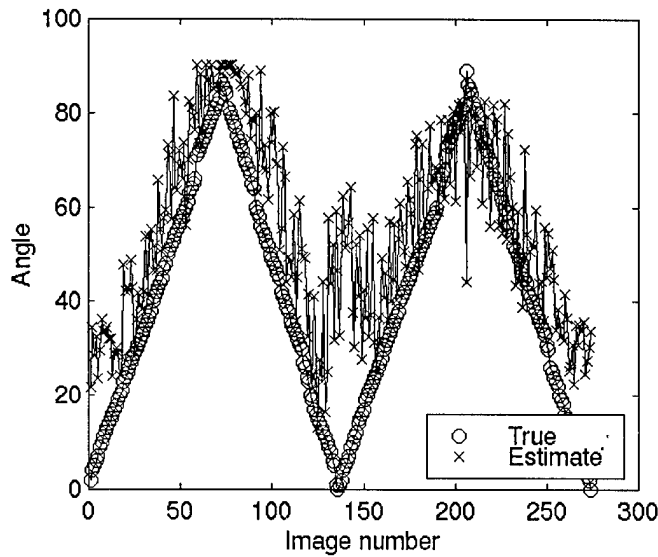


Figure 36 HRR 90 degree results: trained on 5 vehicles, tested on ZSU23/4

4.3.3 Performance Analysis. Overall, HRR pose estimation of ground targets is not as accurate as estimation using SAR. The error mean is generally higher by several degrees. Also, the additional symmetries require that once the 0-90 degree estimate is made, all four possible symmetrical poses must be searched during the target recognition process.

Several significant features of the HRR results merit closer analysis. First, the poor performance of the mutual information method was examined. It was observed that the high error means were caused by the estimates being artificially skewed toward the center of the range as shown in Figure 37. The probable cause is the form of the template pdf. As shown in Figure 38, the template is not periodic as in the SAR system, and the sum of Gaussians used for Parzen window probability density estimation causes the pdf to trail off at 0 and 90 degrees. When the joint pdf of the template and the test point is calculated, the center peak of the template causes the estimate to be skewed toward the center of the range.

Like the MI network, the MSE-trained network also had large errors near 0 degrees (but not near 90 degrees). These errors can be seen in Figures 35 and

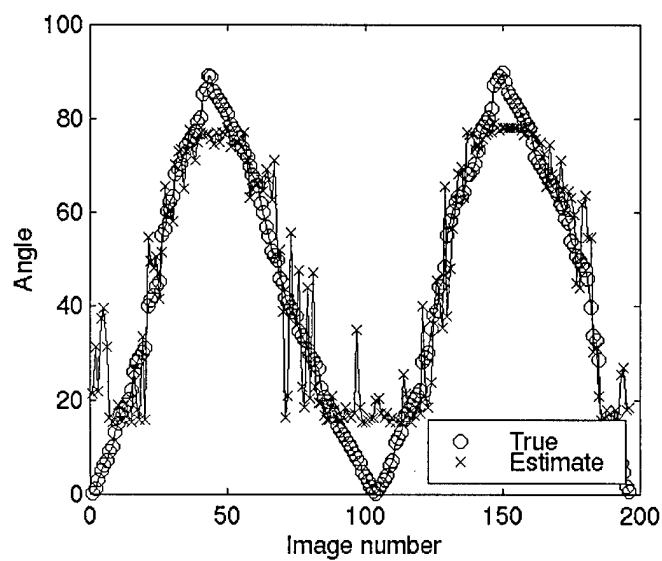


Figure 37 HRR 90 degree results: Mutual information training/estimation for BMP2

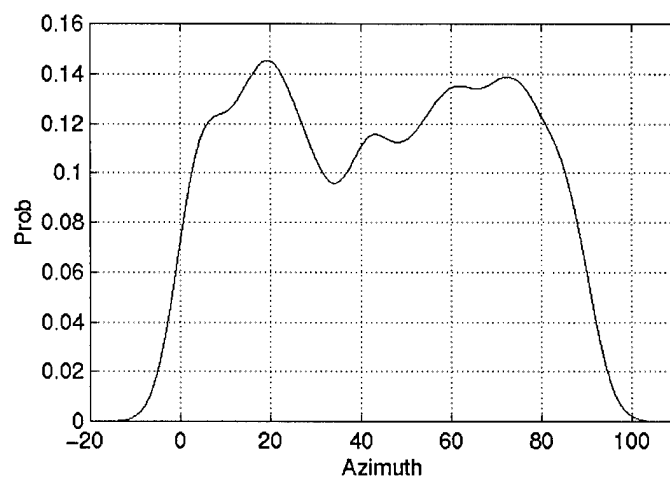


Figure 38 Template pdf for mutual information training on BMP2

36. The errors could not be attributed to the template pdf problem, since the MSE network provides a direct azimuth estimate without using the pdf. Insight can be gained by observing the network output for the BMP-2 *training* set in Figure 39. It can be seen that the network experienced problems resolving differences even in the training set for the 0-30 degree range. The problem in this region can be traced to the following apparent cause: the network is using signature width as a major factor in determining pose. Figure 40 shows that the length of a rectangular vehicle (and thus the width of its range profile) increases and then decreases again as the vehicle turns relative to the viewer. This effect was verified for the BMP-2 and is shown in Figure 40. The width of the signature is the same at more than one azimuth; thus, the network has difficulty predicting the correct azimuth for profiles of equal width.

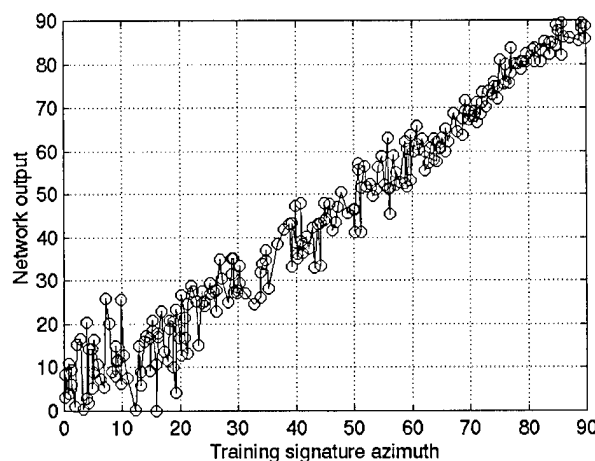


Figure 39 Network outputs for BMP-2 training signatures

In conclusion, HRR pose estimation of ground vehicles was shown to be possible but not as accurate as SAR. HRR pose estimation has only the advantage of being usable for fast-moving targets. The problems of symmetry and discerning between signatures of equal width need further study.

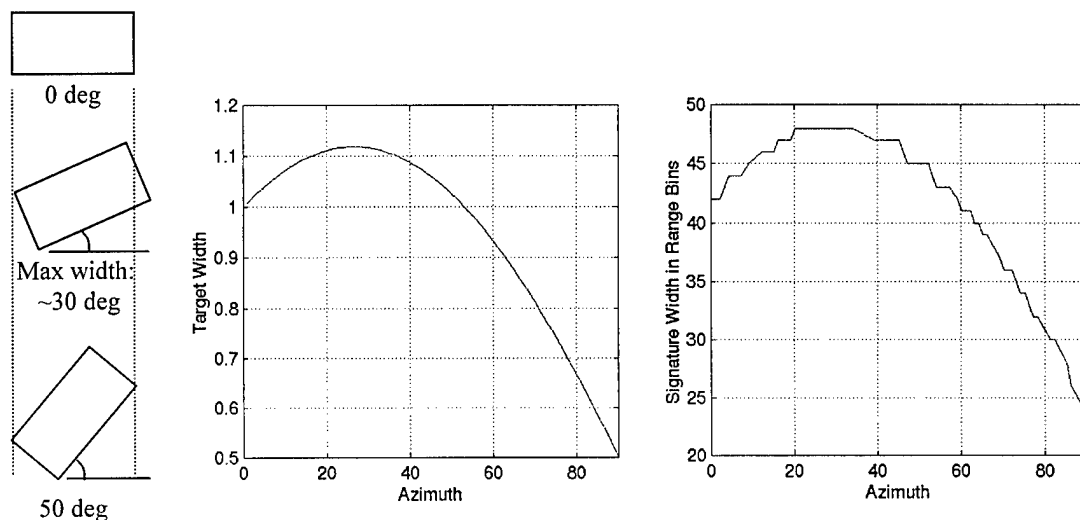


Figure 40 Sample target visual width and actual BMP-2 signature width

4.4 Application to Two Degrees of Freedom in SAR

Next, the pose estimation technique was extended to predict both azimuth and elevation for SAR using the MSTAR(IU) Mixed Targets data set (DARPA, 1998a). Three targets in this data set (2S1, BRDM-2, and ZSU-23/4) include 15, 30, and 45 degree elevation coverage. These targets and elevations were used to construct the training and testing set. Training was performed using azimuths separated by approximately 3.5 degrees. All remaining non-training images were used for testing.

Two alternative representations for elevation were tested as discussed in Chapter 3. Results for these two alternatives are presented in the following sections.

4.4.1 Concentric Circle Representation. The network was trained for the 0-180 degree range on the 2S1 target. Initially, increasing radius was used to represent increasing elevation. However, elevation estimation results were poor.

Comparing target images at different elevations provided insight into how to change the representation for improved results. Figure 41 shows the 2S1 target at 15, 30, and 45 degrees elevation. The lower elevations produce a larger radar image, a

Class/type	# test signatures	Azimuth	Estimate	Elevation	Estimate
		Error mean (deg)	Error std dev (deg)	Error mean (deg)	Error std dev (deg)
2S1	560	2.72	2.46	3.72	2.84
BRDM2	558	7.22	7.73	13.92	7.98
ZSU23/4	561	5.37	4.94	11.34	8.07

Table 4 SAR 2-DOF concentric circle results (MSE training on 2S1)

larger radar shadow, and less clutter noise than higher elevations. Thus the network weights applied to a larger, clearer image would tend to produce a *larger* output. The code was thus changed to reverse the concentric circles so that lower elevations produced circles of larger radius. This change improved test results dramatically. Results for all three vehicles are shown in Table 4. The output space for the 2S1 target is shown in Figure 42, and the estimation results are shown in Figures 43 and 44.

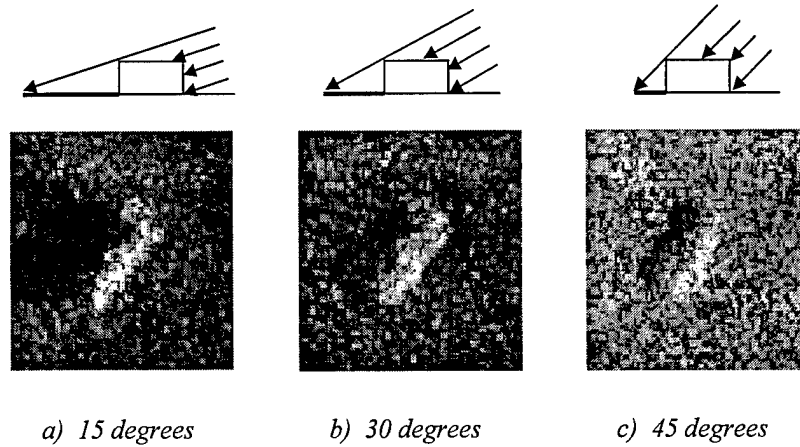


Figure 41 Sample 2S1 images at varying elevation

It is apparent from Table 4 that single-vehicle training generalizes well for azimuth but not for elevation. To correct the large elevation errors, a network was trained using all three targets. Results are shown in Table 5. For this multi-class training, mean elevation error decreased for the two additional vehicles, but azimuth

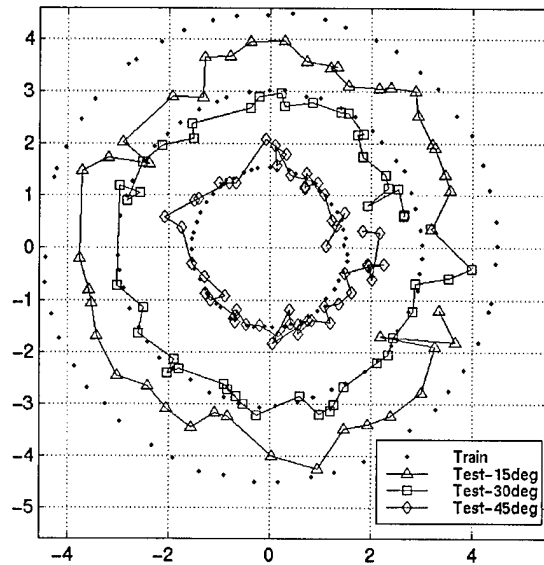


Figure 42 SAR concentric circle output: trained on 2S1, tested on 2S1

Class/type	# test signatures	Azimuth	Estimate	Elevation	Estimate
		Error mean (deg)	Error std dev (deg)	Error mean (deg)	Error std dev (deg)
2S1	560	3.68	3.15	5.62	3.98
BRDM2	558	10.35	15.02	9.04	7.98
ZSU23/4	561	6.17	9.23	7.85	7.98

Table 5 SAR 2-DOF concentric circle results (MSE training on three vehicles)

error increased slightly for all vehicles. Accuracy could probably be increased further by using more targets and more elevations.

A significant discovery was also made by comparing MI and MSE training. The mean-squared-error method performed as expected, but the mutual information method would not converge to concentric circles. (All the above results were obtained using mean-squared-error training.) Mutual information succeeded in producing circles of different diameter, but their usual final shape was crossed or twisted. This effect is analyzed in more detail later.

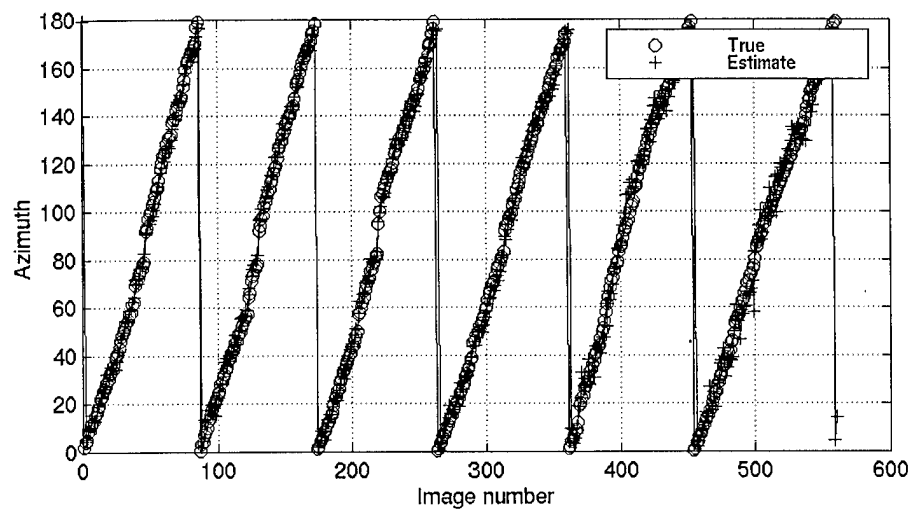


Figure 43 SAR concentric circle azimuth results: trained on 2S1, tested on 2S1

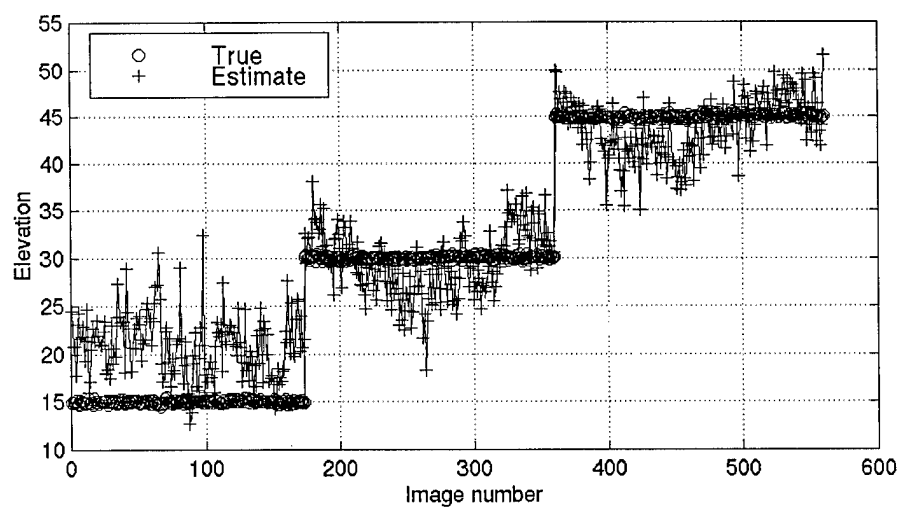


Figure 44 SAR concentric circle elevation results: trained on 2S1, tested on 2S1

Class/type	# test signatures	Azimuth	Estimate	Elevation	Estimate
		Error mean (deg)	Error std dev (deg)	Error mean (deg)	Error std dev (deg)
2S1	560	2.33	1.91	1.58	1.23
BRDM2	558	5.80	5.30	3.33	2.40
ZSU23/4	561	5.02	4.87	4.52	2.48

Table 6 SAR 2-DOF cylinder results (MSE training on 2S1)

4.4.2 Cylinder Representation. With elevation now represented by height in the third dimension, the network was again trained on the 2S1 target for the 0-180 degree range. Results for all three vehicles are shown in Table 6. The output space for the 2S1 target is shown in Figures 45 through 47, and the estimation results are shown in Figures 48 and 49. Performance was significantly better than the concentric circle representation, especially for elevation. Generalization also improved; the network trained on just one vehicle produced low errors for the other vehicles as well.

In contrast to the concentric circle representation, both the MSE and MI training methods converged easily for the cylinder representation. Performance of the MI and MSE networks was similar. The MSE network produced lower mean azimuth errors (up to seven degrees lower), while the MI network produced lower mean elevation errors (up to two degrees lower).

4.4.3 Performance Analysis. As stated previously, network performance (both accuracy and generalization) with the cylinder representation was significantly better than with the concentric circle representation. This improvement is undoubtedly due to the addition of the third output node. The two-output network was forced to use the same set of weights to calculate both azimuth and elevation in the output space. This dual utilization was possible (due to the difference in image size at different elevations), but it proved to produce sub-optimal results. When a third output was added to represent elevation exclusively, the additional set of

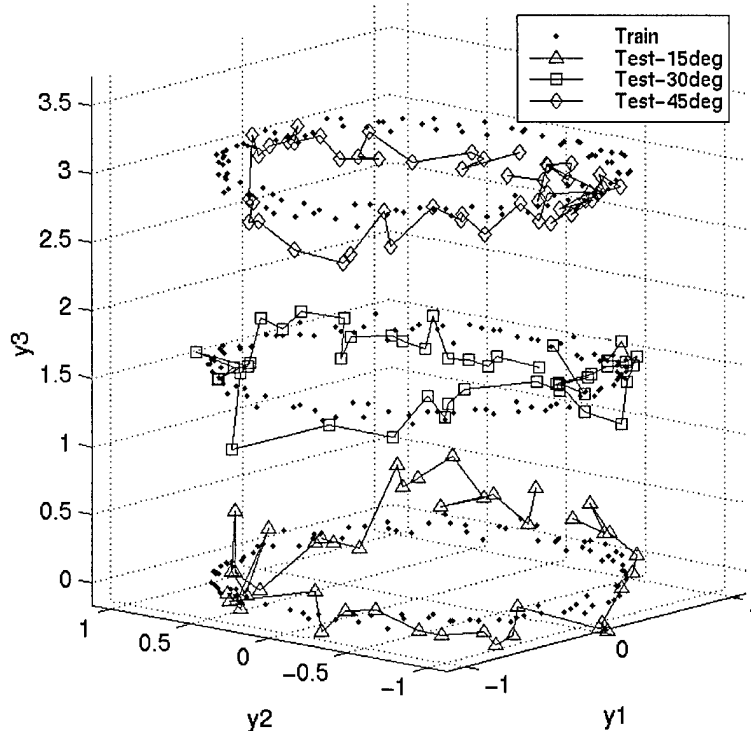


Figure 45 SAR cylinder output: trained on 2S1, tested on 2S1

weights was optimized to provide the elevation estimate. The weight visualizations for the concentric circle network and cylinder network are shown in Figures 50 and 51, respectively. Note that the weights for output y_3 emphasize the size of the target image and shadow, and thus they provide an optimal elevation estimate.

Another significant difference between the concentric circle and cylinder representations was the success of the mutual information method. MI training was unable to converge for concentric circle training, creating instead crossed or twisted circles as shown in Figure 52. This crossed circle effect occurs because the MI method adapts the network based on the *shape* of the output pdf rather than the *exact location* of the output points. If the circles do not become concentric within the first few iterations, then untangling them requires a temporary large *decrease* in mutual information to make the circles cross, and the algorithm does not allow this

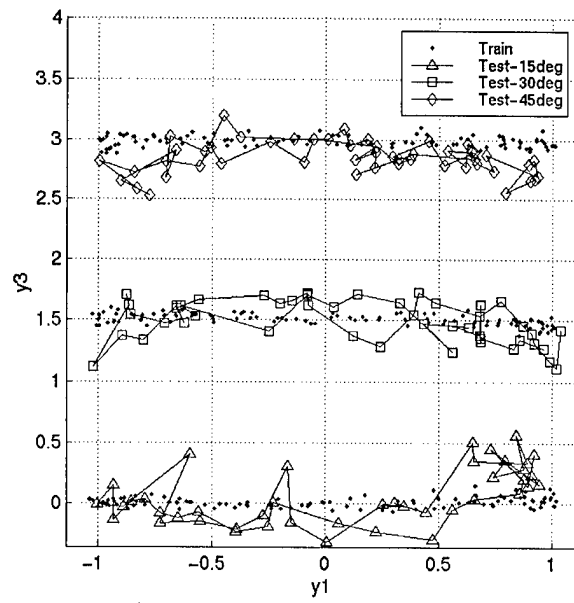


Figure 46 SAR cylinder output (side view)

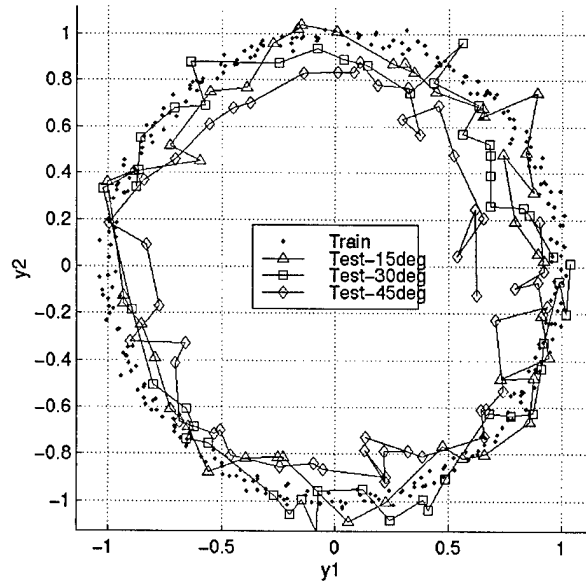


Figure 47 SAR cylinder output (top view)

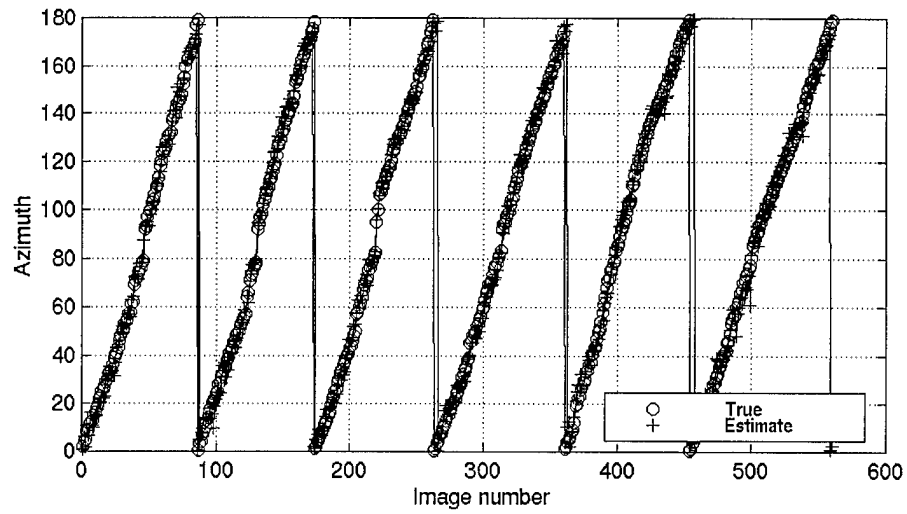


Figure 48 SAR cylinder azimuth results: trained on 2S1, tested on 2S1

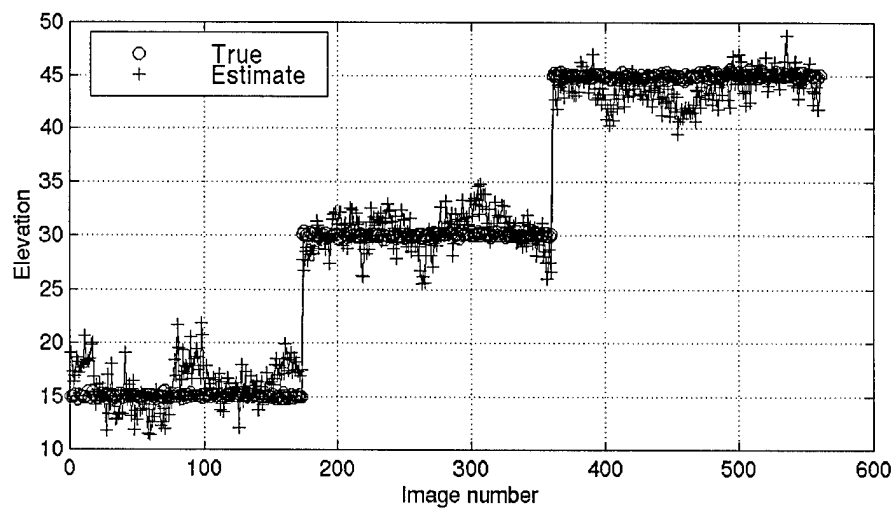


Figure 49 SAR cylinder elevation results: trained on 2S1, tested on 2S1

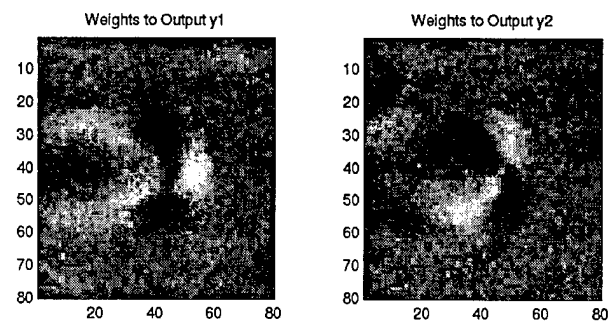


Figure 50 Weight visualization for SAR concentric circle network

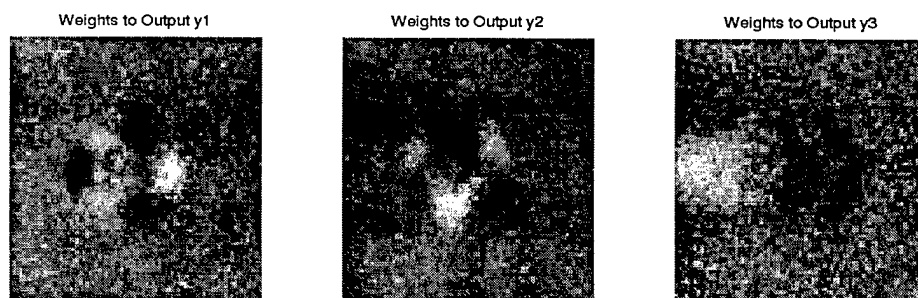


Figure 51 Weight visualization for SAR cylinder network

decrease. Thus the mutual information training method has difficulty converging for complex pdfs of this type. When the third output node is added, the circles for each elevation can separate in three-dimensional space and orient themselves correctly, as shown in Figure 53.

In conclusion, the feasibility of estimating both azimuth and elevation for SAR was clearly demonstrated. The cylindrical representation using three output nodes proved more accurate than the concentric circle representation with two output nodes. Both mean-squared-error and mutual information were validated as useful and roughly equivalent training criteria, although mutual information showed convergence problems for some complex pdfs, such as concentric circles.

4.5 *Application to Two Degrees of Freedom in HRR*

The final application investigated was azimuth and elevation estimation for HRR. Data at multiple elevations was not available for ground targets, so all tests were performed on aircraft targets. The data set poses included a range of 30 degrees in azimuth and 35 degrees in elevation.

Because the azimuth range was not periodic, the circular azimuth representation was not used. Instead, two output nodes were used, one each to represent azimuth and elevation. The output space was thus in the form of a grid, as shown in Figure 54. Due to the end effects caused by the mutual information method for a non-periodic data set, all training was performed using the mean-squared-error method.

4.5.1 Synthetic Data Testing. Testing was performed on synthetic HRR data. Two thirds of the data was used for training, and the remaining data was used for testing. Testing points were spread uniformly over azimuth and elevation.

The network was trained on each aircraft and then tested on the same aircraft. Results for each aircraft are shown in Table 7. The table indicates that the network

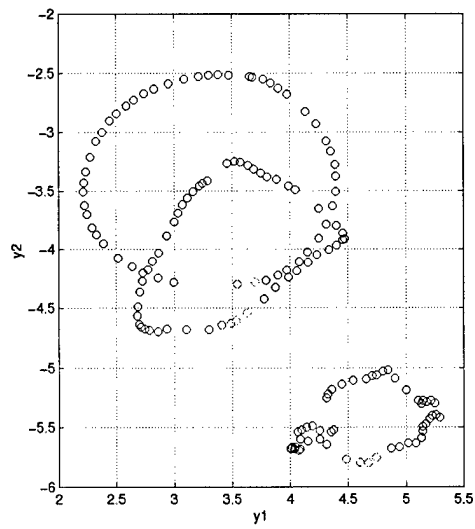


Figure 52 Concentric circle training results using mutual information

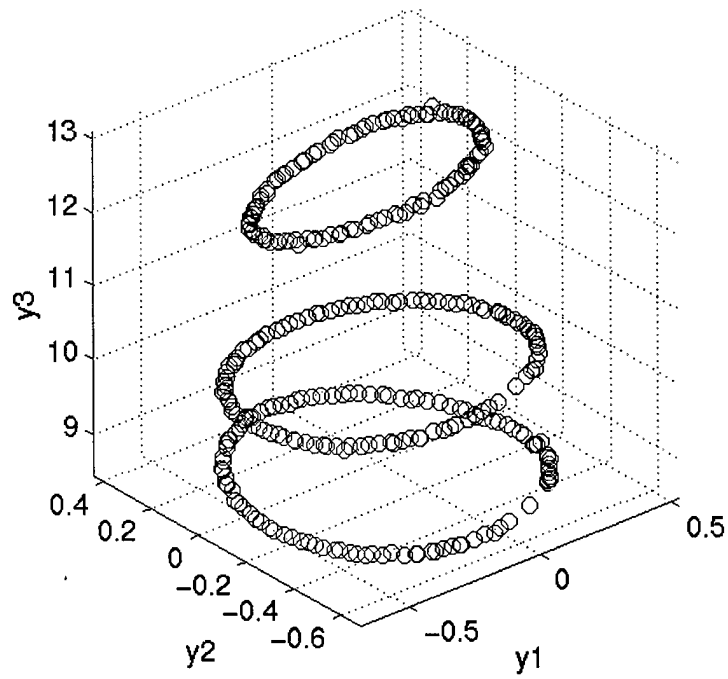


Figure 53 Cylinder training results using mutual information

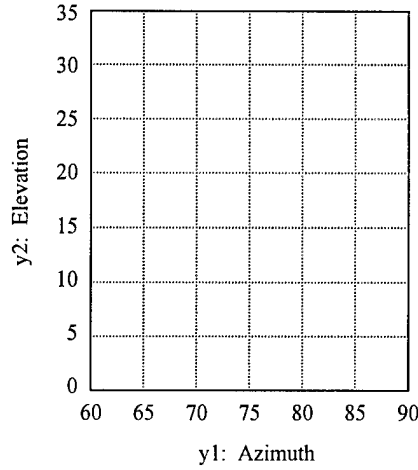


Figure 54 Output space for HRR aircraft pose estimation

Class	# test signatures	Azimuth	Estimate	Elevation	Estimate
		Error mean (deg)	Error std dev (deg)	Error mean (deg)	Error std dev (deg)
A	255	2.98	2.44	3.77	2.85
B	255	4.09	3.43	5.49	4.24
C	255	2.78	2.19	4.63	3.69
D	255	1.95	1.58	4.10	3.35
E	255	2.15	1.71	3.27	2.55
F	255	2.09	1.47	3.68	3.00

Table 7 Aircraft HRR 2-DOF results (MSE training and testing on same aircraft)

performs well for the target it is trained on, with elevation error slightly higher than azimuth error. A sampling of the output space for aircraft *E* is shown in Figure 55; the estimation results for the same aircraft are shown in Figures 56 and 57.

Limited cross-class testing was also performed. A network trained on aircraft *A* was tested on the other aircraft classes. Results showed a near total lack of generalization—the estimated pose appeared random, as shown in Figure 58.

4.5.2 Performance Analysis. The results indicate that aircraft HRR signatures do contain enough pose-related information to predict the pose of a specific aircraft with good accuracy. Further experiments with measured data and with

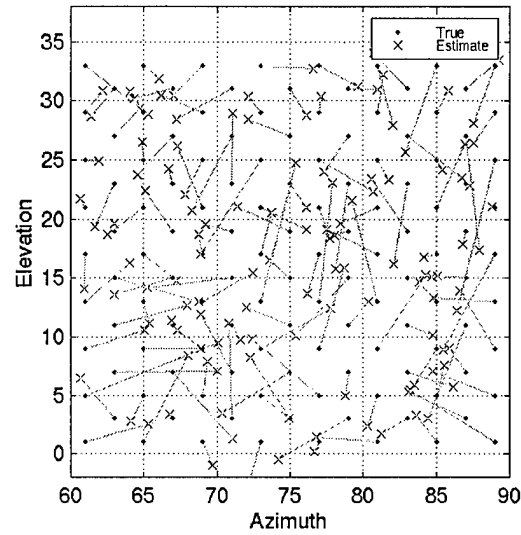


Figure 55 HRR 2-DOF output: trained on aircraft E, tested on aircraft E

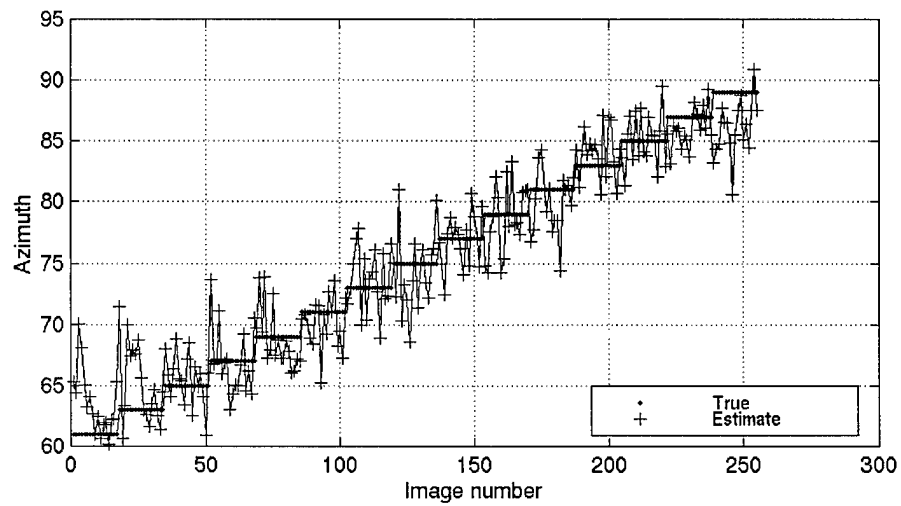


Figure 56 HRR 2-DOF azimuth results: trained on aircraft E, tested on aircraft E

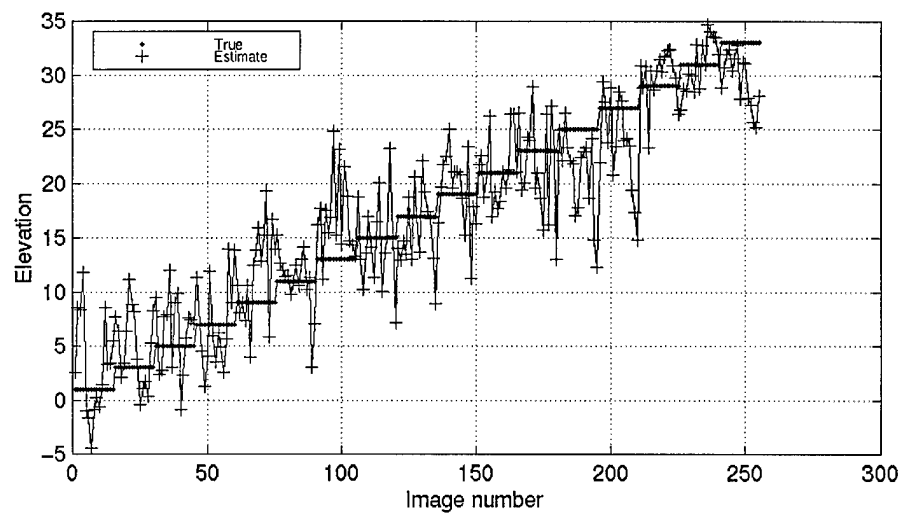


Figure 57 HRR 2-DOF elevation results: trained on aircraft E, tested on aircraft E

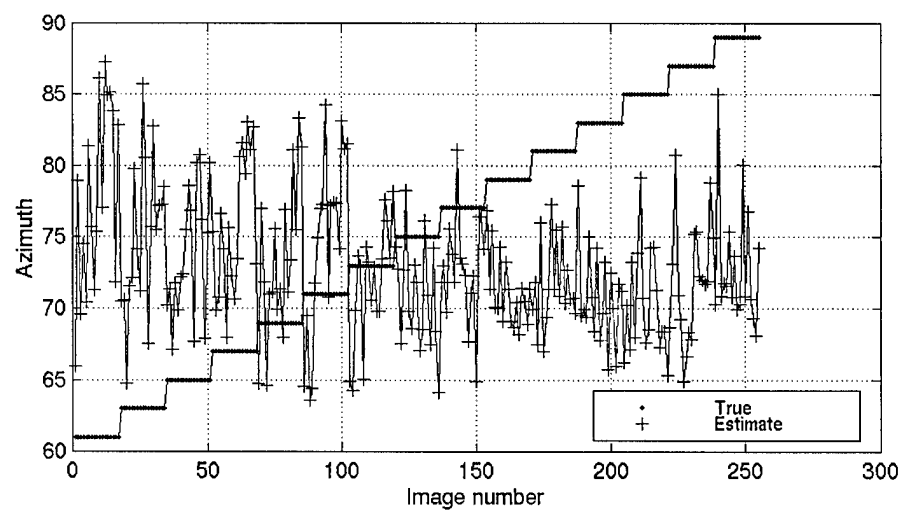


Figure 58 HRR 2-DOF azimuth results: trained on aircraft A, tested on aircraft B

varying configurations of the same aircraft are needed to further validate these preliminary results.

Unlike HRR for ground targets, the network training is highly specialized to the type of aircraft and does not generalize well. This lack of generalization between aircraft classes is not surprising. For SAR ground targets, the network capitalized on the general similarities in target shape and size. Aircraft targets are much more varied in both shape and size than ground vehicles. For example, Figure 59 shows the signatures of three of the aircraft at the same pose. It is evident from the dramatic differences in these signatures that it would be difficult to create a single network to estimate pose for all three aircraft. Further research is needed to solve this cross-class pose estimation problem.

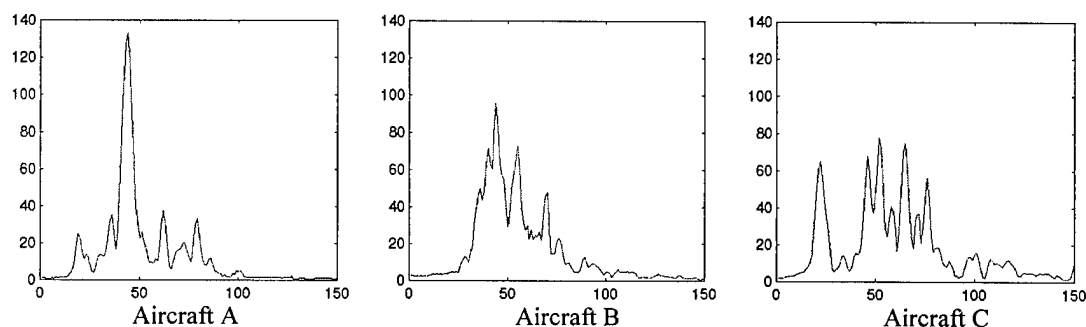


Figure 59 Sample aircraft HRR signatures at identical pose

Because of the limited range of poses in the data set, no conclusions can be drawn about symmetries in the HRR signatures or their effect on pose estimation. If such symmetries exist, the network's pose estimate could possibly be refined through comparison with aircraft tracking information. As demonstrated with ground targets, signature symmetries generate multiple possible true poses from the network's pose estimate. However, aircraft are always oriented roughly in the direction of their velocity vector (i.e., they cannot fly backward) and thus some of the possible poses could be eliminated from consideration. The velocity vector and the HRR

pose estimation could be used together to refine and build confidence regarding the true aircraft pose.

4.6 Summary

The pose estimation techniques of Principe were validated and progressively extended through a series of experiments.

First, weight analysis was performed, revealing that the network uses the edge regions of a SAR target image to predict azimuth. The network accurately predicts azimuth over the 180 degree range for target classes of similar size and shape. However, 360 degree estimation requires use of target-specific characteristics and proved unreliable across classes.

The adaptive network technique was then applied to estimate azimuth of HRR ground targets. Estimates were within 15 degrees of actual azimuth, which was less accurate than for SAR images of the same targets. Vehicle symmetries also restricted the estimation range to 90 degrees instead of 180 degrees. Due to this restriction, the mutual information method produced pdf end effects that rendered it less effective than the mean-squared-error method.

The technique was then extended to predicting both azimuth and elevation for SAR images. Two representations for elevation were tested: the cylindrical representation produced better results than the concentric circle representation, with accuracies within six degrees for azimuth and five degrees for elevation.

Finally, the mean-squared-error training method was applied to estimate azimuth and elevation for a limited set of HRR aircraft data. The network was able to predict azimuth within four degrees and elevation within six degrees, but only for the aircraft the network was trained on. Large differences between the signatures for different types of aircraft prevented the network from predicting pose across classes. Further research is necessary to solve this problem.

V. *Conclusions and Recommendations*

This research investigated the use of adaptive networks for target pose estimation, building upon the work of Principe and Xu at the University of Florida. Principe and Xu developed the mutual information training method for adaptive networks and applied it to SAR azimuth estimation. Their techniques were successfully extended here to estimate pose in two degrees of freedom (azimuth and elevation) and to estimate pose using HRR signatures. Results demonstrate that adaptive networks present significant potential for improving the state of the art in target pose estimation.

Conclusions and recommendations for each of the areas investigated are summarized in the following sections.

5.1 *Conclusions*

5.1.1 Advantages of Adaptive Networks for Pose Estimation. The adaptive network technique provides several advantages over other pose estimation methods:

1. *Accuracy.* Testing on SAR images yielded pose estimation accuracy equal to or better than the other estimation methods reviewed.
2. *Independence from classification.* While most pose estimation methods are closely tied to the algorithm used for classification, the adaptive network technique is independent of the classification method. This independence avoids the preprocessing and representational baggage specific to many other pose estimation methods and contributes to usability in a modular recognition system such as MSTAR.
3. *Speed.* Once a pose estimation network has been trained, execution is fast and efficient, which makes it suitable for real time systems.

4. *Flexibility.* The adaptive network technique is flexible in that the same algorithm can be applied successfully to both SAR and HRR and possibly to other signal types as well.

5.1.2 SAR Image Understanding. Principe's work demonstrated the success of SAR azimuth estimation for the 0-180 degree range, while showing some symmetry difficulties for the 0-360 degree range. This research revealed that for 180 degree prediction, the network performs edge region analysis of the target, which capitalizes on vehicle rectangularity. However, 360 degree prediction uses very target-specific characteristics to distinguish between symmetric poses, rendering the estimate highly unreliable. Thus the 180 degree estimation should be used and a subsequent recognition process should check both symmetrical poses.

5.1.3 HRR Azimuth Estimation (Ground Targets). The network successfully predicted ground target azimuth for the 0-90 degree range, with a mean error of 11 degrees or less for 80 percent of target classes. However, these mean errors are higher than those obtained using SAR images of the same targets.

The effective azimuth estimation range was restricted to 90 degrees due to symmetries in the vehicle signatures. Because of this restriction, estimation produces four possible true poses (versus two for SAR), all of which must be searched during the subsequent recognition process.

The combination of higher errors and more symmetries indicate the need for further research to refine HRR pose estimation for ground targets.

5.1.4 SAR Azimuth and Elevation Estimation. Target pose in two degrees of freedom (azimuth and elevation) was successfully predicted using networks with either two or three output nodes. For the two-dimensional output space, the training points were represented as concentric circles, with one circle for each elevation. The

three-dimensional output space used a cylindrical representation, with the training points at each elevation producing a circle at a different height.

The cylindrical representation produced the best performance, with mean azimuth error less than six degrees and mean elevation error less than five degrees. To predict elevation the network used the size of the target and radar shadow, capitalizing on the fact that low elevations produce an elongated target and shadow.

The ability to predict elevation is a significant step forward and is a departure from the traditional method of trying to generalize azimuth prediction over all ranges in elevation.

5.1.5 HRR Azimuth and Elevation Estimation (Air Targets). Pose estimation for the same class the network was trained on yielded good results, with mean errors of four degrees in azimuth and six degrees in elevation. However, cross-class testing yielded poor results: a network trained on one type of aircraft produced an almost random pose estimate for a different type of aircraft. The cross-class problem is due to the highly variable shape and size of aircraft, which produce highly dissimilar HRR signatures. Further research is needed to solve this problem.

Results for aircraft HRR may be considered preliminary, since testing was performed for a limited range of poses using synthetic rather than measured data.

5.1.6 Network Architecture. For SAR pose estimation, a linear network performed as well as a multi-layer perceptron. For HRR pose estimation, the multi-layer perceptron performed better, presumably because the signatures had more complex information contained in a smaller input vector.

5.1.7 Comparison of Training Criteria. The performance of the mutual information training criterion developed by Principe was compared with traditional mean-squared-error training for all testing scenarios. In most cases, both methods

provided equivalent results and used the same characteristics of the training images to estimate pose.

However, there were two cases in which mutual information showed limitations. First, for complex pdfs (such as the concentric circle representation of azimuth and elevation), the mutual information method was unable to converge to a solution. Second, when the output space of the training set was not periodic (i.e., did not form a circle), pdf end effects caused inaccurate estimation results.

In cases where the MI training method could converge, training was rapid. The MI method took up to an order of magnitude fewer iterations than the MSE method. However, each MI iteration took longer since it has a complexity of $O(n^2)$ (where n is the number of training exemplars) versus a complexity of $O(n)$ for MSE training. Thus MI training is faster than MSE for a small number of exemplars but slower for a large number of exemplars.

5.2 Recommendations

5.2.1 SAR Ground Targets. While the SAR two degree-of-freedom tests yielded very promising results, they were performed with limited data. A sufficient elevation range was available for only three vehicles with three elevations for each vehicle. Further testing should be done (using more vehicles and more elevations) to better evaluate the estimation algorithms.

5.2.2 HRR Ground Targets. The high errors produced for HRR ground targets might be reduced by a change in network architecture. Chapter 2 detailed how Agarwal used a clustering architecture to improve pose estimation for a multi-layer perceptron network (Agarwal, 1998). Clustering divided the training data into groups with similar characteristics, and a separate MLP was then trained for each grouping. Unknown inputs were first run through the clustering layer, which directed them to the appropriately trained MLP for final pose estimation. A similar

architecture could be applied to the HRR pose estimation problem. Because HRR signatures often change dramatically with pose, this architecture might improve estimation accuracy by allowing each MLP to learn a smaller, less complex portion of the entire input space.

The problem of symmetrical poses could be reduced if a velocity vector was available for the moving ground target. A moving ground vehicle is likely to be traveling relatively straight forward (or possibly backward). The vehicle velocity vector and HRR pose estimate could be used together to eliminate some of the four possible poses.

Finally, pose estimation for ground targets should be extended to two degrees of freedom. (Data at multiple elevations was not available for this research.)

5.2.3 HRR Air Targets. The clustering scheme described above should also be implemented for aircraft HRR. Aircraft signatures change with pose even more dramatically than do ground vehicle signatures, so a clustering architecture would very likely provide an improvement in accuracy.

More thorough testing using measured data is also needed, since all tests were performed using synthetic data.

While the limited data set provided no proof of symmetries for aircraft HRR signatures, it is likely that such symmetries exist. If so, the pose estimate could be improved by using the velocity vector, as described in the previous section for ground targets. Even more than ground vehicles, fixed-wing aircraft are always oriented roughly in the direction of their velocity vector – i.e., they cannot fly backward – and thus some possible poses could be eliminated from consideration. The velocity vector and the HRR pose estimate could be used together to refine and build confidence regarding the true aircraft pose.

Finally, the cross-class pose estimation problem needs to be addressed. The only way to enable pose estimation across aircraft classes is to seek out the com-

monalities between the signatures of different aircraft. Some possibilities are to find common peak locations (corresponding to features such as nose, wings, and tail), or to compare the number of peaks for different aircraft, relative size of peaks, etc.

5.2.4 Reducing End Effects. The mutual information estimation method produced inaccurate results for non-periodic data sets, due to pdf end effects. Further investigation may reveal a way to reduce or eliminate these effects and increase accuracy. One possibility is to rescale the pdf near the ends to effectively stretch the region of higher accuracy. Another option is to divide the pdf into a center (accurate) region and two end regions. Separate networks could be trained with overlapping pose angle ranges so that a test image can always fall into the center region of one network. A classifier or clustering layer could then direct a test image to the appropriate network (in which it falls in the center, higher accuracy region).

5.2.5 Leave-One-Out Testing. Experiments throughout this research effort were performed by dividing each data set into a non-overlapping training set and test set. More accurate results might be obtained by using the leave-one-out testing method. In this method, the network is trained on *all* available exemplars except one, which is then used for testing. The same procedure is then repeated, leaving each exemplar out in turn. The average estimation accuracy of all the leave-one-out trials could provide a better measure of the true network estimation ability.

Appendix A: SAR Two-Degree-of-Freedom Code

This appendix contains the MATLAB code used to train and test a two degree-of-freedom pose estimation network using the mutual information method. Due to space limitations, the other programs used to conduct the experiments in this thesis are not included here.

Questions regarding the code and information on obtaining electronic copies of any of the programs may be directed to Andrew.Learn@sensors.wpafb.af.mil or Louis.Tamburino@sensors.wpafb.af.mil. In the event these addresses change over the course of time, you may contact Captain Learn at the permanent address listed in the Vita.

A.1 Mutual Information Network Training

```
%=====
%
% FILE      : train_SAR_2DOF.m
%
% TYPE      : Matlab program
%
% AUTHOR    : Original 1-DOF program done in PV-Wave by Dongxin Xu, Computational
%             NeuroEngineering Lab, Univ of Florida. Copy of original obtained
%             by Capt Andrew Learn on 24 April 1998. Translated to Matlab and
%             modified extensively by Capt Learn for thesis research at the
%             Air Force Institute of Technology.
%
% PURPOSE   : Trains a network to estimate pose using Mutual Information method
%             using SAR signatures from the MSTAR dataset. Estimates both
%             azimuth and elevation of targets.
%
% PARAMETERS: No external arguments. Set parameters below.
%
% NOTES     : - Requires data to be preprocessed by the set_data MATLAB program.
%             - Uses Matlab function files pose_prepare.m and plot_point_force.m
%             - This program modified from the train_SAR program.
%
% LAST MODIFIED : 9 Jan 99
%
%=====

%=====
% SET CONSTANTS
%=====

%Set debug flag for displaying debug plots and variable info (1=on, 0=off)
debug = 0;

%Set a target grayscale colormap: Pixelvalue=0 => white; Pixelvalue=255 => black
maprange = 0 : 255;
maprange = maprange / 255;
```

```

target_graymap = [maprange' maprange' maprange'];

%=====
%!!!!!!!!!!!!!!!!!!!!!! SET CONTROL PARAMETERS !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
%=====
%!!CHOOSE FILE NAME FOR SAVING TRAINED NETWORK AT BOTTOM OF PROGRAM!!

%Choose how many different targets the network will be trained for
num_targets = 3;

%Choose file names of training data to be read
%traindata_filename(1,:) = 'SARdata/t72_s7_17deg_train.dat';
%traindata_filename(1,:) = 'SARdata/t72_132_17deg_train.dat';
%traindata_filename(1,:) = 'SARdata/t72_812_17deg_train.dat';
%traindata_filename(2,:) = 'SARdata/bmp2_c21_17deg_train.dat';
%traindata_filename(1,:) = 'SARdata/bmp2_9563_17deg_train.dat';
%traindata_filename(1,:) = 'SARdata/bmp2_9566_17deg_train.dat';
%traindata_filename(3,:) = 'SARdata/btr70_c71_17deg_train.dat';

traindata_filename(1,:) = 'SARdata/2s1_15deg_train.dat';
traindata_filename(2,:) = 'SARdata/2s1_30deg_train.dat';
traindata_filename(3,:) = 'SARdata/2s1_45deg_train.dat';

%Convert the name matrix to a cell array (removes blank spaces)
traindata_filename = cellstr(traindata_filename);

train_angle_range = 180; % range of training angles
%train_angle_range = 360;

use_symmetries = 1; %Switch to use 180-360 symmetrical images
%in 0-180 training (1=on, 0=off)

num_iterations = 100; % number of iterations

num_output_nodes = 3;
elev_rep = 2; %Representation method for elevation
% 2=cylinder, 3=sphere

sigma2_aspect = 0.1; % variance of Gaussian Window for angles
sigma2_y = 0.1; % variance of Gaussian Window for outputs

step_size = .1e-9; % overall step size
step_size_display = 0.2; % adjustment of force for display purposes

%Calculate readjustment step size:
step_size_readjust = step_size ./ step_size_display;

%=====
% READ TRAINING DATA FROM FILE
%=====

for filenum = 1 : num_targets

    clear images aspect elevation;

    % Open training data file
    fid_train = fopen(char(traindata_filename(filenum)), 'rb', 'ieee-be');

    disp(fid_train);

    % Read the number of images in the training file
    num_training_images = fread(fid_train, 1, 'ushort');

    % Loop through training images
    disp('Reading training images');
    for i = 1 : num_training_images

```

```

        % Print the number of the current image being read
        disp(['Image number = ' num2str(i)]);

        % Read the image into a variable
        images(:,i) = fread(fid_train, 6400, 'uchar');
    end

    % Similarly, read all aspect angles
    disp('Reading aspect angles');
    for i = 1 : num_training_images

        % Print the number of the current aspect angle being read
        disp(['Aspect number = ' num2str(i)]);

        % Read the image into a variable
        aspect(i,1) = fread(fid_train, 1, 'float');
    end

    % Similarly, read all elevation angles
    disp('Reading elevation angles');
    for i = 1 : num_training_images

        % Print the number of the current elevation angle being read
        disp(['Elevation number = ' num2str(i)]);

        % Read the image into a variable
        elevation(i,1) = fread(fid_train, 1, 'float');
    end

    %For multiple targets, use temp variable to accumulate 2nd, 3rd, etc. targets
    if filenum == 1
        temp_images = images;
        temp_aspect = aspect;
        temp_elevation = elevation;
    else
        temp_images = [temp_images images];
        temp_aspect = [temp_aspect ; aspect];
        temp_elevation = [temp_elevation ; elevation];
    end

    % Close training data file
    fclose(fid_train);

end

%Rename variables to include multiple targets
images = temp_images;
aspect = temp_aspect;
elevation = temp_elevation;

num_training_images = length(aspect);

%=====
% Sort images in order of aspect angle, discard images > 180 deg if necessary
%=====

%Sort aspect angles in ascending order
[aspect, index] = sort(aspect);

%Rearrange images in order of increasing aspect angle
images = images(:,index);
elevation = elevation(index);

%If training only on 0..180 degree aspect angles, do the following
if train_angle_range == 180

    if use_symmetries == 0 %Use only 0-180 degree images

```

```

    %Find the last image less than or equal to 180 deg aspect angle
    last_image = 1;
    for i = 1 : num_training_images,
        if aspect(i) <= 180
            last_image = i;
        end
    end

    %Discard angles/images greater than 180 deg (data file contains 0-360 deg)
    aspect = aspect(1:last_image);
    elevation = elevation(1:last_image);
    images = images(:, 1:last_image);

elseif use_symmetries == 1    %Use 180-360 images as second set of 0-180

    %Take aspects mod 180 to convert 180-360 images to 0-180
    aspect = mod(aspect, 180);

    %Resort by aspect
    [aspect, index] = sort(aspect);
    elevation = elevation(index);
    images = images(:,index);

end

%Double the training angles to create a circle for output
aspect = aspect .* 2;

end

%Get size parameters of images matrix: use to set appropriate variables
num_input_nodes = size(images,1);    %number of pixels per image
num_training_images = size(images,2);    %number of images

%=====
% Prepare 4 display windows
%=====

%Position vector = [xposition yposition width height]

%Figure 1 displays the training images
figure(1);
set(1,'position',[0 20 550 600]);

%Figure 2 displays the current network outputs and mutual information forces
figure(2);
set(2,'position',[600 550 450 400]);
set(2,'PaperPosition',[1 1 4 4]); %Ensures square output to JPEG

%Figure 3 shows just the current network outputs (for a cleaner view)
figure(3);
set(3,'position',[300 650 290 250]);

%Figure 4 shows a visual of the current network weights and delta weights
figure(4);
set(4,'position',[0 650 290 250]);

%=====
% Display training images
%=====
%This displays only 56 of the training images. It selects samples evenly from
%throughout the training angles, just to give a visual of the training range.

%Set current window and colormap
figure(1);
colormap(target_graymap);

```

```

%Start image count at 0
count = 0;

%Choose the index interval between images to be displayed
index_step = num_training_images ./ 56;

%If fewer than 56 training images, show them all
if num_training_images < 56
    for i = 1 : num_training_images
        %Change image from column vector to matrix
        target = reshape(images(:,i), 80,80);
        %Plot image
        subplot(8,7,i);
        image(target);
        axis('off');
    end

%If more than 56 training images, show a sample of them
else
    for i = 1 : 56,
        %Choose index of image to plot
        image_index = floor(index_step .* i);
        %Change image from column vector to matrix
        target = reshape(images(:,image_index), 80,80);
        %Plot image
        subplot(8,7,i);
        image(target);
        axis('off');
    end
end

end

%=====
% Rename variables in network terms, for ease of use in following equations
%=====

Dx = num_input_nodes;      % Number of input nodes to neural net
Dy = num_output_nodes;     % Number of output nodes from neural net
Ny = num_training_images;  % Number of training images presented to network

%=====
% Initialize the network
%=====

%Randomly initialize weight matrix [2 x 6400] with normal distribution
weights = randn(Dy, Dx) .* 1.0e-5;
weights(3,1:Dx) = rand(1,Dx) .* 1e-4;

%=====
% Calculate entropy of angles
%=====

%Merge aspect and elevation into single variable "pose", where column1 = aspect
%and column2 = elevation.
pose = [aspect elevation];

%Call function pose_prepare to create potential fields for training angles
[Pasp, Paij, Pai, pose_points] = pose_prepare(pose, sigma2_aspect, Dy,
elev_rep);

%Divide marginal potential for each point by number of points (images)
Pai = Pai ./ Ny;           % just for mean-squared difference criterion

%Divide total aspect marginal potential by number of points (images) squared
Pasp = Pasp/(Ny.^ 2);      % just for mean-squared difference criterion

```

```

%Create factor for use in generating forces?
temp = Pai;
for i = 2 : Ny
    temp = [temp ; Pai];
end
factor = temp + temp' - Paij - Pasp; % just for mean-squared difference
criterion

%Display factor
if debug == 1
    figure;
    surf(aspect, aspect, factor);
    title('Factor: passed to mutual_info_force');
    disp('Hit enter to continue');
    pause;
end

%=====
% Perform training iterations
%=====

%Initialize movie
if debug == 1
    figure(2);
    axis([-1.5 1.5 -1.5 1.5]);
    train_movie = moviein(num_iterations);
end

for iter = 1 : num_iterations,

    %Multiply weights times images to get linear projection for every image
    % $[3 \times 6400] \times [6400 \times N] = [3 \times N] \Rightarrow y$  is a set of 3-element column vectors
    y = weights * images;

    %Find mutual information force
    [force, Jt] = mutual_info_force(Paij, Pai, Pasp, factor, y, Dy, Ny, ...
        sigma2_y, aspect);

    %Adjust forces for display purposes
    force = force .* step_size_display;

    %Record mutual information criterion:  $I = f(y,a)$ 
    J(iter) = Jt;

    %Calculate changes to weights
    dw = (step_size_readjust .* force) * images';

    %Display plots
    nil_return = plot_point_force( y, Dy, Ny, force, weights, dw);

    %Capture movie frame
    figure(2);
    train_movie(:,iter) = getframe;

    %Change weights
    weights = weights + dw;

    disp(['Iteration = ' num2str(iter)]);
    disp([' J = ' num2str(J(iter))]);
    disp([' Min w = ' num2str(min(min(weights)))]);
    disp([' Max w = ' num2str(max(max(weights)))]);
    disp([' Min dw = ' num2str(min(min(dw)))]);
    disp([' Max dw = ' num2str(max(max(dw)))]);
    disp([' Step size = ' num2str(step_size)]);
end

nil_return = plot_point_force( y, Dy, Ny, force, weights, dw);

```

```

=====
%!!!!!!!!!!!!!!!!!!!! SAVE TRAINED NETWORK !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
=====

%Calculate final network output values for all training images
y_train = weights * images;

%Choose file name for saving trained network
save SARnetworks/mi_lin_2sl_180_net_cyl2 weights y_train pose sigma2_y
sigma2_aspect pose_points train_angle_range;


=====
%
% FILE : pose_prepare.m
%
% TYPE : MATLAB function
%
% AUTHOR : Original 1-DOF program done in PV-Wave by Dongxin Xu, Computational
%          NeuroEngineering Lab, Univ of Florida. Copy of original obtained
%          by Capt Andrew Learn on 24 April 1998. Translated to Matlab and
%          modified extensively by Capt Learn for thesis research at the
%          Air Force Institute of Technology.
%
% PURPOSE: Function for pose estimation by quadratic mutual information.
%          Prepares the data for the forces from partial marginal potential
%          and overall space potential.
%
% INPUTS:
%   - pose : Pose angles of the training data
%   - sigma2_pose : Variance of the Gaussian Parzen window in the
%                  space for the pose angle
%   - Dy : Number of network outputs (2 or 3)
%   - elev_rep : Elevation representation
%               - 0 = none (azimuth estimation only, 2 network outputs)
%               - 1 = concentric circles (2 network outputs)
%               - 2 = cylinder (3 network outputs)
%               - 3 = sphere (3 network outputs)
%
% OUTPUTS:
%   - pose_points : A set of column vectors giving (y1,y2, and maybe y3)
%                  coordinates on the 2-d unit circle (or cylinder)
%                  corresponding to each pose angle
%   - Pij : Potential between each pair of data points
%   - Pi : Pose marginal potential for each data point
%   - Pa : Total pose marginal potential
%
% NOTES : This function has been made generic for use with one, two or
%          three network output nodes.
%
% LAST MODIFIED : 11 Jan 99
%
=====

function[Pa, Pij, Pi, pose_points]=pose_prepare (pose, sigma2_pose, Dy,
elev_rep)

%Set display flag (1 = on, 0 = off) for displaying plots and variable info
debug = 0;

%Find number of pose angles (number of points to be found on 2-D unit cylinder)
num_points = size( pose, 1 );

```

```

%Extract azimuth (aspect) angles and convert degrees to radians
aspect_radians = pose(:,1)' .* pi ./ 180;

%Create pose_points, whose first 2 elements give (x,y) coordinates of each
% aspect on the 2-D unit circle in column vector format.
pose_points(1,:) = cos( aspect_radians );
pose_points(2,:) = sin( aspect_radians );

%If only using one output node (Dy=1), revise pose_points to 1-D, range 0-1
if Dy == 1
    clear pose_points;
    pose_points = pose(:,1)' ./ 90;
end

%Extract elevation angles and create point in 2-D or 3-D space according
%to the method of elevation representation
if (Dy == 2) & (elev_rep == 1) %Concentric circles
    %Make each circle at radius elev/10 (el=10 deg at r=1,
    %el=20 deg at r=2, etc.)
    for i = 1 : length(pose)
        pose_points(:,i) = pose_points(:,i) .* (pose(i,2) ./ 10);
    end
elseif Dy == 3
    if elev_rep == 2 %Cylinder
        %Make each circle at height elev/10-1.5. (el=15 deg at h=0,
        %el=30 deg at h=1.5, el=45 deg at h=3.0)
        pose_points(3,:) = (pose(:,2)' ./ 10) - 1.5;
    elseif elev_rep == 3 %Sphere
        % NOT IMPLEMENTED
    end
end

%Calculate and display entire probability density for pose points
% Will overlap elevations if multiple elevations (Dy=3) are used.
if debug == 1
    if Dy > 1
        %[x,y]=meshgrid(-1.5:.05:1.5);
        [x,y]=meshgrid(-5.5:.2:5.5);
        pdf = exp(-((x-pose_points(1,1)).^2+(y-pose_points(2,1)).^2) ...
            ./sigma2_pose)./num_points;

        figure(5);
        set(5,'position',[0 40 450 400],'PaperPosition',[1 1 3.5 3]);
        colormap(gray);
        mesh(x,y,pdf);
        xlabel('t1'); ylabel('t2');
        %axis([-1.5 1.5 -1.5 1.5 0 .02]);
        axis([-5.5 5.5 -5.5 5.5 0 .02]);
        caxis([0 100]);
        pdf = pdf .* num_points;
        pause;
        for k = 2 : num_points
            pdf = pdf + exp(-((x-pose_points(1,k)).^2+(y-pose_points(2,k)).^2) ...
                ./sigma2_pose);
        end
        pdf = pdf ./ num_points;
        mesh(x,y,pdf);
        %axis([-1.5 1.5 -1.5 1.5 0 .08]);
        axis([-5.5 5.5 -5.5 5.5 0 .08]);
        caxis([0 100]);
        xlabel('t1'); ylabel('t2');
    elseif Dy == 1
        x = [-1 : .01 : 2];
        pdf = zeros(size(x));
        for k = 1 : num_points
            pdf = pdf + exp(-((x-pose_points(k)).^2))./sigma2_pose);
        end
    end
end

```

```

        pdf = pdf ./ num_points;
        figure(5);
        plot(x,pdf);
        xlabel('t'); ylabel('Probability');
    end
end

%Create a 3-D difference matrix (Dy x num_points x num_points), which calculates
%the difference between each training point & every other training point

%Expand pose_points into matrices of size [Dy x num_points x num_points]
for k = 1 : num_points
    a_i(:, :, k) = pose_points;
    a_j(:, k, :) = reshape(pose_points, Dy, 1, num_points);
end
if Dy == 1
    a_j = a_j';
    a_j = reshape(a_j, Dy, num_points, num_points);
end

%Calculate difference between each pair of points
diff = a_j - a_i;

%Calculate potential between each pair of points using a Gaussian based
%on sum squared difference
Pij = exp( -(sum((diff.^2), 1)) ./ (4 .* sigma2_pose) );

%Reshape back to 2 dimensions
Pij = reshape(Pij, num_points, num_points);

%Calculate marginal potential for each point
Pi = sum(Pij);

%Calculate total pose marginal potential
Pa = sum(Pi);

%Display debug information
if debug == 1

    %Plot potentials and marginal potentials
    disp('Hit Enter to see potential between all pairs of aspect points');
    pause;
    figure(5);
    colormap(cool);
    surf(Pij);
    xlabel('Pose number');
    ylabel('Pose number');
    title('Potential Between All Pairs of Poses');
    disp('Hit Enter to see the marginal potential for each pose angle');
    pause;
    plot(Pi);
    %axis([0 360 0 12]);
    xlabel('Pose number');
    title('Marginal Potential for All Poses');
    disp('Hit Enter to continue');
    pause;
    close(5);
end

return

```

```

%=====
%
% FILE : plot_point_force.m
%
% TYPE : MATLAB function
%
% AUTHOR : Original 1-DOF program done in PV-Wave by Dongxin Xu, Computational
%          NeuroEngineering Lab, Univ of Florida. Copy of original obtained
%          by Capt Andrew Learn on 24 April 1998. Translated to Matlab and
%          modified extensively by Capt Learn for thesis research at the
%          Air Force Institute of Technology.
%
% PURPOSE : Function to plot the current network outputs and the forces being
%           exerted on those points for each iteration of pose estimation
%           training.
%
% INPUTS:
%   - y : Set of 2 or 3-element column vectors with output node values
%         for each image
%   - Dy : Number of output nodes (2 or 3)
%   - Ny : Number of training images
%   - force : Mutual information force (2 or 3-element vector)
%   - w : Network weights
%   - dw : Change in weights for the current iteration
%
% OUTPUTS:
%   - nil_return : dummy variable
%
% NOTES : This function has been made generic for use for either HRR or SAR,
%         and for either two or three network output nodes.
%
% LAST MODIFIED : 29 Dec 98
%=====

function nil_return = plot_point_force (y, Dy, Ny, force, w, dw)

% Set a target grayscale colormap: Pixelvalue=0 => white; Pixelvalue=255 =>
black
maprange = 0 : 255;
maprange = maprange / 255;
target_graymap = [maprange' maprange' maprange'];

%Select display routines (1=SAR, 2=HRR)
SAR_or_HRR = 1;

%Set display switches (1 = on, 0 = off)
show2 = 1; %Network outputs with forces
show3 = 1; %Network outputs without forces
show4 = 1; %Network weights and delta weights

%Calculate new coordinates dictated by mutual information force
delta = y + force;
delta1 = [y(1,:) ; delta(1,:)];
delta2 = [y(2,:) ; delta(2,:)];
if Dy == 3
    delta3 = [y(3,:) ; delta(3,:)];
end

%=====
% Figure 2: Plot current network output values and forces
%=====
if show2 == 1
    figure(2);

    if Dy == 2

```

```

%Plot output points
plot (y(1,:), y(2,:), 'ko');
if SAR_or_HRR == 1
    axis([-7 7 -7 7]); %Large viewing area for SAR
else
    axis([-1.5 1.5 -1.5 1.5]); %Smaller viewing area for HRR
end
hold on;

%Make beginning and end of circle different color
plot (y(1,1:5), y(2,1:5), 'go');
plot (y(1,Ny-4:Ny), y(2,Ny-4:Ny), 'ro');

%Plot forces acting on each point
line(delta1, delta2, 'color', 'c');

%Label axes
xlabel('y1');
ylabel('y2');

elseif Dy == 3

    %Plot output points
    plot3 (y(1,:), y(2,:), y(3,:), 'mo');
    axis([-3 3 -3 3 -3 6]);
    hold on;
    grid on;

    %Make beginning and end of circle different color
    plot3 (y(1,1:5), y(2,1:5), y(3,1:5), 'go');
    plot3 (y(1,Ny-4:Ny), y(2,Ny-4:Ny), y(3,Ny-4:Ny), 'ro');

    %Plot forces acting on each point
    line(delta1, delta2, delta3, 'color', 'c');
end

grid on;
axis equal;
hold off;
end

%=====
% Figure 3: Show network outputs without forces for a clearer view
%=====
if show3 == 1
    figure(3);

    if Dy == 2
        plot(y(1,:), y(2,:), 'ko');
        hold on;
        plot (y(1,1:5), y(2,1:5), 'go');
        plot (y(1,Ny-4:Ny), y(2,Ny-4:Ny), 'ro');
    elseif Dy == 3
        plot3(y(1,:), y(2,:), y(3,:), 'ko');
        hold on;
        plot3 (y(1,1:5), y(2,1:5), y(3,1:5), 'go');
        plot3 (y(1,Ny-4:Ny), y(2,Ny-4:Ny), y(3,Ny-4:Ny), 'ro');
    end

    grid on;
    axis equal;
    hold off;
end

%=====
% Figure 4: Show network weights and delta weights at each iteration
%=====

```

```

if show4 == 1
    figure(4);

    wmin = min(min(w));
    wmax = max(max(w));
    dwmin = min(min(dw));
    dwmax = max(max(dw));

    if Dy == 2

        if SAR_or_HRR == 1 %Display weights for SAR training
            colormap(target_graymap);

            %Reshape weights into an 80x80 image and shift them to a 255-grayscale
            w1 = (reshape(w(1,:),80,80) - wmin) .* 255 ./ (wmax-wmin);
            w2 = (reshape(w(2,:),80,80) - wmin) .* 255 ./ (wmax-wmin);
            dw1 = (reshape(dw(1,:),80,80) - dwmin) .* 255 ./ (dwmax-dwmin);
            dw2 = (reshape(dw(2,:),80,80) - dwmin) .* 255 ./ (dwmax-dwmin);

            subplot(2,2,1);
            image(w1);
            subplot(2,2,2);
            image(w2);

            subplot(2,2,3);
            image(dw1);
            subplot(2,2,4);
            image(dw2);

        elseif SAR_or_HRR == 2 %Display weights for HRR training

            subplot(2,1,1);
            plot(w(1,:));

            subplot(2,1,2);
            plot(w(2,:));
        end

    elseif Dy == 3

        if SAR_or_HRR == 1 %Display weights for SAR training
            colormap(target_graymap);

            %Reshape weights into an 80x80 image and shift them to a 255-grayscale
            w1 = (reshape(w(1,:),80,80) - wmin) .* 255 ./ (wmax-dwmin);
            w2 = (reshape(w(2,:),80,80) - wmin) .* 255 ./ (wmax-dwmin);
            w3 = (reshape(w(3,:),80,80) - wmin) .* 255 ./ (wmax-dwmin);
            dw1 = (reshape(dw(1,:),80,80) - dwmin) .* 255 ./ (dwmax-dwmin);
            dw2 = (reshape(dw(2,:),80,80) - dwmin) .* 255 ./ (dwmax-dwmin);
            dw3 = (reshape(dw(3,:),80,80) - dwmin) .* 255 ./ (dwmax-dwmin);

            subplot(2,3,1);
            image(w1);
            subplot(2,3,2);
            image(w2);
            subplot(2,3,3);
            image(w3);

            subplot(2,3,4);
            image(dw1);
            subplot(2,3,5);
            image(dw2);
            subplot(2,3,6);
            image(dw3);

        elseif SAR_or_HRR == 2 %Display weights for HRR training

```

```

        subplot(3,1,1);
        plot(w(1,:));
        subplot(3,1,2);
        plot(w(2,:));
        subplot(3,1,3);
        plot(w(3,:));

        end
    end

    %axis equal;

end

nil_return = 0;
return;

```

A.2 Mutual Information Network Testing

```

%=====
%
% FILE      : test_SAR_2DOF.m
%
% TYPE      : Matlab program
%
% AUTHOR    : Original 1-DOF program done in PV-Wave by Dongxin Xu, Computational
%             NeuroEngineering Lab, Univ of Florida. Copy of original obtained
%             by Capt Andrew Learn on 24 April 1998. Translated to Matlab and
%             modified extensively by Capt Learn for thesis research at the
%             Air Force Institute of Technology.
%
% PURPOSE   : Tests pose estimation using the Mutual Information (MI) method
%             for SAR ground targets from the MSTAR dataset. Predicts both
%             azimuth and elevation of test targets.
%
% PARAMETERS: No external arguments. Set parameters below.
%
% NOTES    :
%
% LAST MODIFIED : 25 Jan 99
%
%=====
%
%=====
%!!!!!!!!!!!!!!!!!!!! SET CONTROL PARAMETERS !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
%=====
%
% Set debug flag for debug comments/figures (1=on, 0=off)
debug = 0;
%
% Set elev representation (0=none/1DOF; 1=conc circles/2DOF; 2=cylinder/2DOF)
elev_rep = 2;
%
% Choose file name of trained network to be used; Load trained network
%
%load SARnetworks/mi_lin_2s1_180_net_cyl;
load SARnetworks/mi_lin_2s1_180_net_cyl2;
%load SARnetworks/mse_lin_2dof_2s1_180_net_cyl;
%
%When testing on MSE-trained networks, uncomment the following 2 lines:
%pose = [aspect elevation];
%pose_points = aspect_points;

```

```

%Choose how many different files each test target has
num_targets = 3;

%Choose file names of testing data to be read
testdata_filename(1,:) = 'SARdata/zsu234_15deg_test.dat ';
testdata_filename(2,:) = 'SARdata/zsu234_30deg_test.dat ';
testdata_filename(3,:) = 'SARdata/zsu234_45deg_test.dat ';

% Choose range of angles to be tested
test_angle_range = 360;

% Set a target grayscale colormap: Pixelvalue=0 =>white; Pixelvalue=255 =>black
maprange = 0 : 255;
maprange = maprange / 255;
target_graymap = [maprange' maprange' maprange'];

% Use elevation representation to set number of output nodes from network
if elev_rep <= 1
    Dy = 2;
elseif elev_rep == 2
    Dy = 3;
end

%=====
% READ TESTING DATA FROM FILE
%=====

%Convert the name matrix to a cell array (removes blank spaces)
testdata_filename = cellstr(testdata_filename);

for filenum = 1 : num_targets

    clear images aspect_true elevation_true;

    % Open testing data file
    fid_test = fopen(char(testdata_filename(filenum)), 'rb', 'ieee-be');
    disp(fid_test);

    % Read the number of images in the testing file
    N_test = fread(fid_test, 1, 'ushort');

    % Loop through testing images
    disp('Reading testing images');
    for i = 1 : N_test

        % Print the number of the current image being read
        disp(['Image number = ' num2str(i)]);

        % Read the image into a variable
        images(:,i) = fread(fid_test, 6400, 'uchar');
    end

    % Similarly, read all aspect angles
    disp('Reading aspect angles');
    for i = 1 : N_test

        % Print the number of the current aspect angle being read
        disp(['Aspect number = ' num2str(i)]);

        % Read the image into a variable
        aspect_true(i,1) = fread(fid_test, 1, 'float');
    end

    % Similarly, read all elevation angles
    disp('Reading elevation angles');
    for i = 1 : N_test

```

```

    % Print the number of the current elevation angle being read
    disp(['Elevation number = ' num2str(i)]);

    % Read the image into a variable
    elevation_true(i,1) = fread(fid_test, 1, 'float');
end

%For multiple files, use temp variable to accumulate additional images
if filenum == 1
    temp_images = images;
    temp_aspect_true = aspect_true;
    temp_elevation_true = elevation_true;
else
    temp_images = [temp_images images];
    temp_aspect_true = [temp_aspect_true ; aspect_true];
    temp_elevation_true = [temp_elevation_true ; elevation_true];
end

% Close testing data file
fclose(fid_test);
end

%Rename variables to include multiple targets
images = temp_images;
aspect_true = temp_aspect_true;
elevation_true = temp_elevation_true;

N_test = length(aspect_true);

%=====
% If testing on 360 degrees but network was trained on 180, do mod on angles
%=====

if (test_angle_range == 360) & (train_angle_range == 180)
    aspect_true = mod (aspect_true, 180);
end

%=====
% Calculate the output of the network for all test images
%=====

y_test = simulin(images, weights);

%Calculate number of training images for later use
N_train = size(y_train, 2);

%=====
% Prepare 6 display windows
%=====

%Position vector = [xposition yposition width height]

%Figure 1 displays trained vs. test network outputs
figure(1);
set(1,'position',[720 450 400 450]);

%Figure 2 displays network weights
figure(2);
set(2,'position',[720 200 450 200]);

%Figure 3 shows training network outputs
figure(3);
set(3,'position',[0 500 370 330]);

%Figure 4 shows testing images
figure(4);
set(4,'position',[0 30 450 500]);

```

```

%Figure 5 shows true azimuth vs. estimate
figure(5);
set(5,'position',[380 500 330 330]);

%Figure 6 shows true elevation vs. estimate
if elev_rep == 2
    figure(6);
    set(6,'position',[380 50 400 250]);
end

%=====
% Display network weights
%=====

figure(2);
set(2,'PaperPosition',[.25 1 7 3]); %Ensures proper dimensions for printing
colormap(target_graymap);

wmax = max(max(weights));
wmin = min(min(weights));

w1 = (reshape(weights(1,:),80,80) - wmin) .* 255 ./ (wmax - wmin);
w2 = (reshape(weights(2,:),80,80) - wmin) .* 255 ./ (wmax - wmin);
w3 = (reshape(weights(3,:),80,80) - wmin) .* 255 ./ (wmax - wmin);

subplot(1,3,1);
image(w1);
title('Weights to Output y1')

subplot(1,3,2);
image(w2);
title('Weights to Output y2')

subplot(1,3,3);
image(w3);
title('Weights to Output y3')

%=====
% Display training image network outputs
%=====

figure(3);

%Plot training output values
plot (pose(:,1), y_train(1,:), 'c');
hold on;
plot (pose(:,2), y_train(2,:), 'r');
title('Value of trained network outputs');
xlabel('Aspect Angle');
ylabel('Output Value');
legend('Y1', 'Y2');
hold off;

%=====
% Display testing vs. training outputs
%=====

xmin = min( y_train(1,:) ) - 0.1;
xmax = max( y_train(1,:) ) + 0.1;
ymin = min( y_train(2,:) ) - 0.1;
ymax = max( y_train(2,:) ) + 0.1;
zmin = min( y_train(3,:) ) - 0.1;
zmax = max( y_train(3,:) ) + 0.1;

figure(1);
set(1,'PaperPosition',[1 1 3.9 3.9]); %Ensures proper dimensions for printing

```

```

%Plot training output values
plot3 (y_train(1,:), y_train(2,:), y_train(3,:), 'ko');
axis([xmin xmax ymin ymax zmin zmax]);
hold on;

%Make beginning and end of circle different color
%plot (y_train(1,1:5), y_train(2,1:5), 'go-');
%plot (y_train(1,N_train-4:N_train), y_train(2,N_train-4:N_train), 'ro-');

%Plot testing output values
%plot3 (y_test(1,1:173), y_test(2,1:173), y_test(3,1:173), 'k^-');
%plot3 (y_test(1,174:360), y_test(2,174:360), y_test(3,174:360), 'ks-');
%plot3 (y_test(1,361:560), y_test(2,361:560), y_test(3,361:560), 'kd-');
plot3 (y_test(1,1:173), y_test(2,1:173), y_test(3,1:173), 'k^-');
plot3 (y_test(1,174:358), y_test(2,174:358), y_test(3,174:358), 'ks-');
plot3 (y_test(1,361:538), y_test(2,361:538), y_test(3,361:538), 'kd-');
legend('Train','Test-15deg','Test-30deg','Test-45deg',2);

%Make beginning and end of circle different color
%plot (y_test(1,1:5), y_test(2,1:5), 'g^-');
%plot (y_test(1,N_test-4:N_test), y_test(2,N_test-4:N_test), 'r^-');

xlabel('y1');
ylabel('y2');
zlabel('y3');

grid;

%=====
% Display testing images
%=====
%This displays only 56 of the testing images. It selects samples evenly from
%throughout the testing angles, just to give a visual of the testing range.

%Set current window and colormap
figure(4);
colormap(target_graymap);

%Choose the index interval between images to be displayed
index_step = N_test ./ 56;

%If fewer than 56 test images, show them all
if N_test < 56
    for i = 1 : N_test,
        %Change image from column vector to matrix
        target = reshape(images(:,i), 80,80);
        %Plot image
        subplot(8,7,i);
        image(target);
        axis('off');
    end
else
    %If more than 56 test images, show a sample of them
    for i = 1 : 56,
        %Choose index of image to plot
        image_index = floor(index_step .* i);
        %Change image from column vector to matrix
        target = reshape(images(:,image_index), 80,80);
        %Plot image
        subplot(8,7,i);
        image(target);
        axis('off');
    end
end
end

```

```

%=====
% Pose estimation
%=====

%Set finest resolution of angle step to be used for pose estimation
angle_step = 3;

%Create vector of angles 0-360 deg with this angle step
angle_vector = 0 : angle_step : 360;
N_angles = length(angle_vector);

%Convert degrees to radians
angle_vector_radians = angle_vector .* pi ./ 180;

%Create angle_points which gives (x,y) coordinates of each angle on the 2-D
%unit circle in column vector format.
angle_points(1,:) = cos( angle_vector_radians );
angle_points(2,:) = sin( angle_vector_radians );

%For 2DOF, change angle_points to concentric circle representation
if elev_rep == 2 %cylinder

    %Set finest resolution of elevation angle step for elevation estimation
    elev_step = 3;

    num_circles = floor((45 - 15)/elev_step);
    temp_angle_points = [angle_points; zeros(1,N_angles)];
    temp_angle_vector = angle_vector;
    temp_elev_vector = ones(1,N_angles) * 15;
    for i = 1 : num_circles
        next_circle = [angle_points; ones(1,N_angles)*(i*elev_step)/10];
        temp_angle_points = [temp_angle_points next_circle];
        temp_angle_vector = [temp_angle_vector angle_vector];
        temp_elev_vector = [temp_elev_vector ones(1,N_angles)*(15+i*elev_step)];
    end
    angle_points = temp_angle_points;
    angle_vector = temp_angle_vector;
    elev_vector = temp_elev_vector;
    N_angles = length(angle_vector);

    figure;
    plot3(angle_points(1,:), angle_points(2,:), angle_points(3,:), 'ko');

end

%Expand angle_points and pose_points (training) into matrices
%of size [N_angles x N_train]
for i = 1 : N_angles
    if mod(i,100)==0 disp(i); end
    angle_grid1(:,i,:) = pose_points;
end

for i = 1 : N_train
    if mod(i,100)==0 disp(i); end
    angle_grid2(:, :, i) = angle_points;
end

%Calculate distances: angle_gridx = distance between x-coordinates of
%angles and aspects. Column1 (length=721) of angle_gridx gives x-distance
%between aspect point 1 and each angle point.

angle_grid = angle_grid2 - angle_grid1;

%Note: angle_grid ^ 2 = squared DISTANCE measure.
%Row1 would be squared distance between angle point 1 and all training
%aspect points.
%Thus: when used to make Gaussians below, it is equivalent to placing

```

```

%a Gaussian on each angle point and measuring the height of the
%Gaussian at each training aspect. This yields a height vector with
%a particular shape for each potential angle point. With angle step
%equal to .5, this yields matrix angle_grid with 721 rows, each row
%being the height vector which results from that particular angle point.

angle_grid = exp( -(sum((angle_grid.^2),1)) ./ (4 .* sigma2_aspect) );

%Compress back down to 2 dimensions
angle_grid = squeeze(angle_grid);

%=====
% Loop through all test points and find the pose estimate
%=====

for i = 1 : N_test

    if mod(i,10)==0 disp(i); end

    %Duplicate the outputs for the current test point N_train times
    y_test_i = [ones(1,N_train).*y_test(1,i) ; ones(1,N_train).*y_test(2,i); ...
                ones(1,N_train).*y_test(3,i)];

    %Calculate the distances between the test point and the training points
    test_distance = y_train - y_test_i;

    %Make Gaussian; gives height vector of Gaussian at each training point
    test_distance = exp( - sum(test_distance.^2) ./ (sigma2_y .* 4));

    %Multiply angle_grid * distance to find the match vector which indicates
    %how well each angle's height vector matches the test point height vector.
    %The max of the match vector is the pose estimate.
    % [721 x N_train] * [N_train x 1] = [721 x 1]

    angle_match = angle_grid * test_distance';

    if debug == 1
        figure(7);
        plot(test_distance);
        title(['Height vector of test point ' num2str(i)]);
        xlabel ('Training Point');
        ylabel ('Height of test pt Gaussian measured at training pts');
        pause(.5);

        figure(8);
        plot(angle_match);
        title(['Match vector for test point ' num2str(i)]);
    end

    [dummy max_index] = max(angle_match);
    pose_estimate(i) = angle_vector(max_index);

    if elev_rep == 2
        elevation_estimate(i) = elev_vector(max_index);
    end
end

%=====
% Readjust pose estimate if training was done just for 0-180
%=====

if train_angle_range == 180
    pose_estimate = pose_estimate ./ 2;
end

```

```

%=====
% Plot true pose and pose estimate
%=====

figure(5);
set(5,'PaperPosition',[.25 1 7 3.5]); %Ensures proper dimensions for printing

index = [1:N_test];
plot(index, aspect_true, 'ko');
hold on;
plot(index, pose_estimate, 'kx');
legend('True', 'Estimate', 4);
%plot(index, aspect_true, 'k');
plot(index, pose_estimate, 'k');
xlabel('Image number');
ylabel('Azimuth');
grid;

%Plot true elevation vs. estimate
figure(6);
set(6,'PaperPosition',[.25 1 7 3.5]); %Ensures proper dimensions for printing
index = [1:N_test];
plot(index, elevation_true, 'ko');
hold on;
plot(index, elevation_estimate, 'kx');
legend('True', 'Estimate', 4);
%plot(index, elevation_true, 'k');
plot(index, elevation_estimate, 'k');
xlabel('Image number');
ylabel('Elevation');
grid;

%=====
% Calculate error statistics
%=====

%Calculate azimuth errors
error1 = abs(pose_estimate - aspect_true');
if train_angle_range == 180
    error2 = abs(180 - error1);
elseif train_angle_range == 360
    error2 = abs(360 - error1);
end

error = min(error1, error2);
error_deviation = std(error);
error_max = max(error);
error_mean = mean(error);

disp(['Max error = ' num2str(error_max)]);
disp(['Mean error = ' num2str(error_mean)]);
disp(['Std deviation of error = ' num2str(error_deviation)]);

%Calculate elevation errors
elevation_error = abs(elevation_estimate - elevation_true');
elevation_error_deviation = std(elevation_error);
elevation_error_max = max(elevation_error);
elevation_error_mean = mean(elevation_error);

disp(['Max elevation_error = ' num2str(elevation_error_max)]);
disp(['Mean elevation_error = ' num2str(elevation_error_mean)]);
disp(['Std deviation of elevation_error=' num2str(elevation_error_deviation)]);

```

Bibliography

- Agarwal, 1998. Agarwal, S., Chaudhuri, S. "Determination of Aircraft Orientation for a Vision-Based System Using Artificial Neural Networks," *Journal of Mathematical Imaging and Vision*, 8/3:255-69 (May 1998).
- Bishop, 1995. Bishop, C.M. *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.
- Casasent, 1997. Casasent, D., Shenoy, R. "Feature Space Trajectory for Distorted-Object Classification and Pose Estimation in Synthetic Aperture Radar," *Optical Engineering*, 36/10:2719-28 (October 1997).
- Casasent, 1998. Casasent, D., Neiberg, L.M., Sipe, M.A. "Feature Space Trajectory Distorted Object Representation for Classification and Pose Estimation," *Optical Engineering*, 37/3:914-23 (March 1998).
- Cyganski, 1997. Cyganski, D., Vaz, R.F. *An Information Theoretic Investigation of Automatic Object Recognition and Image Fusion: Final Report, 25 September 1995 - 24 May 1997*. Contract DAAH 04-95-1-0631. Worcester Polytechnic Institute, 24 August 1997 (AD-A332432).
- DARPA, 1997. *MSTAR (Public) Targets CD-ROM*, DARPA product number DAA7B02AA, Veda Inc. 1997.
- DARPA, 1998a. *MSTAR/IU (Public) Mixed Targets CD-ROM*, DARPA product number DLC8B02AA and DLD8B02AA, Veda Inc. 1998.
- DARPA, 1998b. *Public Targets HRR 17 Degree Train Data CD & Public Targets HRR 15 Degree Test Data CD*, DARPA Target Recognition Using Mathematical and Physical Exploitation of Target Signatures (TRUMPETS) Program, Veridian Inc., 1998.
- Dudgeon, 1993. Dudgeon, D., Lacoss, R. "An Overview of Automatic Target Recognition," *The Lincoln Laboratory Journal*, vol. 6 no. 1, 1993.
- Dudgeon, 1994. Dudgeon, D., and others. "Use of Persistent Scatterers for Model-Based Recognition," *SPIE*, 2230:356-68 (1994).
- Keydel, 1996. Keydel, E.R., Lee, S.W., Moore, J.T. "MSTAR Extended Operating Conditions: A Tutorial," *SPIE*, 2757:228-42 (1996).
- Libby, 1996. Libby, E.W., Maybeck, P.S. "Sequence Comparison Techniques for Multisensor Data Fusion and Target Recognition," *IEEE Transactions on Aerospace and Electronic Systems*, 32/1:52-64 (January 1996).

- Mathworks, 1998. *MATLAB Neural Networks Toolbox On-line Manual*, Web site www.mathworks.com/access/helpdesk/help/pdf_doc/nnet/nnet.pdf, Mathworks Inc., 18 Dec 1998.
- MSTAR, 1998. "MSTAR Overview," web document at www.mstar.vdl-atr.af.mil, AFRL Virtual Distributed Laboratory for Automatic Target Recognition, 6 December 1998.
- Principe, 1998a. Principe, J.C. *Feature Extraction Using an Information Theoretic Framework: Yearly Report*. Contract F33615-97-1019. University of Florida, 1998.
- Principe, 1998b. Principe, J.C., Xu, D., Fisher, J. "Pose Estimation in SAR Using an Information Theoretic Criterion," SPIE, 3370:218-29 (1998).
- Principe, 1998c. Principe, J.C. "Statistically Independent Feature Extraction for SAR Imagery," *Image Understanding Workshop*, Pacific Grove, Canada, 1998.
- USAF, 1996. "Global Engagement: A Vision of the 21st Century Air Force," Headquarters Air Force Strategic Planning Directorate, 1996.
- Willis, 1991. Willis, D.J. *Feature Extraction for Pose Estimation*. MS thesis, Air Force Institute of Technology, Wright-Patterson AFB, OH, 1991.
- Xu, 1998a. Xu, D., Principe, J.C. "Learning from Examples with Mutual Information," *Proceedings of the IEEE, Neural Networks for Signal Processing (NNSP) Workshop*, p 155-64, Aug 1998.
- Xu, 1998b. Xu, D., Principe, J.C. "Training MLPs Layer-by-Layer with the Information Potential," draft article at website www.cnel.ufl.edu, 1998.

Vita

Captain Andrew W. Learn was born on 20 September 1967 in Indiana, Pennsylvania. He grew up on his parents' dairy farm and graduated from Purchase Line High School in 1985. He attended the Pennsylvania State University and received his bachelor's degree in aerospace engineering in 1989.

He began his career as a systems engineer for General Dynamics Space Systems Division in San Diego, California. After more than three years working on the Atlas and Titan launch vehicles, he applied and was accepted for Air Force Officer Training School. He was commissioned as a communications officer in January 1994.

Following basic communications officer training in Mississippi, Captain Learn was assigned to the 89th Communications Group, Andrews Air Force Base, Maryland. There he served in turn as officer-in-charge of the Presidential communications software branch, executive officer to the group commander, and network projects officer for the base network control center. During these assignments, Captain Learn supervised the design and implementation of several major communications and computer technology upgrades at the group, wing, and base level, for which he received the Air Force Commendation Medal.

In 1997 Captain Learn was selected for the master's program at the Air Force Institute of Technology. He will graduate in March 1999 with a master's degree in computer systems. His follow-on assignment is to the Automatic Target Recognition branch in the Sensors Directorate of the Air Force Research Laboratory, Wright-Patterson Air Force Base, Ohio.

Captain Learn is married to the former Patricia Crabtree (a proud Air Force brat). They have three children, Megan, Christopher, and Joshua.

Permanent address:

353 Blossom Hollow Road

Commodore, PA 15729-8903