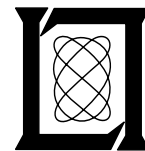


Mode S Surveillance Netting

J. L. Gertz

4 November 1983

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS



Prepared for the Federal Aviation Administration,
Washington, D.C. 20591

This document is available to the public through
the National Technical Information Service,
Springfield, VA 22161

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

1. Report No. FAA-RD-DOT/FAA/PM-83/17	2. Government Accession No.	3. Recipient's Catalog No.	
4. Title and Subtitle Mode S Surveillance Netting		5. Report Date 4 November 1983	
		6. Performing Organization Code	
7. Author(s) Jeffrey L. Gertz		8. Performing Organization Report No. ATC-120	
9. Performing Organization Name and Address Lincoln Laboratory, M.I.T. P.O. Box 73 Lexington, MA 02173-0073		10. Work Unit No. (TRAIS)	
		11. Contract or Grant No. DOT-FA72WAI-261	
12. Sponsoring Agency Name and Address Department of Transportation Federal Aviation Administration Systems Research and Development Service Washington, D.C. 20591		13. Type of Report and Period Covered Project Report	
		14. Sponsoring Agency Code	
15. Supplementary Notes The work reported in this document was performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology, under Air Force Contract F19628-80-C-0002.			
16. Abstract The surveillance performance of a single Mode S Sensor is degraded by several factors, including: poor crossrange accuracy at long range, diffraction-induced azimuth errors, missing or incomplete reports, and extraneous reports. The surveillance netting project reported here sought to overcome these difficulties by employing information from a secondary (and perhaps also a tertiary) sensor. The project was performed to determine what auxiliary information is most useful, how this information could be used for maximum effect, when help should be sought from other sensors, what form this inter-sensor communication should take, and where the netting algorithms should be implemented. It was also planned to include the construction of a real-time netting demonstration system to exercise and test the concepts developed. The central issue in this project was the approach to be used for multi-sensor azimuth determination. In particular, a new form of incremental bilateration, employing a flat earth model, is shown to be both accurate and bias-resistant. Altitude estimation methods and multi-sensor tracker design are also addressed, with new algorithms developed in each case. Finally, the design of the netting subsystem for a Mode S sensor is presented.			
17. Key Words Mode S ATCRBS Netting Multilateration Aircraft surveillance		18. Distribution Statement Document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.	
19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 228	22. Price

CONTENTS

1.0	STUDY DEFINITION AND OVERVIEW	1
1.1	Netting Concept	1
1.2	Need for Netting	12
1.3	Project Objectives	13
1.4	Report Overview	14
2.0	DATA CHARACTERISTICS	16
2.1	Azimuth Inaccuracies	16
2.2	Reflection False Targets	20
2.3	Sensor and Aircraft Biases	20
2.4	Netting Database	25
3.0	LOCATION OF NETTING ALGORITHMS	29
3.1	Centralized Netting	29
3.2	Localized Netting	31
3.3	Hybrid Netting	33
4.0	LOCALIZED NETTING SYSTEM	36
4.1	System Overview	36
4.2	Inter-Sensor Messages	38
4.3	Message Handling Data Structures	39
4.4	Netting System Algorithms	43
4.4.1	Triggering Rules	48
4.4.2	Request Generator	49
4.4.3	Response Processor	51
4.4.4	Netting Subsystem	51
4.5	Support System Algorithms	52
5.0	INTER-SENSOR CORRELATION	53
5.1	Correlation Philosophy	53
5.2	Discrete Correlation	54
5.3	Non-Discrete Correlation	56
6.0	POSITION AND HEADING IMPROVEMENT	60
6.1	Azimuth Improvement	60
6.2	Overcoming Diffraction	63
6.3	Bias Error Effects	63
6.4	Incremental Bilateralation	64
6.5	Transponder Turnaround Delay	77
6.6	Altitude Estimation	80
6.7	Smoothing Filters	84
7.0	NETTING TIMING	87
7.1	Data Frequency	87
7.2	Interpolation vs. Extrapolation	87
7.3	Timing Alternatives	90
7.4	Timing Performance Comparisons	97

CONTENTS (Continued)

8.0	MULTILATERATION WITH TWO OR THREE SENSORS	100
8.1	Earth Models	100
8.2	Two Sensors, Altitude Known, $\Delta=0$	103
8.3	Three Sensors, Altitude Known, $\Delta=0$	104
8.4	Two Sensors, Altitude Unknown, $\Delta=0$	105
8.5	Three Sensors, Altitude Unknown, $\Delta=0$	107
8.6	Two Sensors, Altitude Known, Δ Unknown	108
8.7	Three Sensors, Altitude Known, Δ Unknown	110
8.8	Two Sensors, Altitude and Δ Both Unknown	111
8.9	Three Sensors, Altitude and Δ Both Unknown	114
8.10	Simulation Results	115
8.10.1	No Biases	121
8.10.2	Transponder Delay Bias	122
8.10.3	Sensor Biases	122
8.10.4	Combined Biases	123
8.10.5	Results Summary	123
8.11	Multilateration in Diffraction Zones	123
8.12	Effect of Aspect Angle	127
9.0	FLAT EARTH INCREMENTAL BILATERATION	130
9.1	Spherical-Equivalent Flat Earth	130
9.2	Extended Flat Earth Model	138
9.3	Diffraction Zone Algorithm	141
9.4	Performance of the Extended Flat Earth Model	144
9.5	Transponder Bias Estimation	144
9.6	Altitude Estimation	151
10.0	PREDICTION FILTERS	156
10.1	Alpha-Beta Filters	156
10.2	Curve Fitting Filters	159
10.3	Standard Kalman Filter	166
10.4	Heading Kalman Filter	170
10.5	Extended Kalman Filter	176
10.6	Tracker Performance Comparisons	179
10.7	Tracker Performance in Diffraction	183
10.8	Tracker Class Comparisons	185
11.0	NON-POSITIONAL DATA IMPROVEMENT ALGORITHMS	187
11.1	Data Substitution and Code Improvement	187
11.2	False Data Suppression	188
11.3	Correlation Improvement	188
12.0	NETTING DEMONSTRATION FACILITY	191
12.1	Computer Configuration	191
12.2	Communications Channels	193
12.3	Timing Considerations	193
12.4	Netting Software	197
12.5	Netting Experiments	199

CONTENTS (Continued)

13.0 STUDY ACHIEVEMENTS	200
REFERENCES	202
APPENDIX A INTER-SENSOR REPORT CONVERSION	A-1
APPENDIX B HEADING ERROR DUE TO TRANSPONDER BIAS	B-1
APPENDIX C SENSITIVITY OF THREE-SENSOR ALTITUDE ESTIMATES	C-1
APPENDIX D FLAT EARTH INTER-SENSOR DISTANCE	D-1

ILLUSTRATIONS

1-1. Netting example scenario	3
1-2. Primary sensor data	4
1-3. Data substitution mode	5
1-4. Netting mode	6
1-5. Data substitution expanded	7
1-6. Netting expanded	8
1-7. Secondary sensor coverage	9
1-8. Mode S data accuracy	10
1-9. Netting error ellipse	11
2-1. Long range azimuth jitter	17
2-2. Loss of accuracy in diffraction zone	18
2-3. Diffraction zone performance	19
2-4. Reflection false alarm geometry	21
2-5. False target reports	22
2-6. Bias-produced data jumps	24
2-7. Simulation database trajectories	26
3-1. Centralized netting system	30
3-2. Localized netting system	32
3-3. Hybrid netting system	34
4-1. Netted Mode S sensor	37
4-2. Netting data request messages	40
4-3. Netting report messages	41
4-4. Track status array	42
4-5. Netting report buffer	44
4-6. Request storage buffer	45
4-7. Inter-sensor correlation array	46
4-8. Netting system flowchart	47
4-9. Netting coverage map	50
5-1. Discrete correlation geometry	55
5-2. Non-discrete correlation flowchart	57
6-1. Tracking improvement approaches	61
6-2. Tracker comparisons	62
6-3. Multilateration bias adjustment	65
6-4. Normal 2-sensor scenario	66
6-5. Alternate 2-sensor scenario	68
6-6. Incremental netting	69
6-7. Apparent secondary sensor motion	70
6-8. Aircraft test trajectory	71
6-9. Secondary sensor distance variation	72
6-10. Azimuth error for fixed offset	74
6-11. Azimuth error for moving offset	75
6-12. Reverse measurement geometry	76

ILLUSTRATIONS (Continued)

6-13. Reprocessed Fig. 2-6 data	78
6-14. Transponder bias estimates	79
6-15. Altitude estimation geometry	81
7-1. Inter-sensor report timing	88
7-2. Inter-sensor registration	89
7-3. Interpolation error potential	91
7-4. Extrapolation error potential	92
7-5. Extrapolation timing cases	93
7-6. Hybrid timing cases	94
7-7. Interpolation timing cases	95
8-1. Multilateration diagram	101
8-2. Aspect angle definition	128
9-1. Planar mathematics diagram	131
9-2. Sensor alignment rotation	133
9-3. Transformed sensor coordinates	134
9-4. Simplified ρ_0 calculation	137
9-5. Spherical earth calculations	139
9-6. Flat earth calculations	140
9-7. Multilateration data jumps	146
9-8. Flat earth improved data	147
9-9. Flat earth Δ computation	149
9-10. Flat earth Δ estimates	150
10-1. α , β filter	157
10-2. Two-point interpolator search box	160
10-3. Linear curve fitting	161
10-4. Quadratic curve fitting	164
10-5. Heading estimate procedure	173
10-6. Turn correction factor	174
11-1. Alternate view of garble	189
12-1. Demonstration facility architecture	192
12-2. AMPS/Eclipse link	194
12-3. Header block format	195
12-4. Data block format	196
12-5. Time synchronization report	198
A-1. Coordinate conversion geometry	A-2
B-1. Heading geometry	B-2
C-1. Three sensor altitude geometry	C-2
D-1. Inter-sensor distance geometry	D-2

TABLES

6-1	Altitude Estimation Errors, Unbiased System	83
6-2	Altitude Estimate Breakdown	85
6-3	Altitude Estimation Errors, Biased System	86
7-1	Timing Type Performance Comparisons	98
8-1	Simulation Run Parameters	116
8-2	Multilateration Accuracy, No Biases	117
8-3	Multilateration Accuracy, Transponder Bias	118
8-4	Multilateration Accuracy, Sensor Biases	119
8-5	Multilateration Accuracy, Combined Biases	120
8-6	Multilateration Accuracy, Diffraction, No. Biases	125
8-7	Multilateration Accuracy, Diffraction, Combined Biases	126
8-8	Multilateration Accuracy vs. Geometry	129
9-1	System Comparisons	145
9-2	Effects of Δ bias	152
9-3	Effects of Unknown h.	154
10-1	Summary of Continuous Discrete Extended Kalman Filter	177
10-2	Tracker Performance, 24 Aircraft Trajectories	180
10-3	Tracker Performance, Straight Trajectory	181
10-4	Tracker Performance, Turning Trajectory	182
10-5	Tracker Performance, 24 Aircraft Trajectories, Diffraction Zones	184

1.0 STUDY DEFINITION AND OVERVIEW

The FAA has endeavored to improve its ATCRBS surveillance system for many years. Lincoln Laboratory has contributed to this effort by developing the Mode S Beacon System^[1]*. Mode S will provide not only an upgraded ATC surveillance system, but enhanced performance of the present ATCRBS system as well. Several engineering model sensors incorporating Mode S have been implemented by Texas Instruments and are currently being tested at the FAA Technical Center.

Although the Mode S sensor has been shown to be capable of improving overall system performance, it cannot completely eliminate the inherent ATCRBS problems of reduced cross-range accuracy at long range, diffraction, missing reports, and extraneous reports. The surveillance netting project sought to overcome these difficulties by employing information from a secondary (and perhaps also a tertiary) sensor. The project was performed to determine what auxiliary information is most useful, how this information could be used for maximum effect, when help should be sought from other sensors, what form this inter-sensor communication should take, and where the netting algorithms should be implemented. It was also planned to include the construction of a real-time netting demonstration system to exercise and test the concepts developed.

Unfortunately, early termination of the project precluded the completion of some part of this planned effort, in particular the construction and exercise of the demonstration system. Thus, the algorithms developed during the study phase were not verified, nor their performance quantified on real data. Simulation results, however, did indicate promising performance improvement.

This document presents all work performed during the life of the project. Some areas, such as position improvement, were completed; their algorithms and results are presented in detail. Other areas, such as data editing, were only begun; their algorithms, still in the idea stage, are described to the extent possible, although neither completeness nor performance have been confirmed in these cases.

1.1 Netting Concept

Netting is defined as the simultaneous use of radar data from two or more sensors having overlapping coverage, to improve surveillance performance on aircraft in the joint region.

Multilateration, defined as the use of two or more range measurements and no azimuth measurement to determine the position of the aircraft, constitutes netting under this definition. Data substitution as practiced by NAS, on the other hand, is not netting in this sense.

*Previously designated as the Discrete Address Beacon System (DABS).

An example of the performance differences produced by these two approaches is shown graphically by Figs. 1-1 through 1-7. The first figure presents the aircraft trajectory and sensor locations assumed in this example, while Fig. 1-2 presents the target report stream generated by the primary sensor. As seen, this data stream is discontinuous due to the sensor's non-unity blip/scan ratio, as well as to positional jitter. With data substitution (Fig. 1-3), the data continuity, but not accuracy, is improved. Netting (Fig. 1-4), on the other hand, improves both aspects of surveillance. A clearer picture of the positional smoothness of the two approaches is shown in Figs. 1-5 and 1-6, where an expanded scale has been employed. Either approach, of course, covers the situation of primary sensor outage, as illustrated by the secondary sensor coverage shown in Fig. 1-7.

Single sensor surveillance of beacon-equipped aircraft uses the radar range (ρ) and azimuth (θ) in conjunction with the altitude (h) reported via the aircraft transponder to determine the three-dimensional position of the aircraft in space:

$$z = \frac{(h-h_s)^2 + 2(h-h_s)(E+h_s) - \rho^2}{2(E+h_s)} \quad (1-1)$$

$$x = \sqrt{\rho^2 - z^2} \sin \theta \quad (1-2)$$

$$y = \sqrt{\rho^2 - z^2} \cos \theta \quad (1-3)$$

where E is the radius of the earth and h_s is the height of the sensor above sea level.

For most surveillance systems, the range measurement is considerably more accurate, and possesses less variance, than the azimuth measurement. Thus, whenever two or more sensors view the target, surveillance netting schemes such as multilateration are considered. When aircraft altitude is available, only two ranges are required.

The theory underlying two sensor netting is shown pictorially in Figs. 1-8 and 1-9. In the first figure, the typical $1-\sigma$ range and azimuth measurement errors are drawn to scale at various target ranges for a Mode S sensor. As seen from this figure, the cross-range positional error grows linearly with target distance. Thus for almost all targets, range accuracy exceeds cross-range accuracy; for long range targets, the discrepancy is as much as 50 to 1.

The second figure demonstrates the degree of improvement in the measurement error ellipse that is attainable via netting. If the two sensors view the target at nearly perpendicular aspect angles, the joint error ellipse becomes symmetrical, and range and cross-range errors are then comparable.

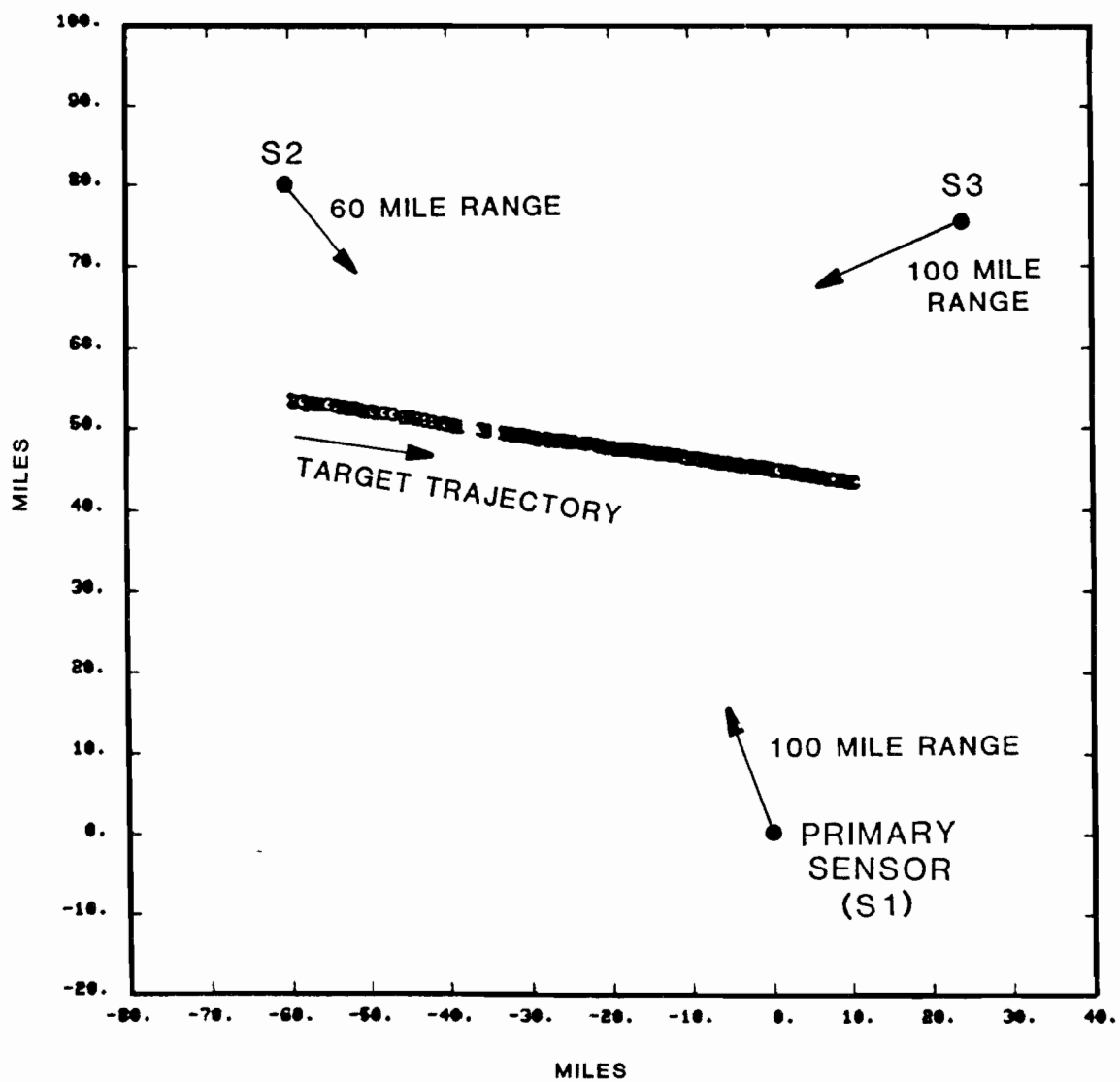


Fig. 1-1. Netting example scenario.

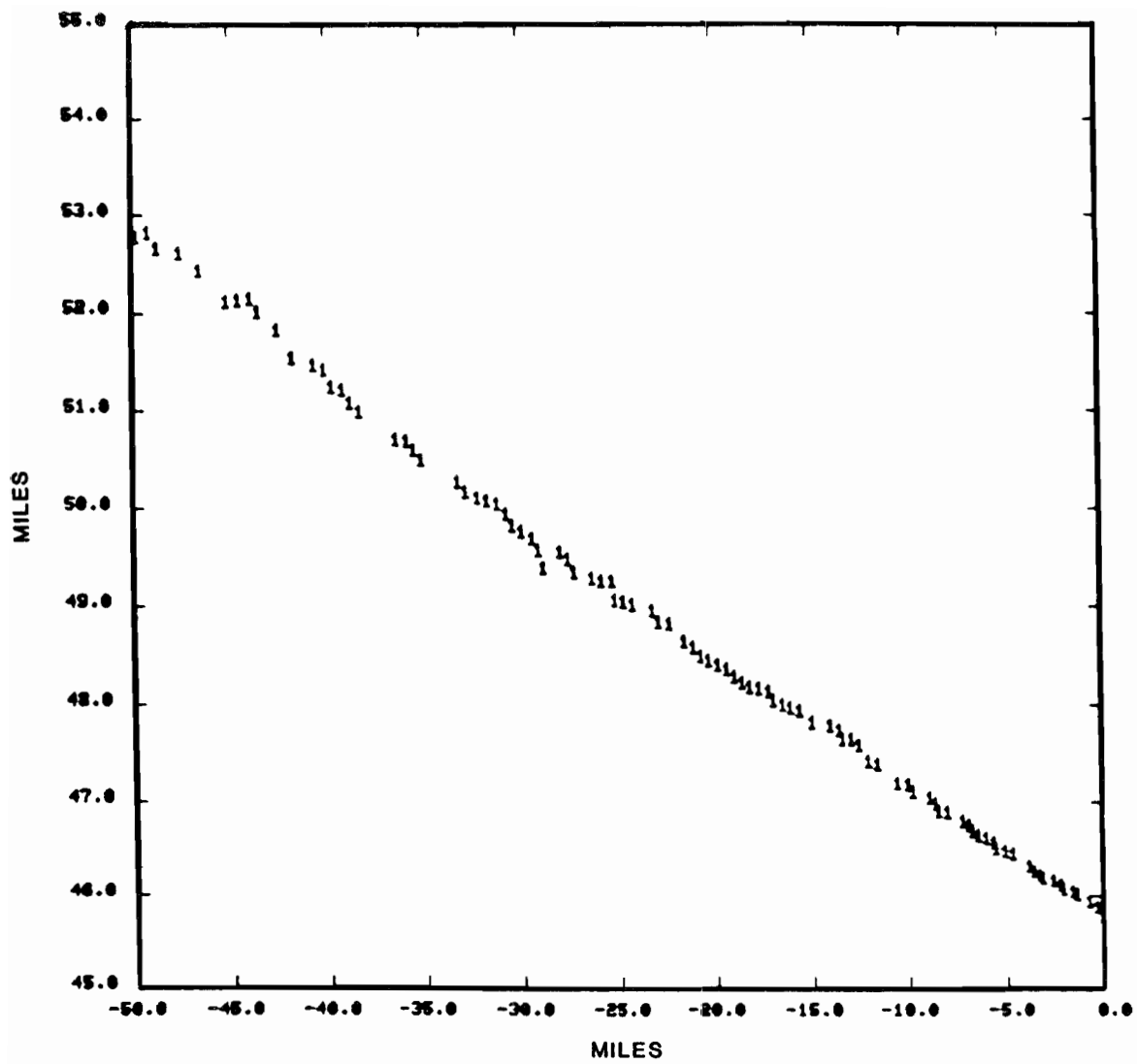


Fig. 1-2. Primary sensor data.

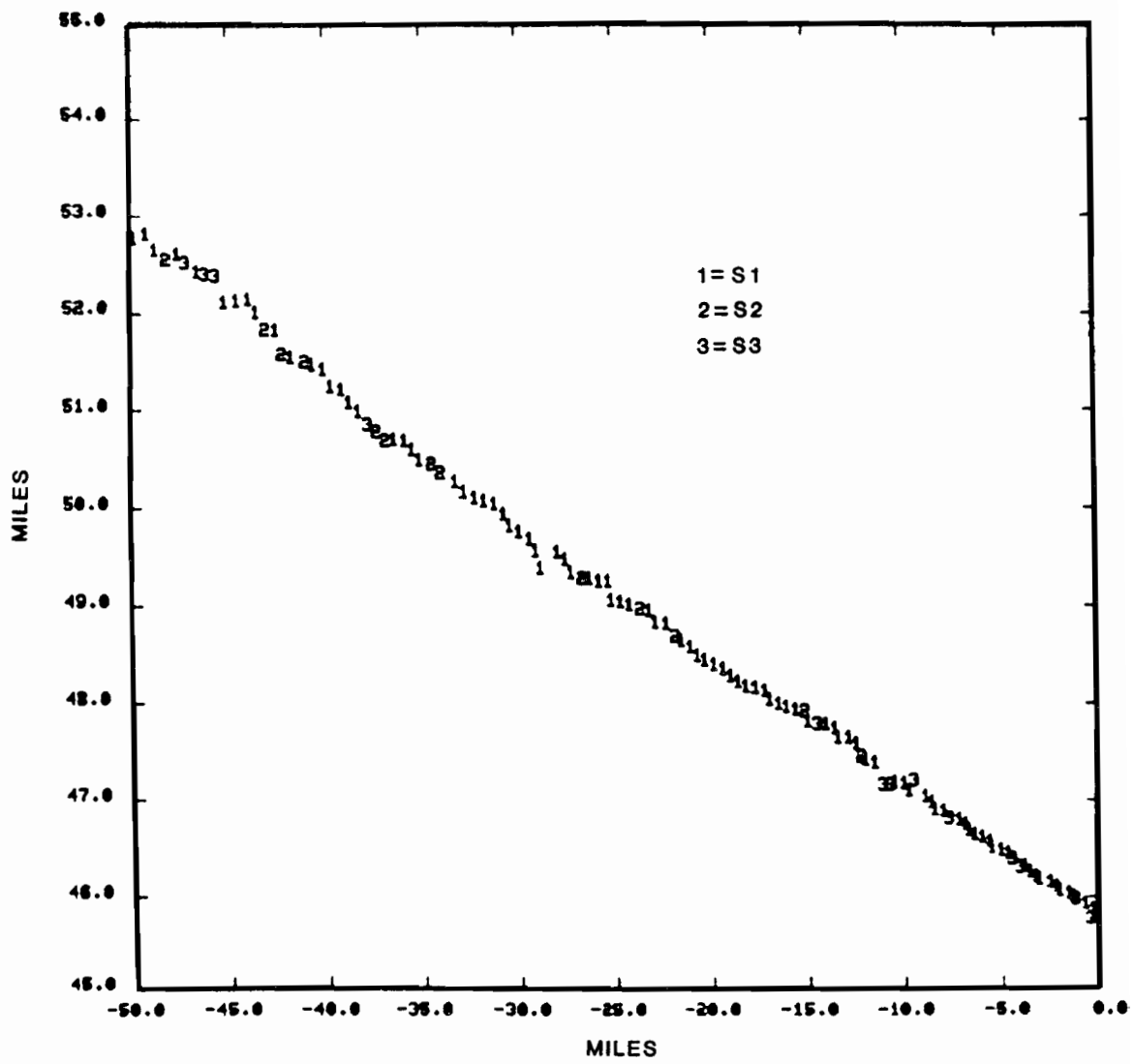


Fig. 1-3. Data substitution mode.

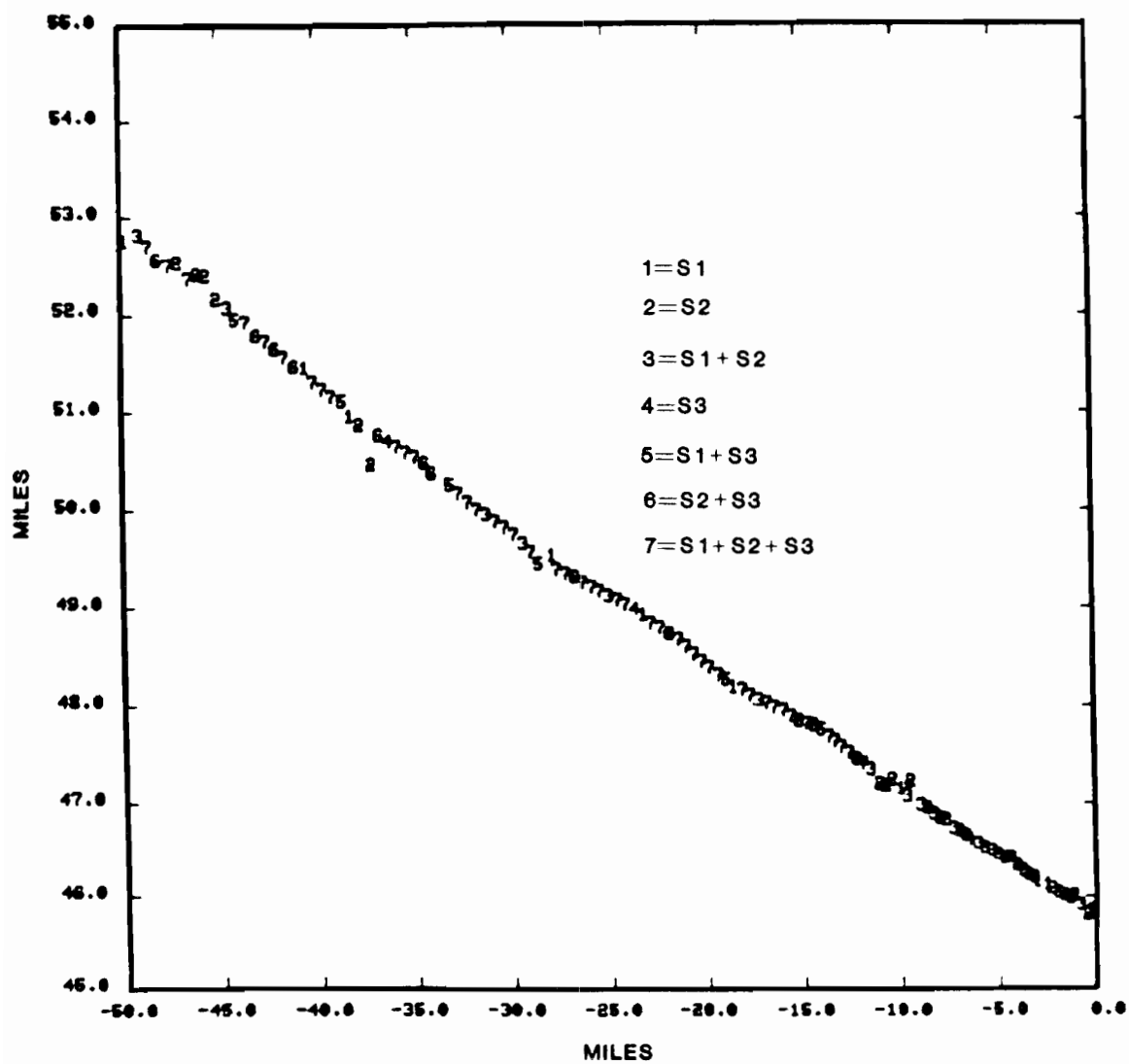


Fig. 1-4. Netting mode.

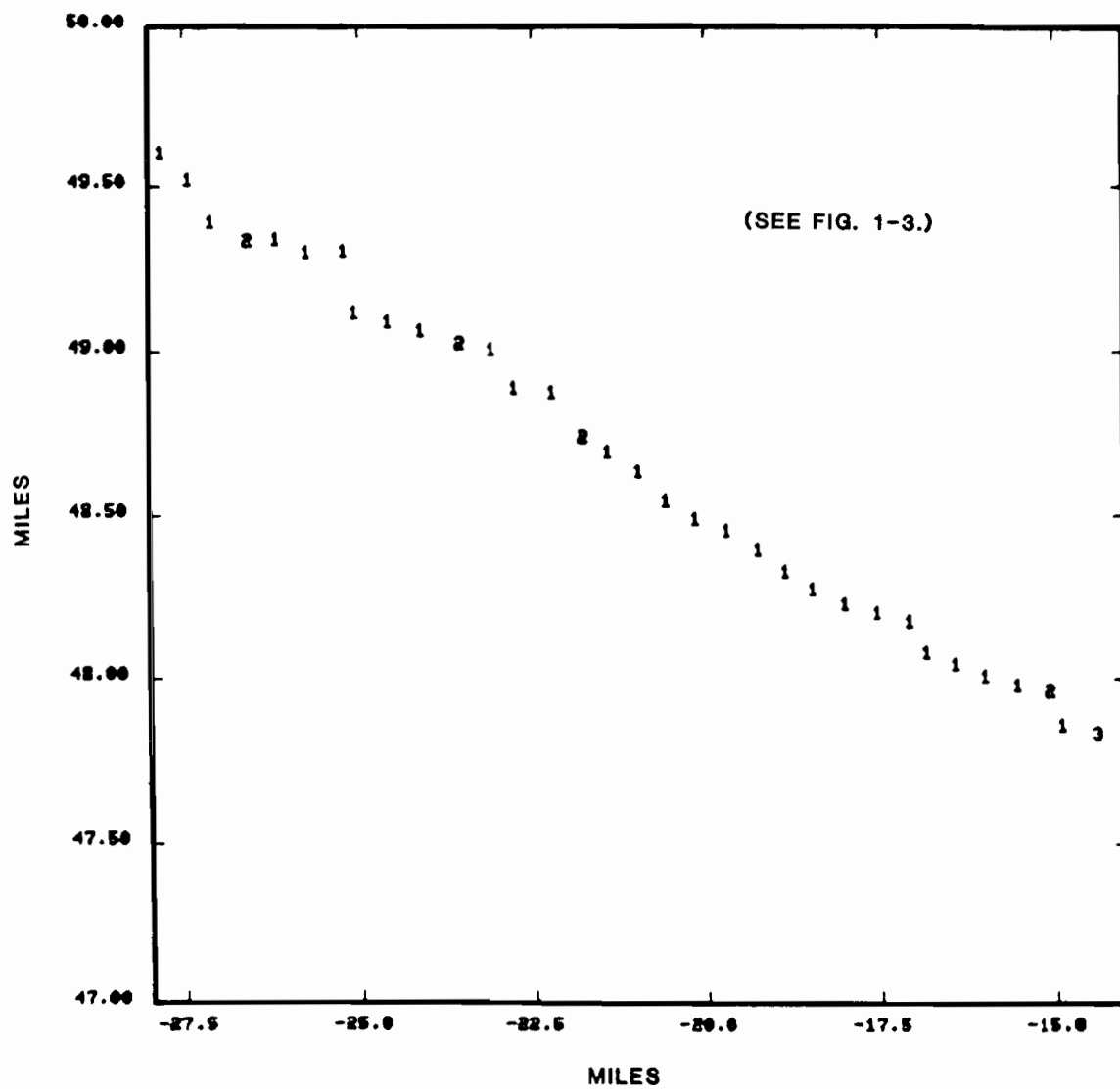


Fig. 1-5. Data substitution expanded.

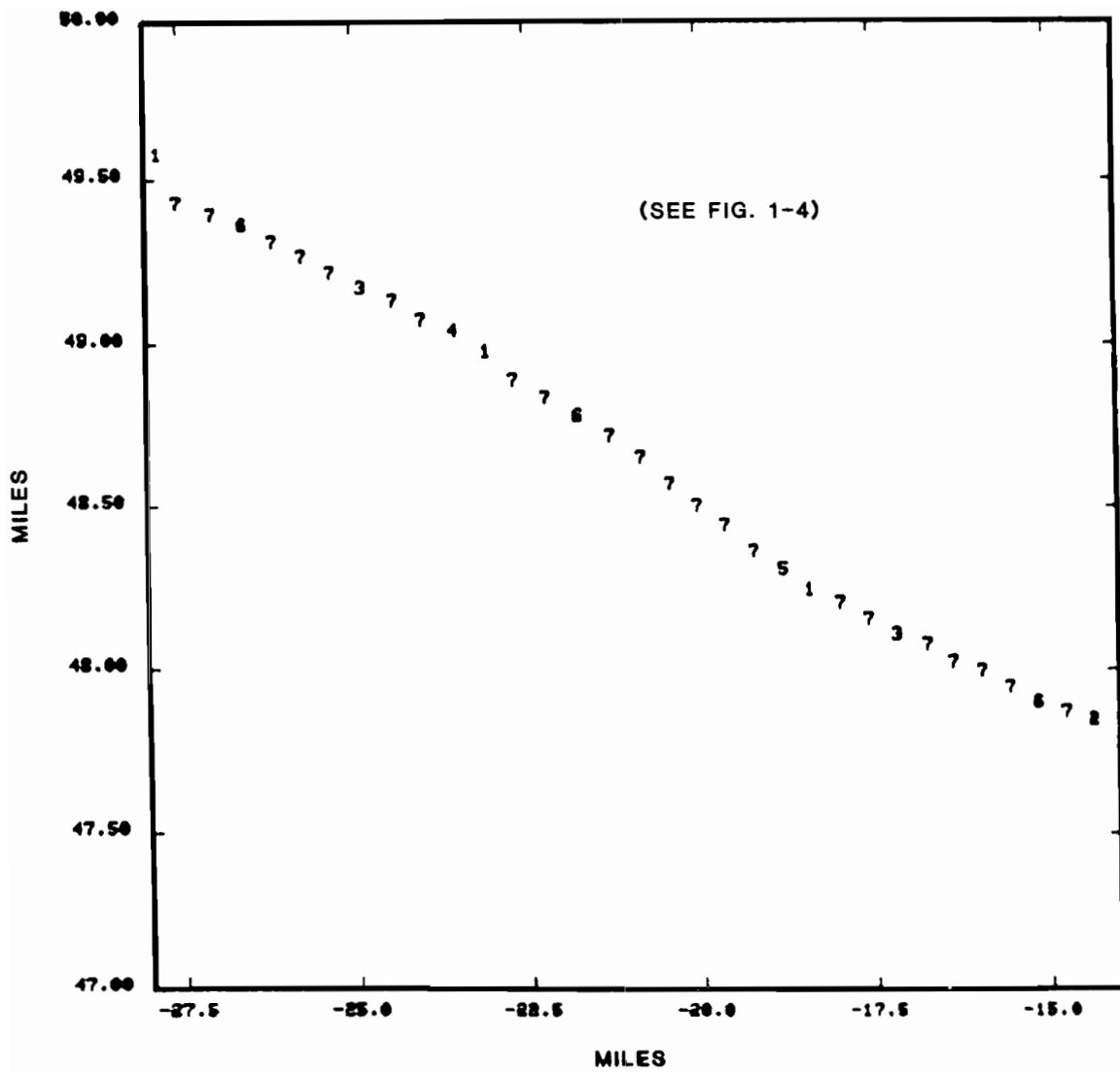


Fig. 1-6. Netting expanded.

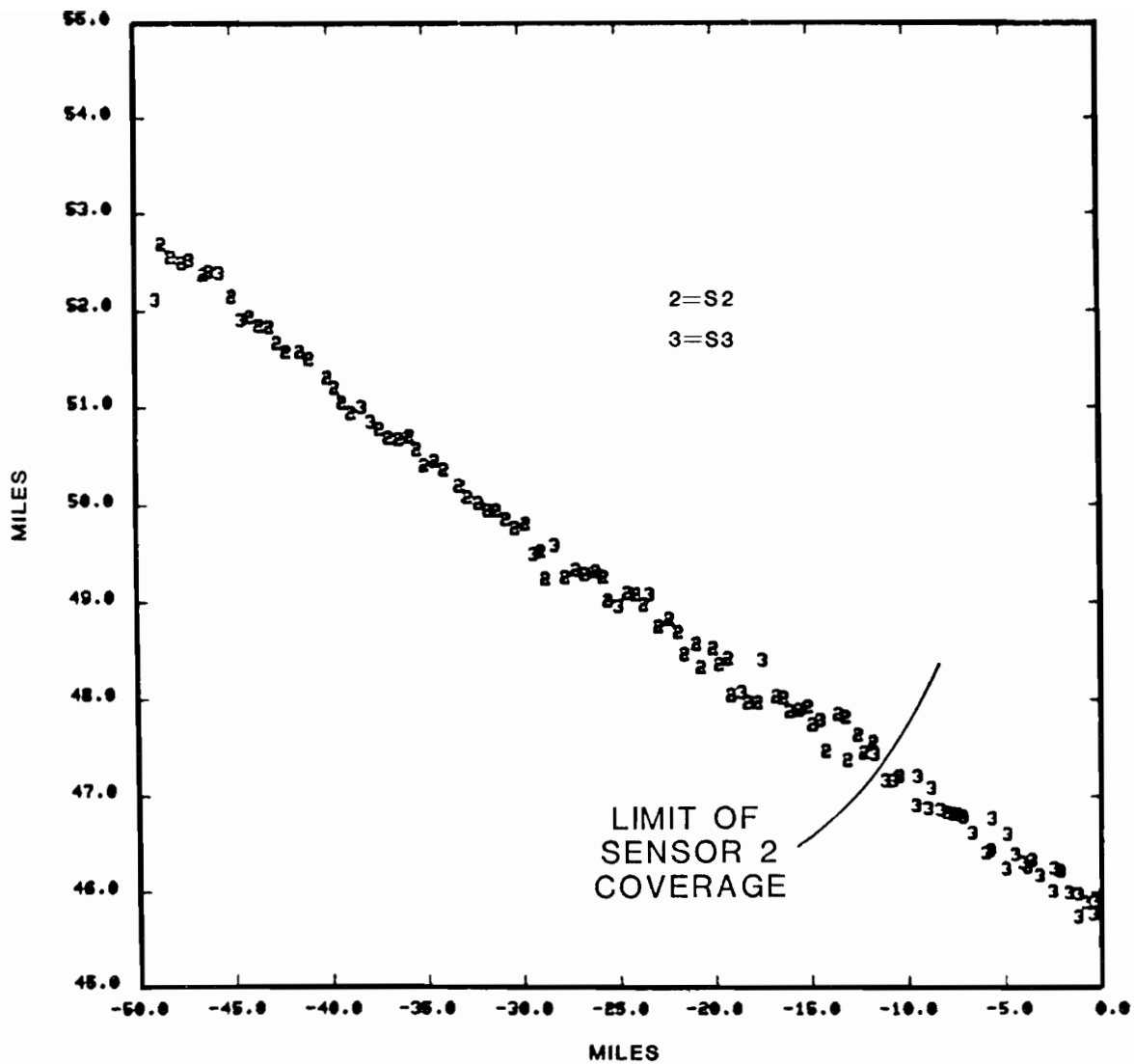


Fig. 1-7. Secondary sensor coverage.

ASSUME FOR MODE S

$$\sigma_{\rho} = 25'$$

$$\sigma_{\theta} = .05^{\circ} (\approx 1 \text{ MILLIRADIAN}) \approx \frac{6'}{\text{MILE}}$$

ERROR ELLIPSES:

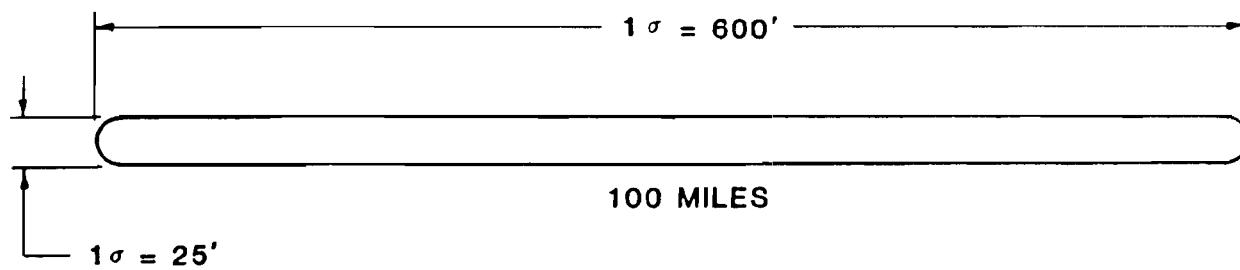
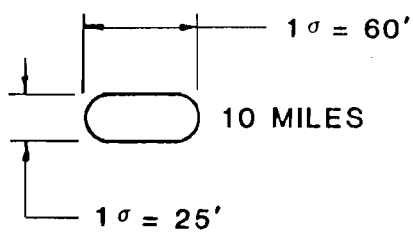
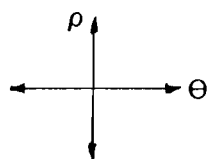


Fig. 1-8. Mode S data accuracy.

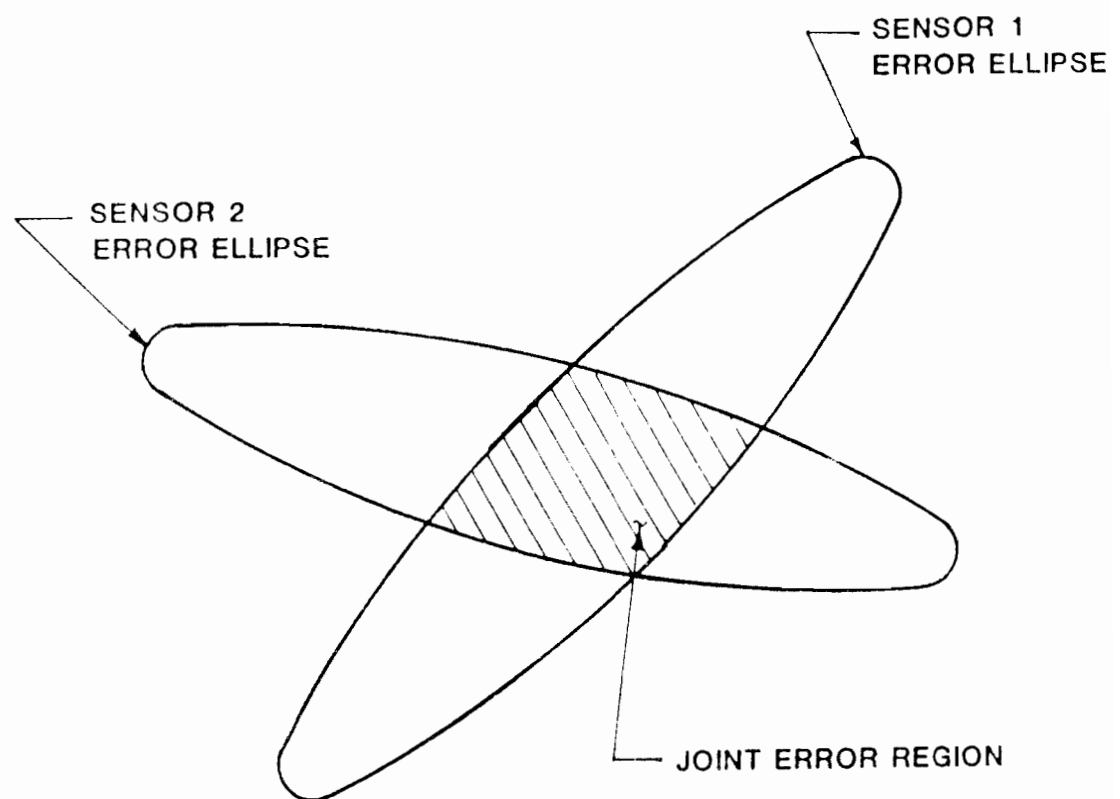


Fig. 1-9. Netting error ellipse.

This improvement leads to significantly better target tracking, and hence a better prediction of future aircraft location. With single sensor cross-range error magnitudes, the measurement jitter can easily exceed the scan-to-scan target motion. Thus, in the worst case, a target could be perceived to be heading in its opposite direction. With multiple sensor coverage, on the other hand, heading determination becomes feasible at all ranges.

In reality, however, netting fails to perform as well as this theory would indicate. Approximations to the earth's true shape reduce the accuracy of multi-sensor data. Various system biases, of minor import in a single sensor case, seriously degrade performance in netting applications. Time misalignment of measurements from the various sensors require data extrapolation. All of these factors must be addressed if netting is to be of practical use.

System biases, in particular, cause problems when secondary sensor data is to be utilized, either for substitution when primary sensor data is missing or for multilateration when it is present. In the former case, the registration errors of the alternate data could produce worse tracking errors than would have existed under coast conditions; in the latter case, the multilateration position could have less accuracy than that of the single sensor report. A previous study [2] developed an approach to overcome the substitution problem. To the author's knowledge, however, this project is the first to develop a method that eliminates bias and registration effects while simultaneously providing lower variance data.

1.2 Need for Netting

There are many problems with single sensor aircraft surveillance that can cause difficulty for users of tracked data. The most common manifestations of poor sensor surveillance, in decreasing order of importance, are:

1. unreliable azimuth in diffraction areas;
2. large azimuth variances at long range;
3. loss of data due to obstructions, cone of silence, fades, or interference;
4. extraneous reports due to reflections, correlating fruit, or other phenomena;
5. garbled, or absent, ATCRBS mode C altitude; and
6. range bias errors due to aircraft with incorrect transponder turn-around delays.

All of these effects can be alleviated by netting.

In addition, netting can increase report accuracy beyond that possible from a single sensor. For example, it can permit the estimation of altitudes of non-altimeter-equipped aircraft via multiple range measurements. Also, it can provide higher data rates due to time interlaced measurements from multiple sensors. This in particular will improve aircraft tracking for enroute sensors.

Finally, netting can improve surveillance in ways other than simply increased data accuracy. A list of such uses would include:

1. composite area-wide coverage to permit the big picture to be seen on one screen
2. smooth transitions among sensors to provide for clean handoffs
3. survive loss of a radar by permitting other sensors to cover its region

Data jumps that can occur when an aircraft is handed off from one sensor to another are troublesome to tracked users. In particular, these jumps can cause velocity and heading transients that disrupt the tracker. These jumps are caused by biases in the sensors or the aircraft itself (usually the transponder turnaround delay). Netting can alleviate the handoff problem in one of two ways. First, it can permit the biases to be computed and thus eliminated. Or second, the fact that the new primary sensor was supplying data (as the secondary sensor) prior to the handoff permits the tracker to initialize itself in the new coordinate system prior to activation. This prevents the velocity or heading discontinuities from occurring, even though a positional jump will still appear.

1.3 Project Objectives

The primary objective of the netting project was the development and implementation of algorithms for employing data from multiple sensors to overcome the limitations in Mode S sensor surveillance just described. Although range/range multilateration was expected to be the procedure to be used, other possibilities, such as higher data rates and incremental tracking, were also investigated. Surprisingly, a modified form of incremental bilateration, using a flat earth model, was found to be superior to all standard spherical earth approaches. In particular, it overcame the bias problems discussed above.

The netting project attempted both to derive theoretical algorithms to address these problems and to develop practical real-time applications based upon this theory. In particular, site-to-site data biases, non-coincident reports, and "reasonable" computer processing were all to be considered. Strategies were also to have been developed for selective utilization of these techniques in order to minimize the ground communications requirements. Thus, algorithms for triggering the request for multisite data were also required.

The major issues to have been covered in the netting study were:

1. characterization of multi-sensor data, with both expected system biases and measurement errors studied;
2. development of optimum netting algorithms, for both report and tracking improvement;
3. development of rules for both triggering netting requests and responding to them, including inter-sensor correlation;
4. determination of optimum number and location of sensors for netting;
5. construction and operation of a real-time demonstration system to prove concepts and measure performance improvements.

At the time of funding termination, the first two topics had been completed, and work was actively underway on the remaining three. The demonstration system was nearing completion. Unfortunately, since it was never put to use, no live testing of any algorithms developed in steps 2 and 3 has occurred.

1.4 Report Overview

Before netting algorithms can be developed to improve the quality and accuracy of single sensor data, the defects and characteristics of such data must be understood. Also, the nature of multi-sensor data, particularly with respect to system biases, must be explored. Chapter 2 covers both these issues. It also describes the multi-sensor simulation database that is employed throughout this report to test the expected benefits of each algorithm.

Chapters 3 through 5 then deal with the design and implementation of a netted sensor system. The first of these chapters discusses the tradeoffs between centralized and localized netting realizations. Chapter 4 then develops in detail a localized sensor implementation that includes all required functions, data structures, and communications links. Great care was taken to design a system that fit well with a standalone Mode S sensor. Finally, Chapter 5 provides solutions for the most complex netting issue, inter-sensor correlation, or the matching of a track in one sensor with that for the same aircraft in another sensor.

The heart of the netting study, surveillance accuracy improvement, is covered in Chapters 6 through 10. Chapter 6 first presents a complete overview of all the issues and algorithms involved in this area. In particular, it compares the various approaches to multi-sensor azimuth determination, and discusses the issue of altitude estimation for non-altimeter-equipped aircraft. Then Chapter 7 considers the netting report timing issue: how many and when should these reports be generated. Next, Chapter 8 presents multilateration algorithms to handle all cases of aircraft knowledge concerning altitude and transponder turnaround delay.

Chapter 9 then develops a new form of incremental bilateration that is able to handle any form of system or aircraft bias. This approach, although employing a flat earth model, is shown to be more accurate than any form of spherical multilateration. This algorithm maintains consistency of tracking when biases are present, while simultaneously reducing the data variance. This technique will still have data jumps during sensor handoffs, but they can be handled as described above. The approach in this chapter is felt to be the major accomplishment of the netting study.

Finally, Chapter 10 presents and develops a number of smoothing trackers that are capable of dealing with multi-sensor inputs. In each of the chapters 6 through 10, results obtained via the simulation database are provided.

Chapter 11 deals with data editing, correlation improvement, and code improvement issues. The presentation is more discussion than algorithms, as this phase of the project had just been started at the time of the project's termination.

The culmination of the project was to have been the construction and deployment of a real-time demonstration netting system. Chapter 12 describes the work that has been completed to date.

Finally, Chapter 13 summarizes the major accomplishments and results of the surveillance netting project.

2.0 DATA CHARACTERISTICS

Netting is intended to correct the deficiencies of single sensor Mode S data. These deficiencies include azimuth inaccuracies at long range and in diffractions zones, reflection false alarm targets, garbled codes and altitudes, missing reports, and tracking errors. This chapter discusses and demonstrates the more serious azimuth and reflection issues; the other problems are discussed in a later chapter.

When data from two or more netted sensors is combined, system biases suddenly play a prominent role. Both sensor biases, such as location, range, and azimuth errors, and aircraft biases, such as transponder turnaround delays and altimeter errors, must be considered. This chapter discusses this bias issue in detail; later chapters refer back to it when algorithms are tested.

Finally, this chapter presents the multiple sensor data base that has served as the testbed for the netting and tracking algorithms that were developed in the study to overcome these deficiencies. This data base consists of live aircraft data collected from three sensors. It also has served as the basis for the simulated data needed for various tests that could not be made on these live inputs. This live/simulation connection is described in detail.

2.1 Azimuth Inaccuracies

The surveillance accuracy of the Mode S sensor, although uniformly good everywhere in the coverage region in the range coordinate, grows linearly poorer with distance in the cross-range (or azimuth) dimension. As the positional error in a target report becomes comparable to the per scan aircraft movement, determination of aircraft trajectories becomes very difficult.

The magnitude of the azimuth measurement standard deviation versus range was presented in the previous chapter. Figure 2-1 illustrates the effect of this size uncertainty with a track history plot of a typical aircraft. As seen, the jitter in the cross-range dimension is quite large. In fact, a standard turn detector would have signalled the presence of a turn several times during the segment shown, even though the aircraft was flying essentially straight.

The other major source of azimuth error is diffraction. This phenomenon occurs when the radar beam is distorted by the presence of narrow objects such as tall buildings. The target azimuth reported when diffraction is present is virtually a random variable over the antenna beamwidth.

Two visual examples of the effect of diffraction are presented in Figs. 2-2 and 2-3. In the first figure, a normally smooth track is severely distorted when it passes through a diffraction zone. In the second figure, a longer stretch of diffracted track is shown. It is clear in this case that any attempt at real-time heading determination would be impossible.

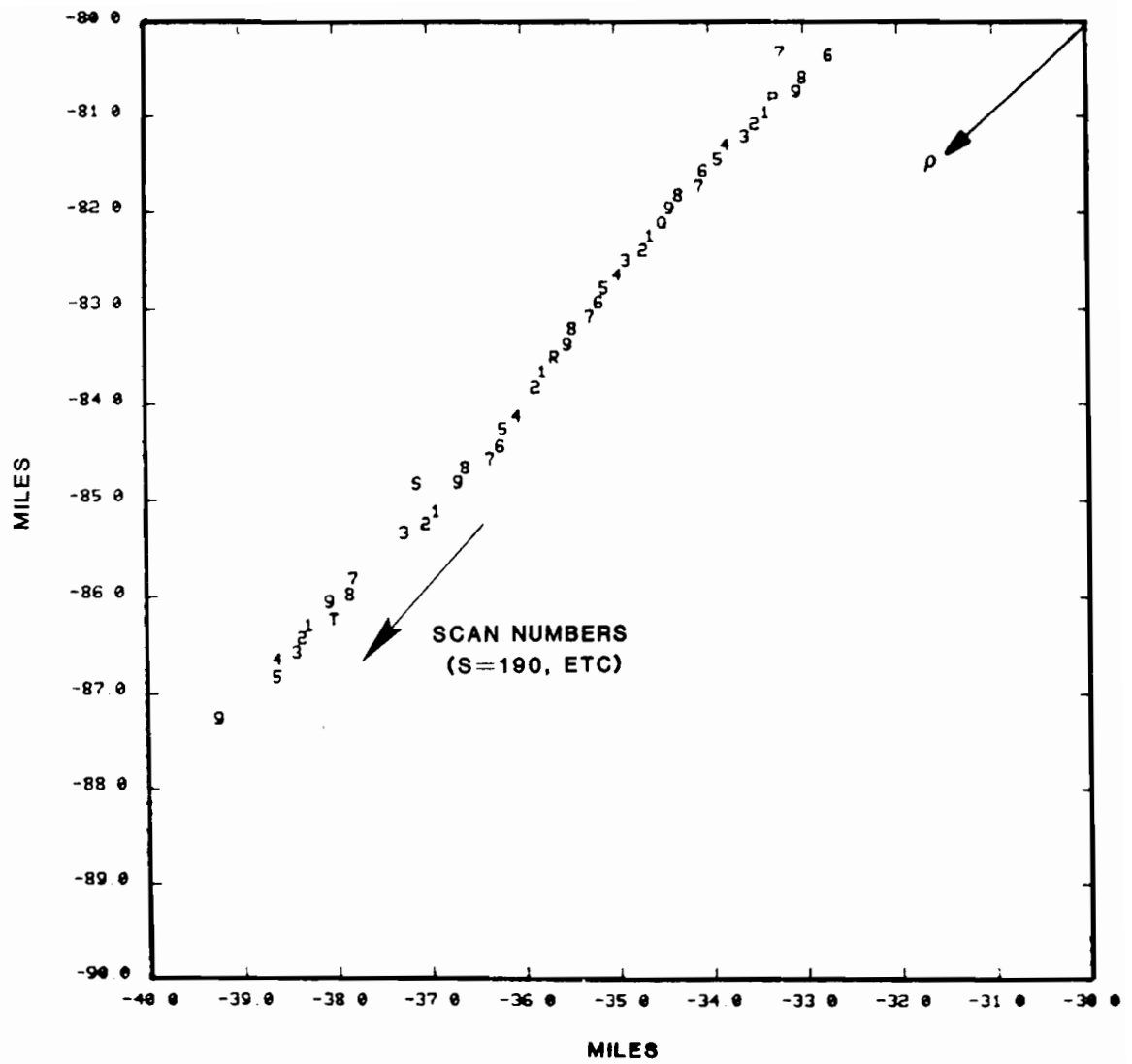


Fig. 2-1. Long range azimuth jitter.

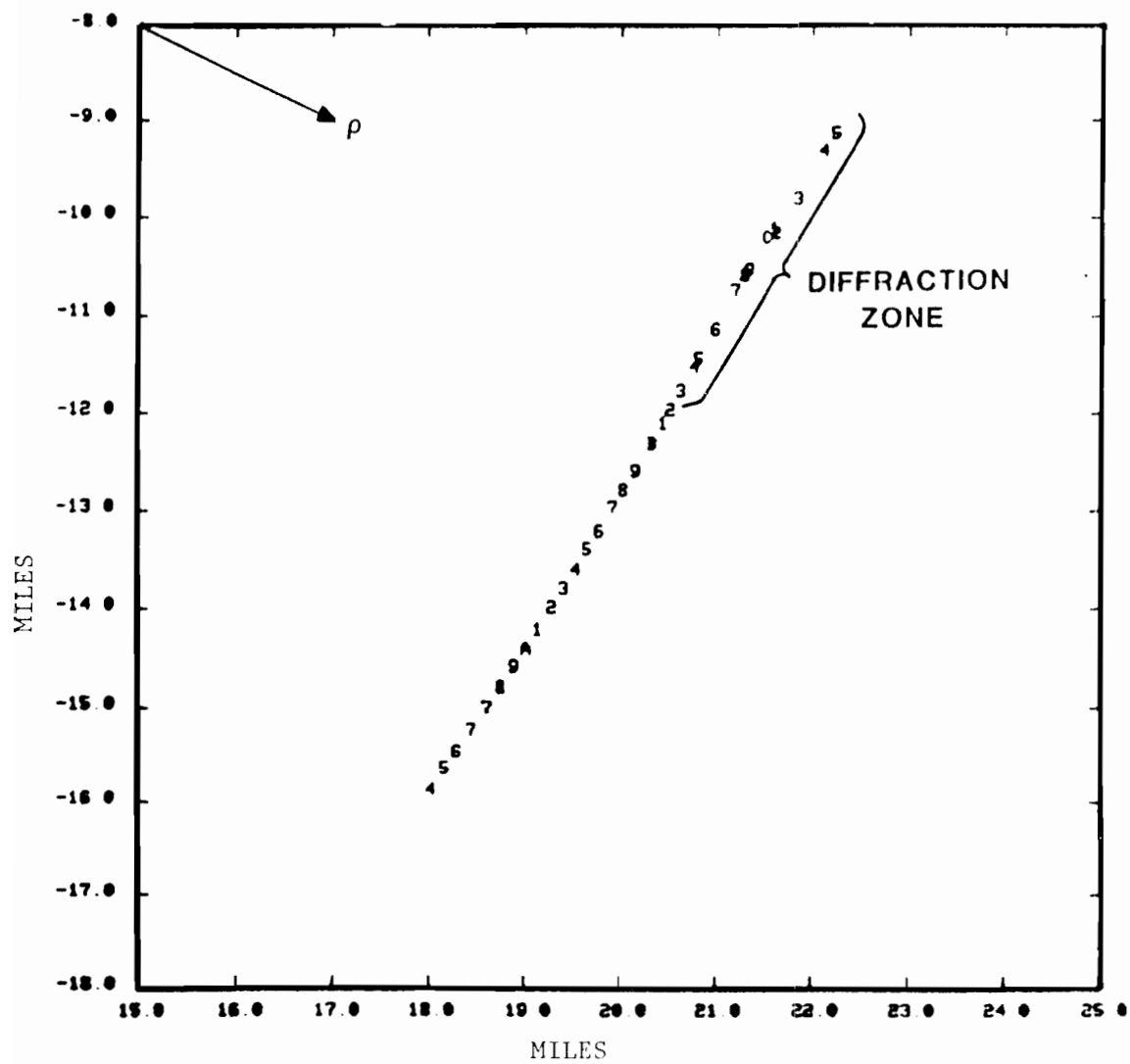


Fig. 2-2. Loss of accuracy in diffraction zone.

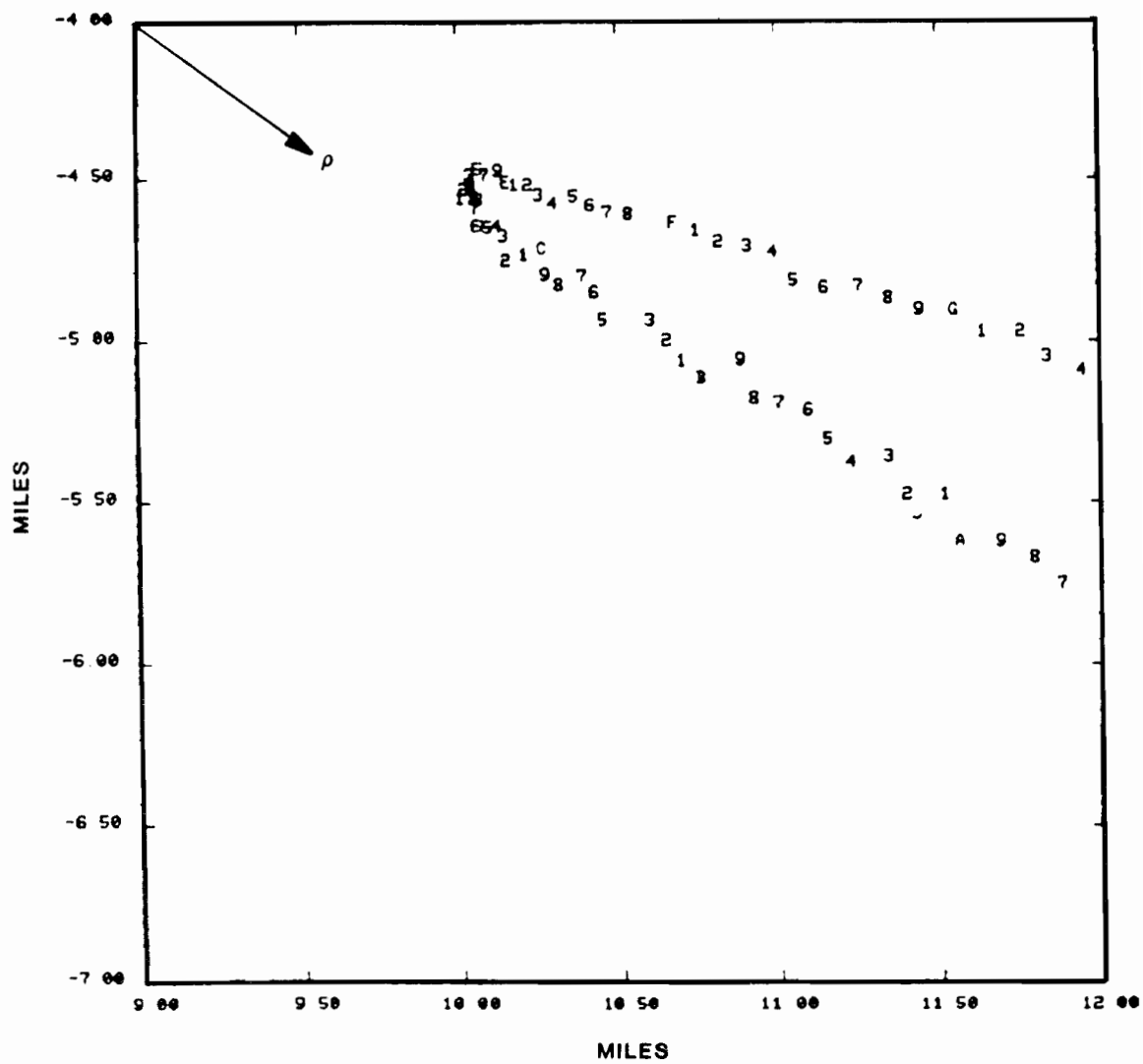


Fig. 2-3. Diffraction zone performance.

Fortunately, the diffraction zones for any sensor are usually fixed and well known. Thus when an algorithm such as multilateration is available to overcome its effects, the times to employ it can be pre-programmed into the surveillance system.

2.2 Reflection False Targets

A Mode S sensor will generate numerous types of extraneous reports during reply-to-reply correlation. Some of the more common of these are reflection false targets, correlating fruit, ringaround reports, and range or azimuth splits. Except for reflection false targets, these reports are either transitory and form no tracks, or identifiable, and hence suppressed by target-to-track correlation. Reflection false targets, however, often persist for a sufficient time to initiate tracks, and look exactly like real reports.

The mechanism leading to reflection false alarms, illustrated in Fig. 2-4, is the reflection of uplink and downlink signals off buildings or other large objects. The sensor, as shown, is led to believe that an aircraft is present behind the building. As long as the reflection geometry is maintained, this illusion will continue. Figure 2-5 presents a plot of all false targets found on a Mode S test run made at Lincoln. It is clear that several long false tracks are present.

Single sensor identification of these false targets requires:

- (a) knowledge of the reflector,
- (b) code agreement between false and real reports, and
- (c) predictable reflection angle.

Unfortunately, one or more of these conditions is often violated: new reflectors appear, incomplete reflection alters a code, and non-smooth surfaces lead to scattered signals. Thus, developing a netting algorithm for reflection false target confirmation is worthy of pursuit.

2.3 Sensor and Aircraft Biases

No sensor can deliver perfect data. Each position measurement contains random errors and, usually, bias errors as well. Random errors are uncorrelated from scan to scan and unavoidable, while bias errors are consistent and often (but not always) removable. Examples of bias errors in an aircraft surveillance system are:

Range

- 1. incorrectly zeroed range gate
- 2. non-exact aircraft transponder turnaround delay
- 3. radar signal refraction effects

Azimuth

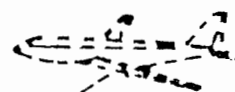
- 4. incorrect north alignment
- 5. antenna tilt

REAL AIRCRAFT



$\rho - d$

FALSE TARGET



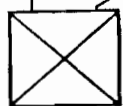
RANGE ρ
AZIMUTH θ

d

REFLECTOR

N

θ



SENSOR

Fig. 2-4. Reflection false alarm geometry.

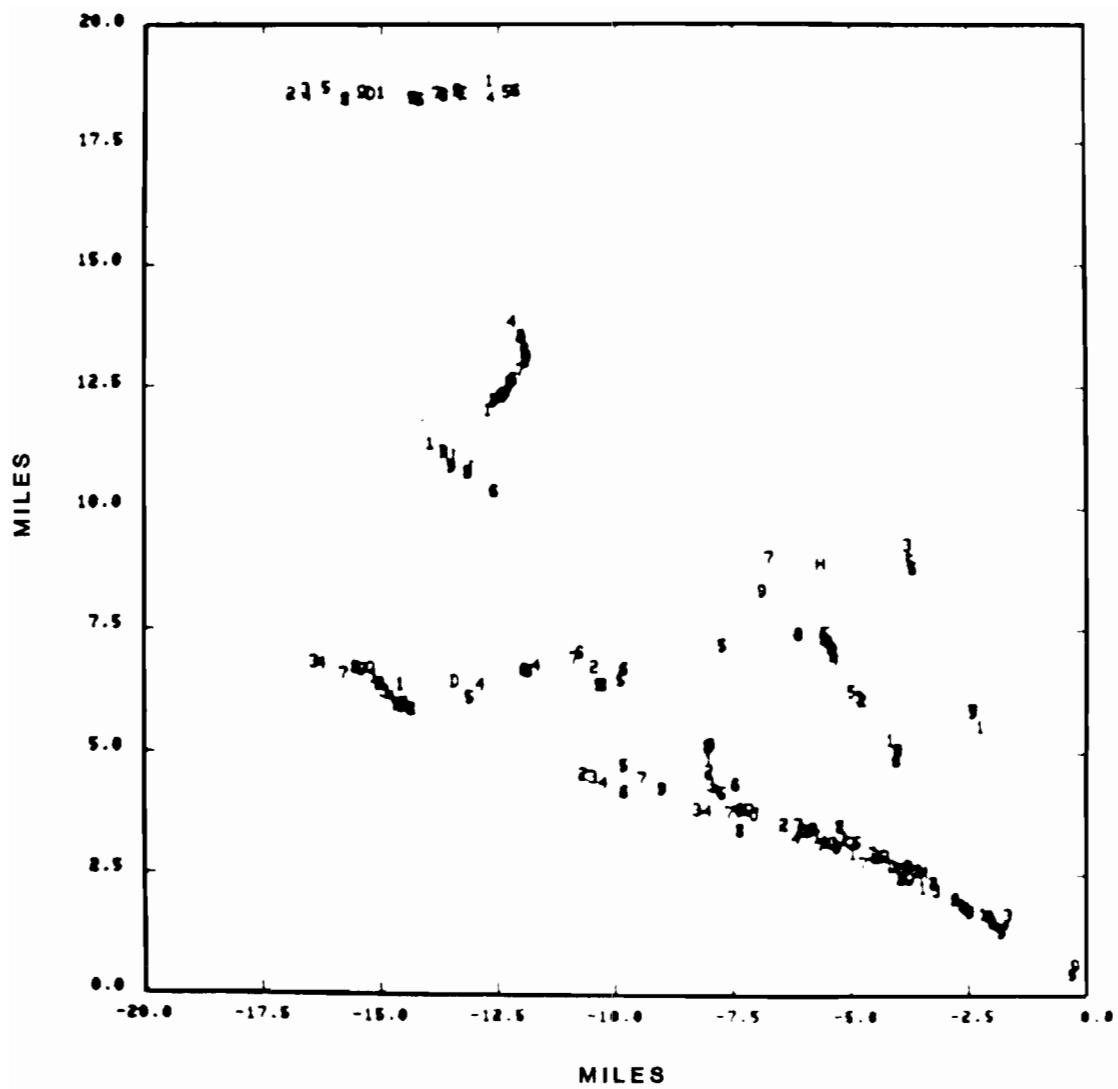


Fig. 2-5. False target reports.

Netting Data

6. incorrect secondary sensor position
7. non-exact earth model
8. inter-sensor time alignment error
9. data extrapolation error
10. altitude reporting error

When a single sensor is used for surveillance, bias errors tend to cause only minimal problems. None of the five single sensor biases listed above results in tracking anomalies, and only the second one produces relative position errors in nearby aircraft. Thus single sensor traffic control performance is determined by random errors and sensor measurement accuracies.

With netted sensors, on the other hand, bias errors can have a major impact on surveillance. First, the bilateration formulas are quite sensitive to some of the biases. Thus, even small bias errors can produce large position deviations. More importantly, bias errors produce different errors when different sensor combinations are used. For example, the data from

sensors 1 and 2 are used when both report on the target
sensor 1 only is used when sensor 2 has no report on the target
sensor 2 only is used when sensor 1 has no report on the target

These cases tend to intermix randomly, with all occurring a significant fraction of the time with normal sensor blip/scan ratios. The result of the different bias effects is data jumps each time the case on the current scan differs from that of the previous scan. Thus, tracking errors can become quite prominent when biases are present. Figure 2-6 illustrates this jump phenomenon for a typical surveillance system using real data. The plotted points indicate the sensor used on each scan, with 3 meaning both sensors.

The ideal method of preventing bias-induced problems is to eliminate all biases. At best, this would introduce great complexity into the system and necessitate frequent measurements and tests. See, for example, [3]. Even then, not every bias is detectable. A more practical method of dealing with biases is to devise a bilateration algorithm that is insensitive to them. Such an algorithm would have the following three properties:

1. bias errors would produce mean positional deviations for bilateration reports of the same magnitude as those for single sensor data, even though two sensor systems contain more possible biases.
2. the bias errors would produce the same mean positional deviations independent of the data source on a scan: sensor 1 only, sensor 2 only, or both sensor bilateration.
3. the bias errors would not prevent the bilateration from producing a lower positional variance than that for single sensor data.

NOTATION:

1=SENSOR 1 ONLY

2=SENSOR 2 ONLY

3=JOINT COVERAGE

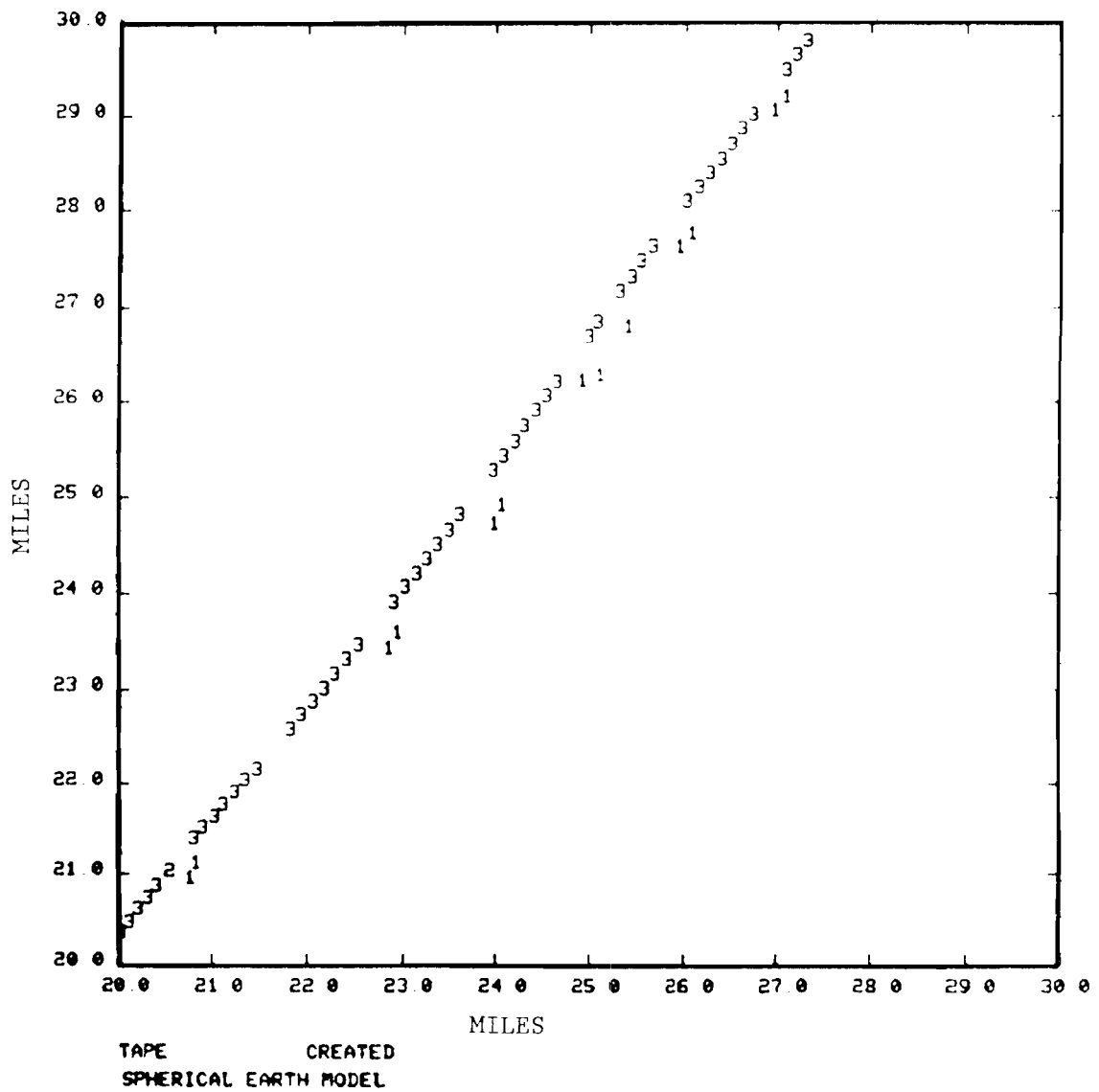


Fig. 2-6. Bias-produced data jumps.

An algorithm with these characteristics would be very satisfactory for applications that depend on the prediction of future aircraft position, e.g. conflict alert. Consistent data permits a tracker to accurately calculate heading and velocity, the two quantities that have the greatest affect on future position estimates. Chapter 9 describes a bias-insensitive algorithm of this type.

Although this algorithm will perform quite well in biased systems, greater positional accuracy, as opposed to consistency, can be achieved if system biases were removed whenever possible. Thus, careful sensor calibrations and exact sensor locations should be pursued. Also, daily updated refraction compensation formulas can be maintained. The goal to be obtained from these and similar measures is a system with all sensor biases eliminated.

Even if such perfection could be attained, however, one bias will remain in beacon systems: the aircraft transponder turnaround delay error. Current specifications permit this nominal 3-microsecond delay to have a bias of up to 500 nanoseconds, corresponding to a sensor-computed range error of 0.04 miles. Not all aircraft transponders are within specifications, so even larger errors will be encountered in practice.

In addition, not all aircraft report their altitude, since encoding altimeters are not required in all airspace. With primary radar surveillance, of course, no aircraft altitude is known (unless height finder radars are employed). Finally, even if altitude is reported, it may be in error.

Unfortunately, aircraft biases such as these are more serious than sensor system biases, since they are the ones that affect nearby aircraft differently. Thus relative positions are altered, compromising conflict alert techniques. The only way to remove these biases is in real time, using the data from the two (or more) sensors.

2.4 Netting Database

Several years ago, prior to the start of this program, a multisite database was generated by simultaneous operation of three sensors: the Lincoln Mode S experimental facility (MODSEF), the Lincoln transportable measurement facility (TMF) stationed at Providence, and the existing ARTS at Logan Airport. These three data streams were then transformed into a common format and merged. From this overall database, reports corresponding to two dozen aircraft were extracted and used to form an input test package for the netting algorithms under development. The 24 tracks chosen satisfied the criteria of long length, visibility to all sensors, a mix of trajectory types, and a geographic distribution over the joint coverage region. Figure 2-7 presents these tracks as viewed by the primary sensor, the TMF, as well as the locations of the three sensors. The figure also shows the adjusted position of the tertiary sensor that was used by simulation studies to improve the system geometry.

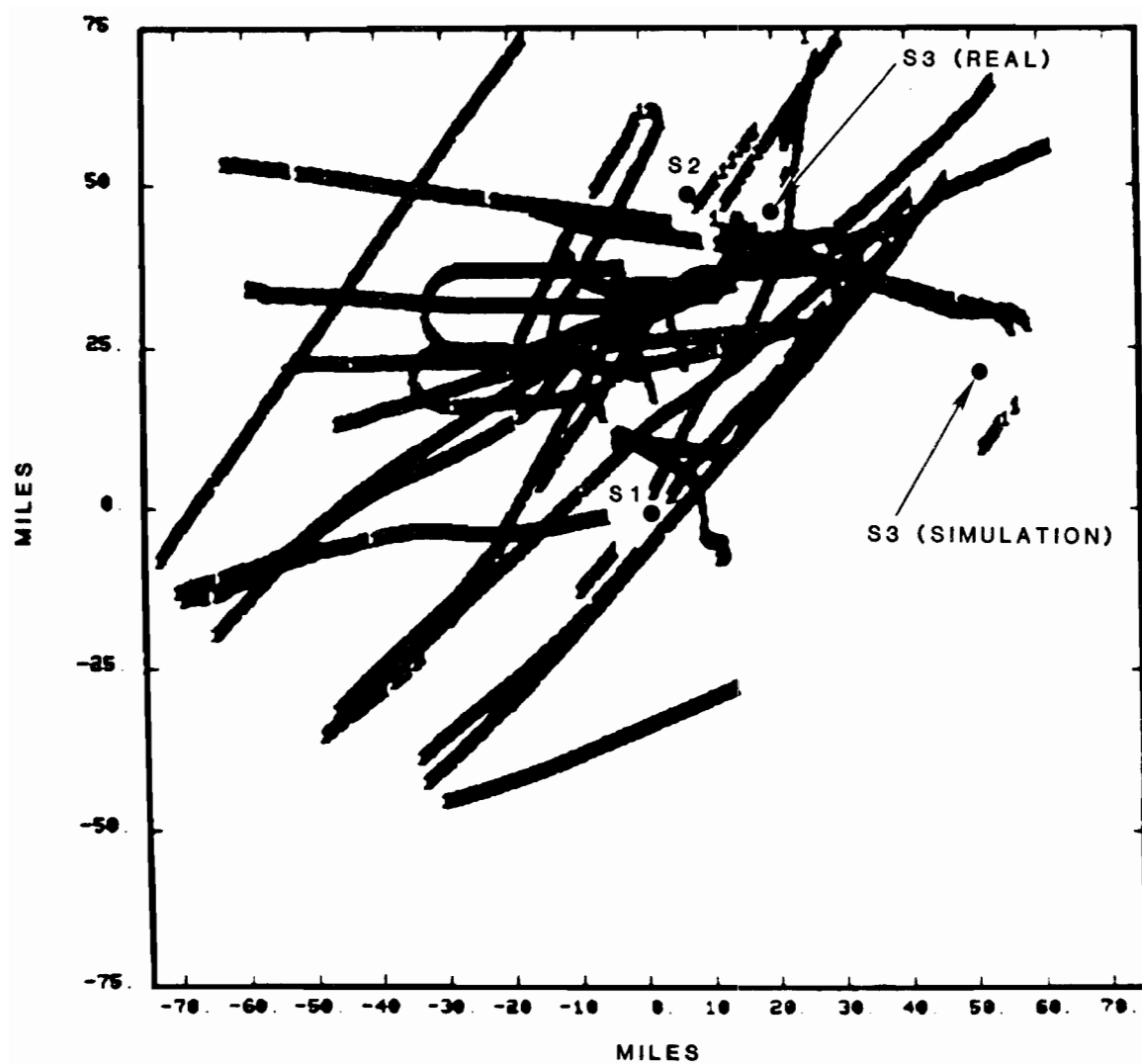


Fig. 2-7. Simulation database trajectories.

The main problem with the database was lack of accurate report time tagging. Each sensor's clock was independently set, with the relative zero offsets unknown. Also, neither MODSEF nor TMF had the ability to record the time at which the reports were received; only approximate sector times were known. Thus, complex time alignment and adjustment procedures had to be developed for the sensor merging step. These procedures employed altitude reporting on climbing (or descending) aircraft for identifying the relative zero offsets, and interpolation for individual report time tagging. The results were usable, but not perfect, and introduced biases into the data.

In addition, it was impossible to measure precisely the various sensor biases that existed in our multisite system, or to obtain exact sensor location coordinates. Thus, the overall effect of the totality of biases in the database was input data not sufficiently accurate for detailed evaluation of all netting position improvement algorithms.

In the past year, synchronized clocks have been added to MODSEF and TMF, so that time alignment is now possible. Also, time tagging of all reports has been included. Thus the new database that was to have been generated would have been suitable for testing.

The existing database, however, served as an excellent source for a simulation input generator. This facility worked as follows. First, the actual 24 trajectories were used, so that realistic aircraft data would be produced. These trajectories were determined by smoothing the TMF reports. Then a report stream was created for each sensor, with the time of each report specified by the times of the actual recorded data, so that a realistic inter-sensor data interlace was maintained. Finally, the position of each report was offset from the "real" aircraft position at that time, the offset determined by the system biases and measurement standard deviations selected by the user. The result of this process was the creation of a combination real/simulation database.

This simulation process was later augmented to provide a greater variety of test cases. In particular, relocation of sensors was permitted to test the affects of inter-sensor distances and relative geometries, blip/scan was made controllable, diffracted azimuths could be admitted, and sensor scan rates were allowed to be modified to simulate enroute performance. Finally, controlled trajectories were added to the basic aircraft set to test tracking algorithms. These trajectories were straight, constant turning, accelerating, and any sequential combination of these.

The simulation processes the data one group of scans at a time, each such group preceded by a short initialization period. For example, if the initialization and statistical sections are 5 and 20 scans long respectively, a section of the simulation might proceed as follows:

```
scans 66-70: initialization
scans 71-90: data gathering
scans 86-90: initialization
scans 91-110: data gathering
```

The backup in time for each initialization insures that all scans can be included in the overall set of data. In fact, all scans are included in the statistics, with the exception of time periods during which the primary or secondary sensor has no track on the target and thus netting is impossible.

The constant reinitializations serve to make the statistics more meaningful. In real life, netting will be started when difficulty is approaching (diffraction zone, conflict, etc.), and be expected to provide accurate results after a short period of reaching steady state. Thus algorithms such as those for transponder delay and altitude estimation should be analyzed in such a fashion. Also, for diffraction algorithms, accuracy tends to drop off the longer the aircraft stays in the diffraction zone. Thus a valid test is constructed by an initialization period of normal data followed by a statistical period whose length matches a typical diffraction zone. The above procedure provides for such an approach.

The original data base contained the total sensor output including false targets, marginal tracks, incorrect correlations, etc. It was intended that this collection of reports be used to test data editing algorithms. However, lack of time prevented this type of testing.

3.0 LOCATION OF NETTING ALGORITHMS

Two conceptually different approaches exist for netting, namely centralized and localized. With centralized netting, all sensors in a region transmit their target reports to a common facility, at which facility the netting algorithms are exercised. On the other hand, with localized netting, each sensor is responsible for performing netting for the aircraft under its coverage.

In reality, this dichotomy is not quite so pronounced, as various hybrid schemes exist for the netting location problem. This chapter discusses the full set of alternatives. No "best" choice is uncovered, as the proper scheme is dependent upon the functions to be supported by the data. In particular, the NAS enroute system would tend to employ a centralized approach, while the terminal sensors would require a localized one.

Two of the major differences between the various location alternatives are computational and inter-facility communications load requirements. Full netting benefits, as presented in later chapters, assume that all supplementary data is always available for netting. Unfortunately, the computer and communications requirements for such a methodology could be overwhelming. Thus this and the next two chapters explore ways to minimize the data processing and communication requirements. Protocols on what data is sent, where it is sent, and the basis or criteria for its triggering are all explored.

3.1 Centralized Netting

A typical centralized netting architecture is illustrated in Fig. 3-1. As shown, all sensors, as well as all users, are connected to a single facility. If all sensors in an area send their target reports to a common FAA facility, it would be possible to implement the netting algorithms at such a location. If no such facility existed, it could easily be constructed by connecting the sensors accordingly. The advantages of such a centralized netting procedure are twofold:

1. all possible data for the netting algorithms are available
2. communications cost is minimized, as no data transfer between sensors is required.

One-way centralized netting is not sufficient, of course, if critical functions exist at the local sensors. In such a case, the local sensors may require the best surveillance data known at the central facility. Thus, a reverse flow of data could be required on the links. This type of protocol is examined later in this chapter.

Centralized netting can be characterized as follows:

1. the totality of data exists at one location
2. all users are fed from a common source

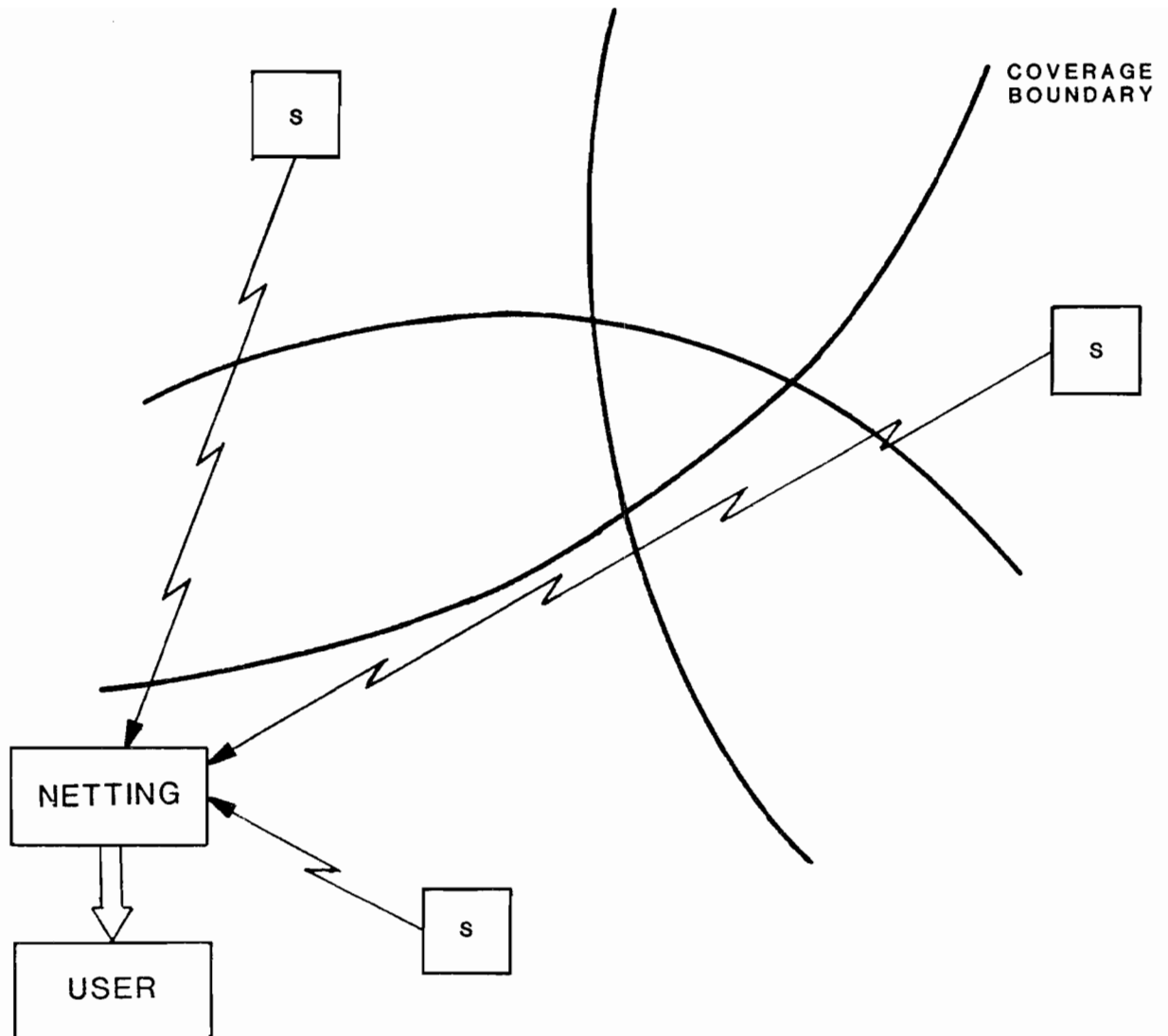


Fig. 3-1. Centralized netting system.

3. sensor locations are irrelevant, provided coverage constraints are met
4. users are disassociated from the radars
5. netting algorithms can always be employed.

For many systems, these features could all be considered advantages.

The separation of users from the sensor system is particularly attractive in many cases. This feature allows unencumbered optimization of the data collection network. It also ensures that all users receive consistent information on which to base joint decisions. Thus, no contradictory decisions should be possible.

Two diverse methodologies exist for centralized netting. The first is for the central facility to use data from all sensors to perform optimum surveillance at all times. The computer capacity requirements may well make this approach infeasible. The second possibility is to have each sensor flag every output report for which help is required, such as when the azimuth is in a low reliability region or the correlation is suspect or absent. The central facility, upon noting such a flag, would examine alternate sensor data and perform the required improvement operation.

With either approach, the central facility must perform sensor-to-sensor correlation on all tracks. This correlation, as a minimum, must be done each time a new sensor track is initiated as well as each time a sensor inadvertently makes a track swap error. If the sensor were able to indicate to the central facility when such an error potentially existed, this requirement would not be too time consuming. It is suspected, unfortunately, that this will not prove to be the case. As a result, continuous scan-by-scan correlation verification may be necessary. Only experience with real data will demonstrate the magnitude of this problem.

The other potential problem, relating to the second methodology of sensors flagging problems, is how a sensor will know when to trigger netting by the central facility. This question is examined in the next chapter.

3.2 Localized Netting

At the opposite extreme, each sensor could perform the netting algorithms itself for all targets under its control. The architecture for this approach is illustrated in Fig. 3-2. The advantages of this scheme over the centralized one are:

1. the computer load for netting is spread out among the sensor computers
2. a sensor can receive data from sensors tied to many different FAA facilities
3. a new sensor can be added to an existing netted system with minimum change to the processing performed in any other part of that system.

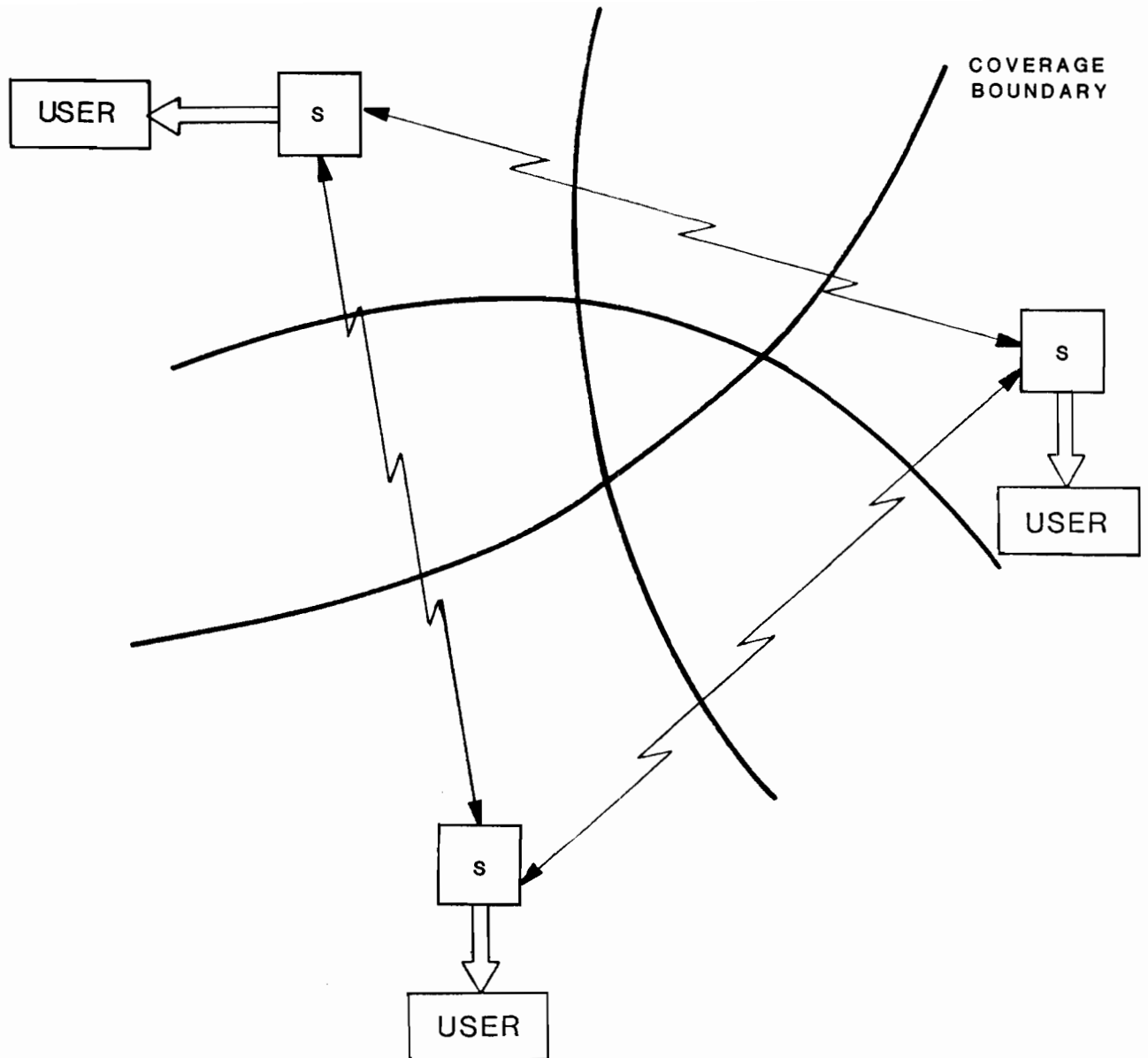


Fig. 3-2. Localized netting system.

Localized netting can be characterized as follows:

1. the netting algorithm load is distributed over the entire system
2. system reliability is high, as no one facility is vital to its operation
3. sensors are located at the users, so data is supplied locally
4. netting is performed only when required, at some times, and in some areas
5. alternate sensor coverage is available during outages.

The distributed nature of this approach is usually viewed as its main advantage.

The key to localized netting is the ability of a sensor to determine, by itself, that help is required. Thus, for each situation in which netting is to be employed, an algorithm is required which signals its onset. The characteristics of each situation that lead to the triggering of netting are discussed in the next chapter.

Localized netting requires modifications to the Mode S sensor. Clearly, interfaces for transmitting and receiving data from other sensors must be added. In addition, software algorithms for employing secondary sensor data must be included, as well as ones for identifying the data to be sent to other sensors. The new sensor functional diagram and the additional algorithms are discussed fully in the next chapter.

3.3 Hybrid Netting

Two hybrid architectures, combining some of the advantages of both centralized and localized systems, can be devised. In each of these, the local sensor is responsible for its own surveillance and user support, while a central facility handles all inter-sensor communications. Both systems have the same architecture, as shown in Fig. 3-3. The difference between them is the location of the netting algorithms: in the central facility, or distributed among the sensors.

With the centralized netting hybrid, the central facility executes the netting algorithms precisely the same as for a purely centralized system. Either the at-all-times or the on-demand netting schemes can be utilized. However, since the local sensor is responsible for serving its own users, this central facility must supply the local sensor with the improved data. Expressed differently, the central facility must correct local sensor tracks whenever errors or inaccuracies are identified via the use of the netting algorithms.

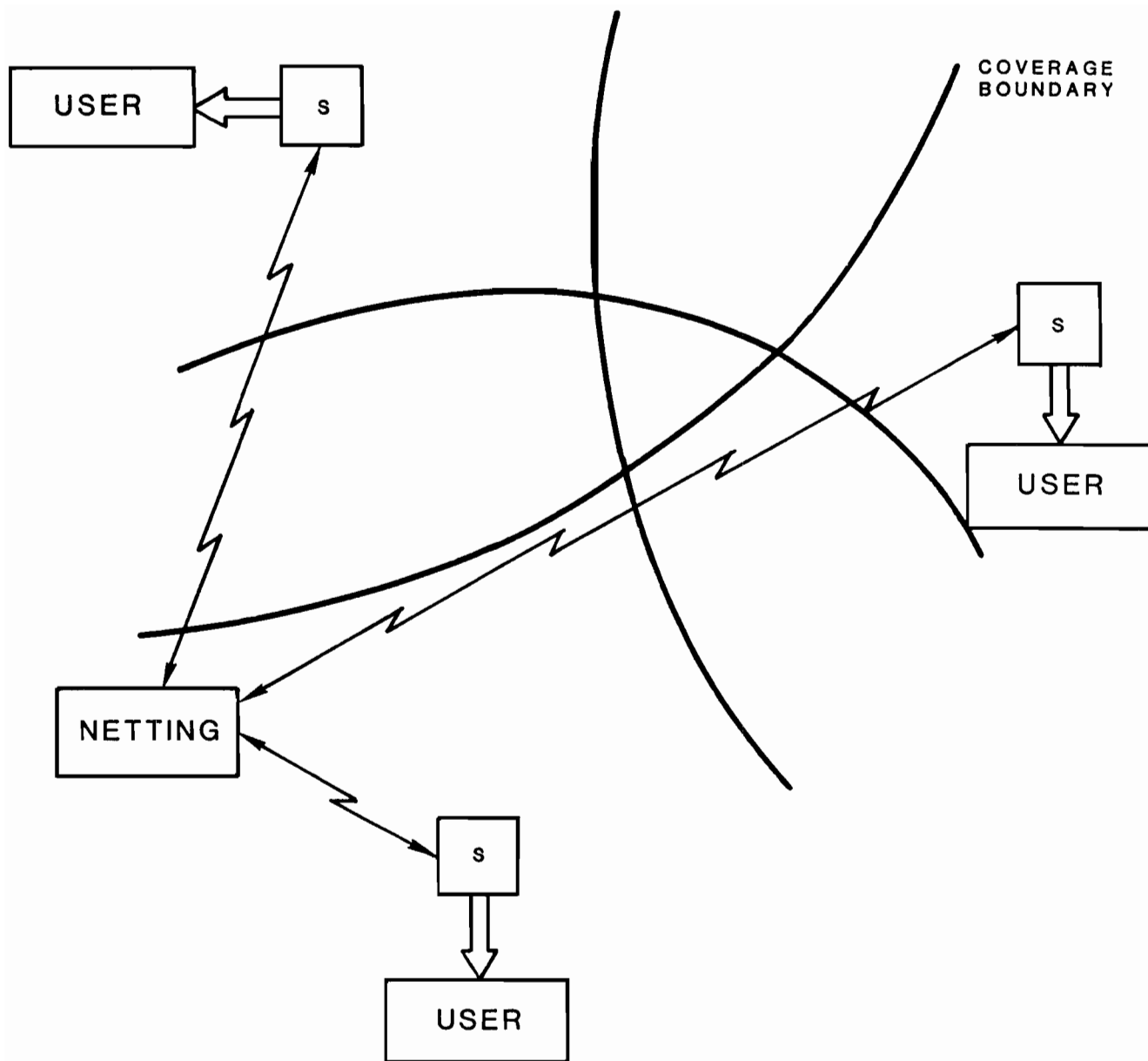


Fig. 3-3. Hybrid netting system.

The simplest, and most direct, method for achieving this result is for the central facility to transmit a replacement track file entry to the local sensor. The sensor would then substitute this improved data for its own, non-netted version. Then any user would be subsequently fed information from this netted file.

The localized netting hybrid operates virtually the same as a pure localized netting system. The main difference is that all data sent from secondary sensors in response to help messages is routed through the central facility. This architecture can significantly reduce communications line expenses, as no sensor-to-sensor lines are required.

This hybrid architecture makes sense whenever sensors are regionalized, and natural central facilities exist. It cannot be employed, though, if sensors are treated as a continuous grid system. In that situation, no closed set of sensors would exist, and all sensors would thus have to be connected to a single facility. This would clearly increase, not decrease, communications costs.

4.0 LOCALIZED NETTING SYSTEM

This chapter presents a detailed description of the modifications that must be made in the sensor processing algorithms [1,4] to incorporate netting. The algorithms and procedures to be described represent the expected best method for implementing localized netting. Since no testing, either live or simulation, has yet been attempted, no guarantees can be provided. It is probable, in fact, that various small algorithm omissions exist and that some modifications will be required to the ones presented.

Although no description of a centralized netting system is developed in this document, this lack is unimportant. Besides inter-sensor correlation, which is covered in the next chapter, no system modifications of consequence are required for centralized netting beyond implementation of the netting algorithms presented in the remainder of this paper.

4.1 System Overview

A high-level block diagram indicating the functions and messages in a netted Mode S sensor is presented in Fig. 4-1. As shown, a number of processes and data structures are required to convert a standalone sensor to a netted one. These additions include netting items and supporting items. Netting items are those items needed by the sensor to permit it to act as a primary netting sensor receiving aid from neighboring joint-coverage sensors, while supporting items permit the sensor to support the netted surveillance of these same nearby sensors. The new algorithms, by section, are:

Netting Algorithms

1. triggering rules
2. request generator
3. response processor
4. netting subsystem

Support Algorithms

5. request processor
6. inter-sensor correlator
7. response generator

In addition, new data structures are required to store request and response information, and new fields are needed in the system track file.

The netting algorithms operate generally as follows. The triggering rule subroutine examines each track as it is updated. If a situation requiring secondary sensor data is identified (such as diffraction zone or correlation difficulty), the track number is passed to the request generator. This function sends start messages whenever necessary to other sensors to initiate their data flow. The returning flow of reports is processed and

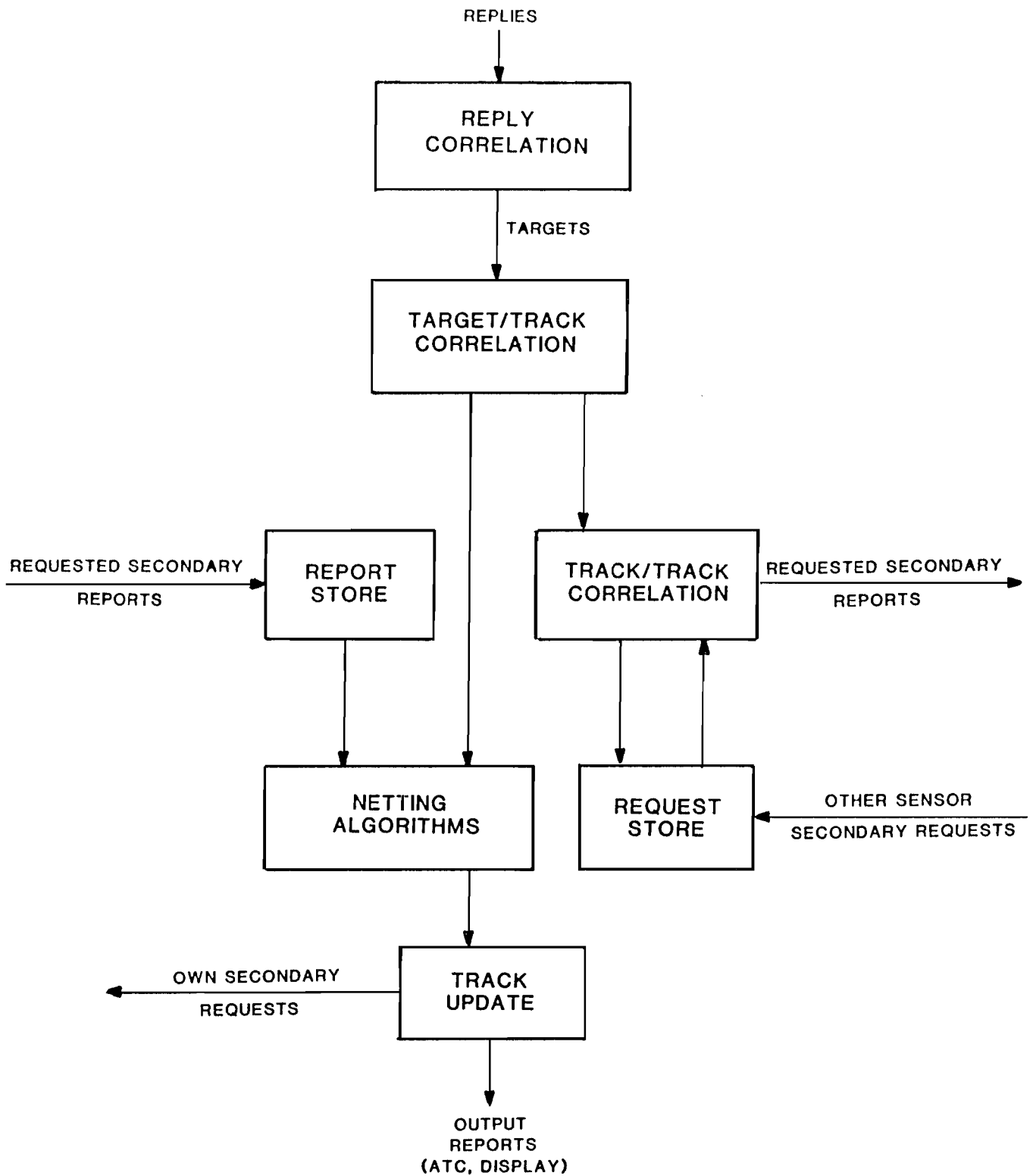


Fig. 4-1. Netted Mode S sensor.

stored by the response processor, in a manner linking them to sensor track files. Finally, the netting subsystem jointly processes primary and secondary sensor reports whenever a track marked by the response processor is ready to be updated. This netted report is then used to update the track, and the cycle restarts. Details of these actions are provided in the next sections.

The support algorithms begins with the reception from another sensor of a request for data. This request is coordinate converted and projected forward if related to a non-discrete ATCRBS aircraft, and left as is if related to a Mode S or discrete ATCRBS aircraft. The request is then sent to the inter-sensor correlation routine, which attempts to match it to a local track. If successful, the track is marked, and future reports correlating with the track are transmitted to the requesting sensor by the response generator. Details of these actions are provided in the final section of this chapter and in the next chapter.

4.2 Inter-Sensor Messages

A sensor requiring netting help for a track must transmit a message to an appropriate overlapping coverage neighbor (or neighbors) specifying the aircraft for which help is desired. This secondary sensor must then respond with its target reports for the aircraft. There are many possible message protocols that can implement this basic transfer of information to define the communications system to cover all needed information exchange.

The basic guidelines that were followed in selecting the message protocol were:

1. no formal ack/nak requirement would be imposed
2. any message, in either direction, should be able to be missed without harm to the system
3. requests for netting help should be made only once per track, whenever possible.

A formal message acknowledgement protocol adds both complexity and overhead to the system. When every message must be properly received, it is necessary; in this system, that requirement is not present. However, provision must be made for proper operation in the presence of lost or received-in-error messages. Thus, in particular, messages are repeated until actions of the receiving sensor make it clear that they have been properly received.

The requirement that requests for netting help not be repeated each scan serves to reduce the communications load significantly. However, it does add two complicating features to the system. First is the need for termination of help messages, and second is the need for inter-sensor correlation in the absence of continuous position updates on the aircraft. These features are discussed below.

Two types of messages from primary to secondary sensor are defined, namely, initiation and termination of netting data on a given aircraft. The general formats of these messages, which are differentiated by the start/end bit, are given by Fig. 4-2. The initiation message, as shown, contains a complete target report plus additional features needed for inter-sensor correlation. These features, namely time of relevance of data and aircraft velocities, permit extrapolation of data as needed. The track number, as will be shown, is used in several ways by the receiving sensor for request identification. The only termination message field that is relevant is the track number. This is sufficient for proper operation. This message uses the same format as an initiation message to simplify the reception of intermingled message streams.

There are also two message types defined for the return link, differentiated by the setting of the "help" bit as illustrated in Fig. 4-3. The first, and most obvious, is the secondary sensor target report for the aircraft in question. This message again contains the time and velocity information to enable data extrapolation, and the primary sensor track number for identification. The second message, containing relevant information only in the track number field, is a "help" message from the secondary sensor. It is employed when that sensor needs new position information on the aircraft to aid in its inter-sensor correlation. The response to this message is a new initiation message from the primary sensor. As above, common formats are used for simplicity.

The rules for employment of each message type are supplied later in this chapter. As suggested above, each message will be repeated if necessary to guarantee its successful acceptance by the receiving sensor. Methods of recognizing this condition are keyed to the data structures to be described next.

4.3 Message Handling Data Structures

A number of new data structures are required by a sensor involved in netting. These structures are needed for record keeping, request storage, and netting data report storage. This section presents a set of structures that meet all the netting requirements, while being efficient in both memory and user processing time measures. Other implementations are of course conceivable.

The first structure, depicted in Fig. 4-4, is the track status array. This array stores the type of netting (if any) each sensor track is currently undergoing. Each entry, indexed by track number, contains two types of information in three fields. The first type, in the first field, is the netting condition value for the track as set by the triggering routine. The meanings of each legal value are defined in the next section. The other information, requiring two fields, is the status of netting data from auxiliary sensors. The first subfields of the second and third fields are used to record the sensor numbers of the secondary and tertiary sensors respectively selected to supply netting data. The other subfields are single bits that are set to "1" by the response processor whenever a returned response is received from the corresponding sensor. The presence of this "1" is used to indicate the successful reception of a netting initiation message.

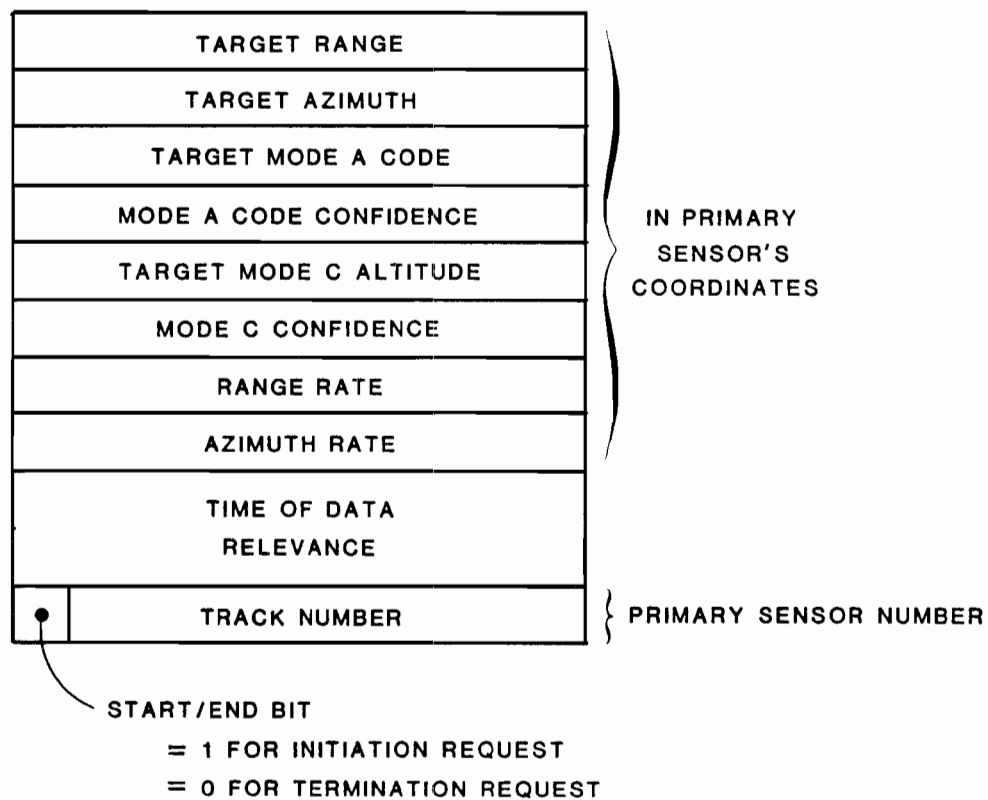


Fig. 4-2. Netting data request messages.

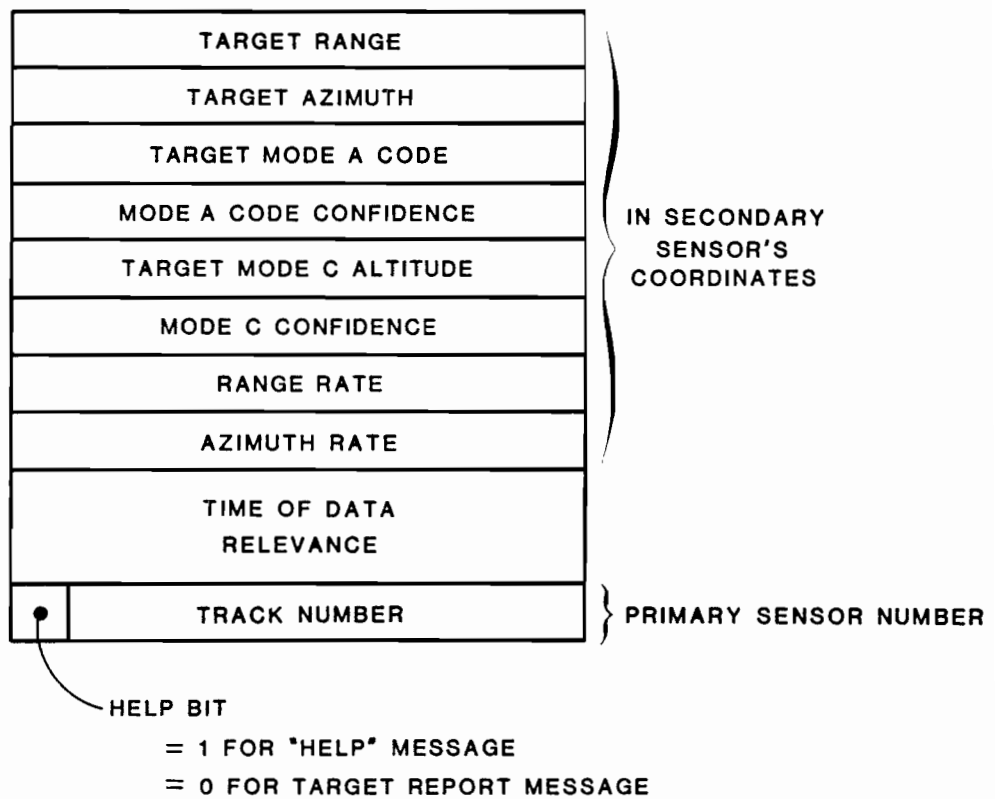


Fig. 4-3. Netting report messages.

TRACK NUMBER

1

2

3

N

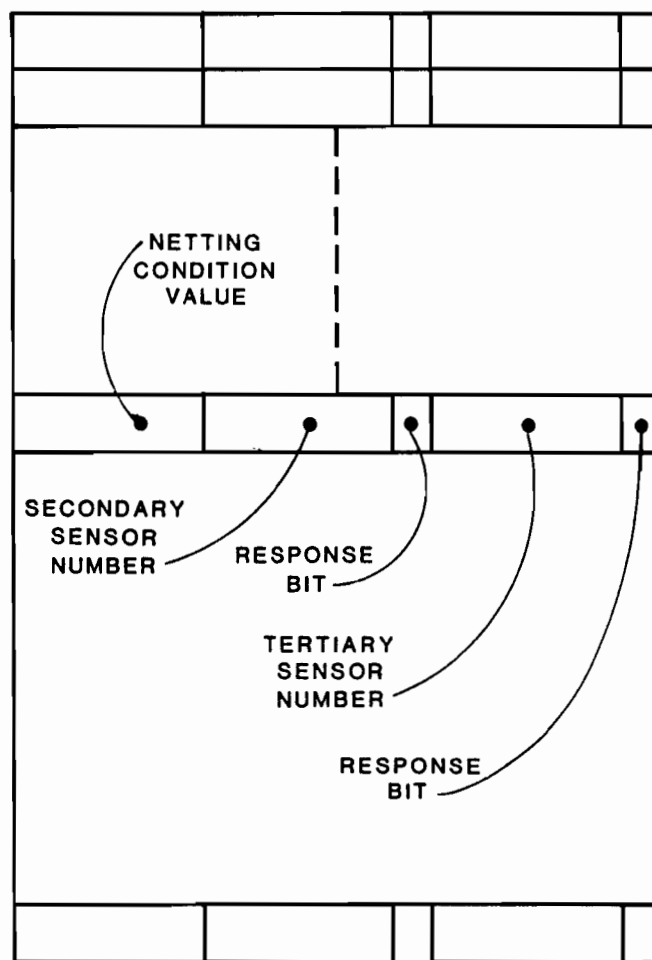


Fig. 4-4. Track status array.

The second data structure, the netting report buffer, is needed to store netting reports sent by other sensors. Since each such report contains the local sensor track number, association of message with track is straightforward. Thus the links from track file to report buffer, shown in Fig. 4-5, are easy to build. Linked lists, rather than dedicated buffer segments, are used because the maximum number of netted reports returned for a given track can be quite large, particularly for enroute sensors employing terminal sensor help. Thus much efficiency is gained by links. The penalty for this implementation is the need for a free-storage list linking available buffer slots, a minor overhead. Newly arriving reports take the first entry on this list; entries are returned to the head of the list after their reports are processed.

The link pointer to each new report received is always placed in the track file's pointer field; if a pointer already exists, it is moved to the pointer field of the new buffer entry. Thus new entries are always inserted at the head of the list. This method eliminates chain following, and causes no problems as the netting reports for a track need not be time ordered.

Two additional data structures are needed by the sensor when it serves in its capacity of netting support for another sensor. The first, shown in Fig. 4-6, is simply a circular buffer for the storage of netting requests. Such a first-in-first-out buffer suits the scheme of processing requests in order of arrival. Two pointers, as shown, specify the limits of the requests awaiting processing.

The other data structure is the inter-sensor correlation array. This array stores the track number matches identified by the inter-sensor correlation routine. The array implementation, shown in Fig. 4-7, permits determination of the match from either side, that is when either the local or other sensor track number is known. The need for both directions is shown later. When the index is interpreted as a local track number, the first field of the i^{th} pair of fields of the entry specifies the track number for the aircraft used by the i^{th} sensor netted to this one. (It is assumed that an ordered set of other sensors that can be netted to each sensor will be known). On the other hand, when the index is interpreted as an adjacent sensor track number, the last field of the i^{th} pair of fields specifies the local track number for the aircraft represented by the index track number at the i^{th} sensor netted to this one. The example in the figure is meant to clarify this representation.

Finally, transmission and receive buffers are needed by the inter-sensor data links. These structures are standard, and will not be discussed here.

4.4 Netting System Algorithms

A flowchart of the actions of the overall netting system is presented in Fig. 4-8. The details of the four routines that constitute this system are provided below.

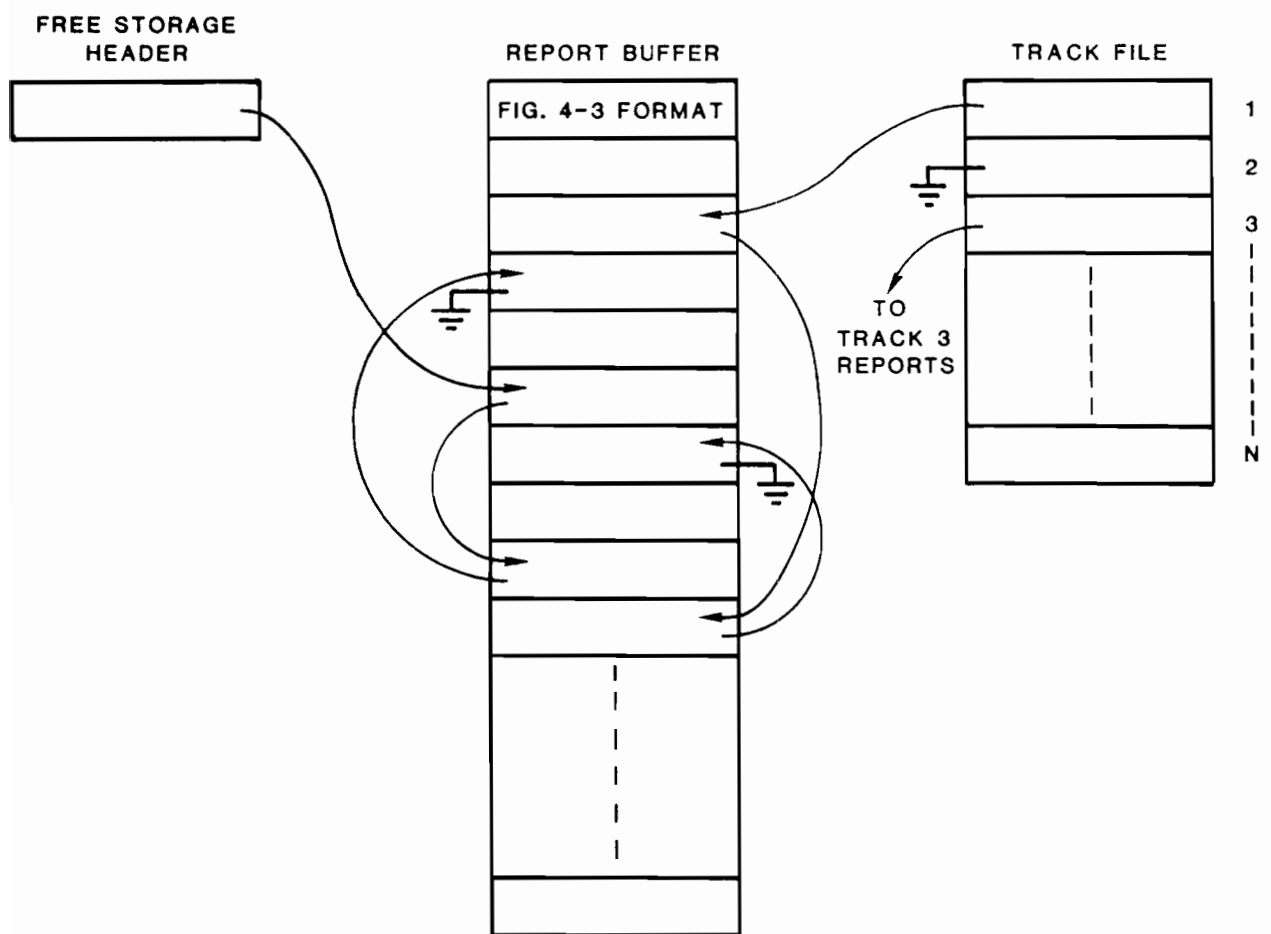


Fig. 4-5. Netting report buffer.

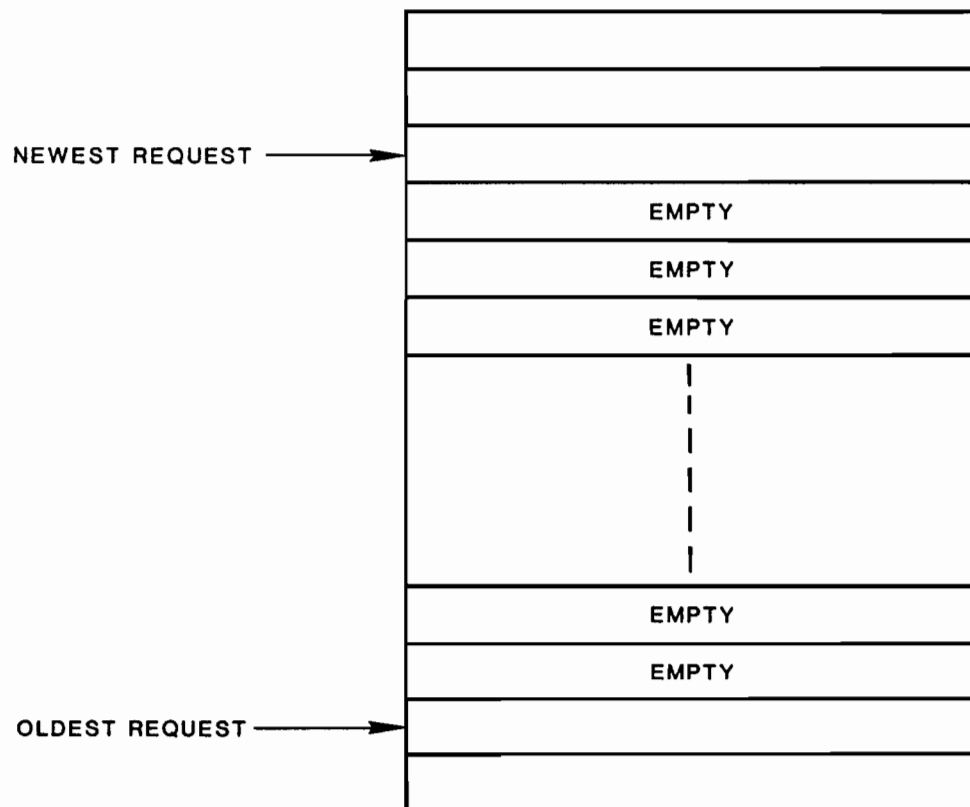


Fig. 4-6. Request storage buffer.

TRACK NUMBER

1
2
|
|
|
|
|
n
|
|
|
N

7	5	13	1
SECONDARY SENSOR FIELD		TERTIARY SENSOR FIELD	

FOR THE EXAMPLE ENTRIES SHOWN ABOVE:

- (a) TRACK #2 AT THIS SENSOR CORRESPONDS TO:
TRACK #7 AT THE SECONDARY SENSOR
TRACK #13 AT THE TERTIARY SENSOR
- (b) TRACK #2 AT THE SECONDARY SENSOR CORRESPONDS TO:
TRACK #5 AT THIS SENSOR
- (c) TRACK #2 AT THE TERTIARY SENSOR CORRESPONDS TO:
TRACK #1 AT THIS SENSOR

Fig. 4-7. Inter-sensor correlation array.

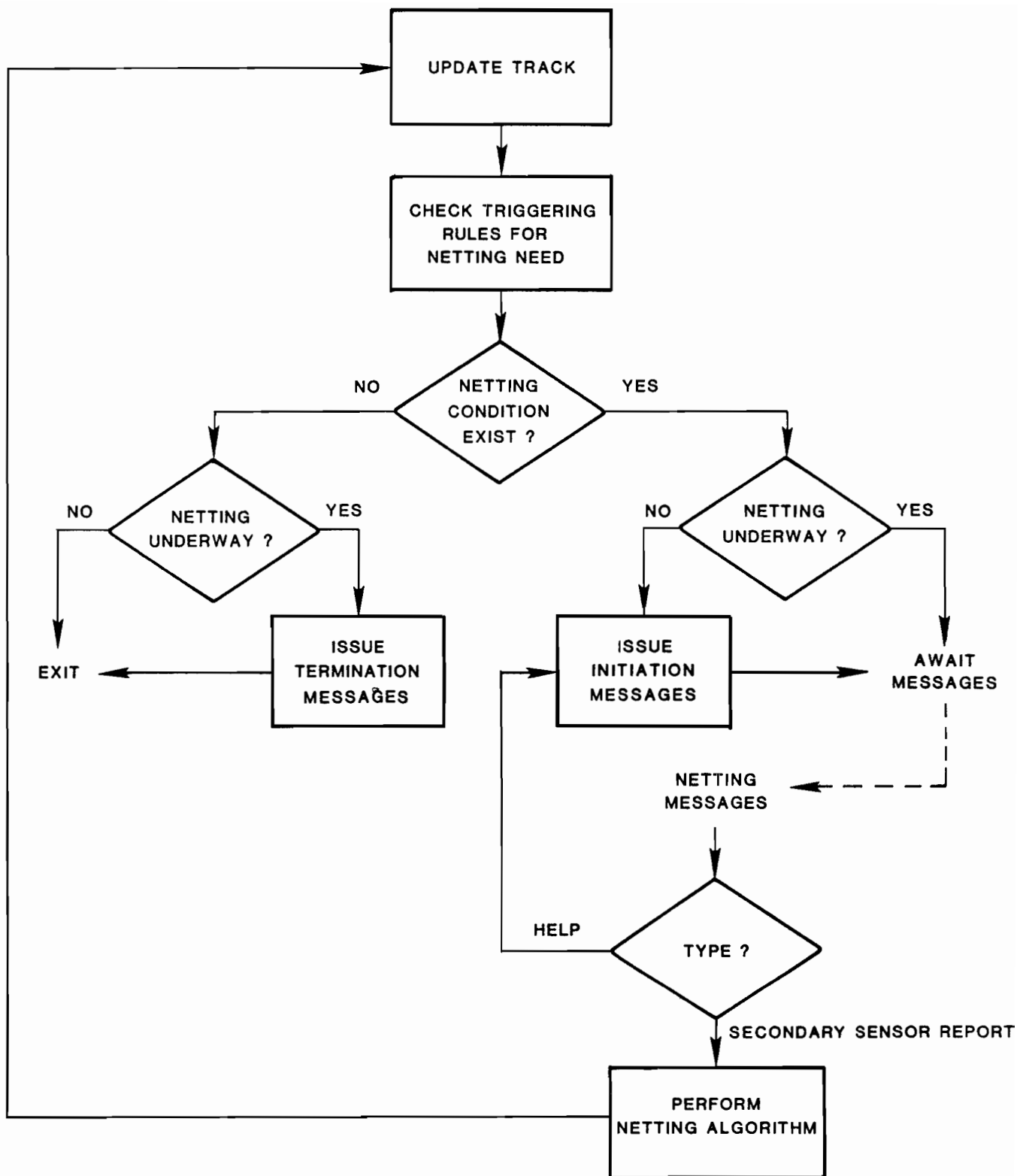


Fig. 4-8. Netting system flowchart.

4.4.1 Triggering Rules

The function of the request triggering routine is, for each track after its update, to place a value in the first field of its track status array entry which indicates the netting algorithm to be performed for that track. A sample list of values and interpretations for this netting condition field might be:

- 0: no netting required
- 1: newly initiated track
- 2: position improvement for conflict alert
- 3: position improvement for diffraction
- 4: false track screening
- 5: code improvement
- 6: altitude determination
- 7: track swap prevention
- 8: correlation resolution
- 9-15: other as yet undefined problems

The case "no netting required" will hopefully dominate, or else sensor processing might exceed its computers' capabilities.

Every newly initiated track for which a sensor has coverage responsibility should be entered into the netting domain. Two major reasons exist for this requirement. First, inter-sensor correlation is often most difficult just when the real netting need arises. Report garble or report matching uncertainty due to closely spaced aircraft tend to increase in conflict situations. By performing the inter-sensor track pairing before it is required, the correlation process is immeasurably simplified when the real problems arise (see next chapter). Second, false alarm tracks can arise for many reasons other than reflections. By checking each new track to see if the neighboring sensor also sees an aircraft at that position, track validity can be established early in its existence.

Conflict alert netting is triggered when two aircraft are approaching each other. The signal for this occurrence must come from the conflict alert user itself. The exact nature of the inter-connection between user and sensor to permit this reverse flow of information is presently undefined.

The sensor itself, on the other hand, can detect when netting is required for overcoming diffraction or false track screening. In each case, a site dependent table of azimuths is resident in the sensor to permit the identification: diffraction zones for the former, reflector zones for the latter. Any track existing in or entering either a diffraction or reflection zone will be marked by the triggering routine.

The sensor can also note when code or altitude improvement help is required from a secondary sensor. In either case, the absence of a high confidence code is the signal. In addition, altitude help is sought when aircraft without encoding altimeters are encountered.

Finally, netting help is sought whenever various possible target/track correlation errors are likely. The correlation algorithm within the sensor will know when it had a difficult correlation choice, and thus might have performed a track swap or track capture. It must signal these possible events via bits in the track file. This algorithm enhancement and the track file bits are both additions to the current stand-alone sensor code. Fortunately, both additions are simple to implement.

4.4.2 Request Generator

The request generator is responsible for ensuring that request messages are transmitted to the proper neighboring sensors, and at the proper times, to produce the secondary sensor reports needed by any tracks in netting situations. It consults three sources of data in determining when and where to send requests:

1. the netting condition field, as set by the triggering routine, in each updated track's status array entry
2. the netting status information in this same entry
3. a sensor coverage map.

The track status array was described above.

The request generator proceeds as follows. First it checks the netting condition of each updated track. If it is null, and the corresponding netting status entry is also null, no further action is required. If on the other hand, the netting status entry is not null, it must send termination messages to each sensor designated in the entry. The entry is then nulled. It is assumed here that all tracks dropped by the sensor are passed to the triggering routine so that any outstanding requests for them can be cancelled by this mechanism.

When the netting condition field of the updated track contains a non-zero value, the request generator checks the current netting situation as represented by the second and third fields of the track's status array entry. If any first subfield is non-zero (indicating an ongoing netting condition), and the response bits for each non-zero sensor are set to "1", no action need be taken, as the netting data stream is already underway. However, if the response bit for any non-zero sensor is still at "0", it must be assumed that the previous request message was either lost or not able to be processed by the sensor. Thus another netting initiation message must be transmitted.

Finally, if no netting is currently underway for the track, the process must be initiated. First the sensors to be asked for help must be identified. By assumption, this information is obtained by a sensor coverage map that might, for example, be represented as shown in Fig. 4-9. Here, we have divided the coverage region into ρ , θ , h cells. For each such cell, a secondary and tertiary sensor can be specified. Then an initiation message is sent to each specified sensor, the sensor numbers are recorded in the track status array, and the response bits are initialized to the "0" state.

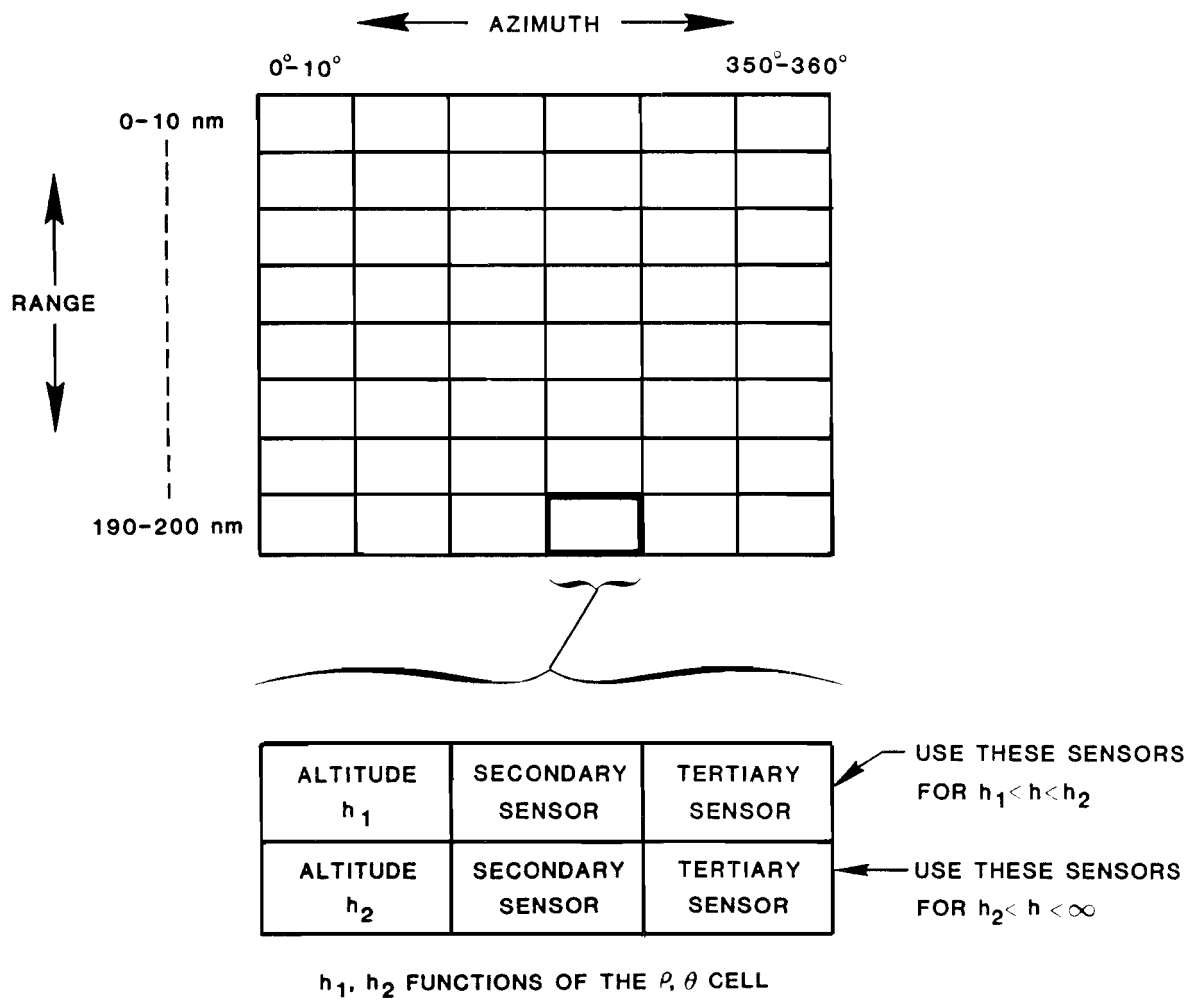


Fig. 4-9. Netting coverage map.

4.4.3 Response Processor

The response processor checks the stream of netting reports that arrive from other sensors in response to the messages issued by the request generator. In the normal case, its function is simply to enter each report into the netting report buffer, establishing or splicing into the chain of links emanating from the proper local track file entry. This entry number is specified in the report itself.

This normal case applies when the netting report was expected. That is, when the responding sensor number exists in the track status array entry for the track. If this check is satisfied, the above action is taken. Also, the appropriate response bit in the track status array entry is set to "1", recording the fact that the report was received.

If the report was not expected, however, a correction action must be taken. This event will occur either when inter-sensor timing results in a report being sent after a termination message was issued, or when this message was not properly received. In either case, the request generator is instructed to re-issue the termination message. The received report, of course, is discarded.

The response processor must also handle the occasional help message found interspersed among the netting reports. As discussed above, this message is a request by a secondary sensor for a new position report on the aircraft. The request processor produces the desired effect by zeroing the response bit for the corresponding secondary sensor in the track status array. As described in the last subsection, this forces the transmission of a new initiation message at the next track update time.

4.4.4 Netting Subsystem

The netting subsystem performs the actual netting algorithm for the track. Each time a track is to be updated, this process checks whether a netting action is signalled by the track status array entry. If so, all the auxiliary sensor netting reports (if any) are gathered together and grouped with the local sensor report (if any). These reports are then entered jointly into the proper netting algorithm (multilateration, code improvement, correlation verification, false track flagging, etc). The resultant report generated by this algorithm is then employed for the actual track update action.

The netting subsystem also performs a bookkeeping action, namely it zeroes the pointer field in the track file and all links in the netting report buffer for the track. While doing this, all netting report buffer entries occupied by the reports are returned to the free storage list.

4.5 Support System Algorithms

The support system consists of the routines to supply reports to other sensors to support their netting actions. The heart of this system is the inter-sensor correlation routine. Since this routine is the most complex in the entire netting system, and as it applies to centralized and hybrid architectures as well as to localized ones, its presentation has been reserved for a separate chapter.

The request processor precedes the correlator. Its function is simply to move requests from the I/O buffer to the circular request buffer, where they await correlation processing. This transfer is required to free the I/O buffer for the next batch of requests. The request processor is a high priority foreground task that interrupts the correlation activity whenever a new request arrival is signalled.

The final support function, the response generator, is responsible for sending to the requesting sensor local target reports that meet its netting needs. Such reports are those correlating to local tracks which the inter-sensor correlator has matched to the requestor's tracks. The inter-sensor correlation array (Fig. 4-7) is utilized for this purpose.

Each time a track is updated, the response generator examines that track's entry in this array. If any of the first half pair of fields has a non-zero track number, this indicates that another sensor requires the report correlated to this track. The generator then constructs the proper report message, using this other sensor track number, and transmits it to that sensor.

The response generator has two other responsibilities. First, whenever a local track is dropped, it must clear the first half fields from the inter-sensor correlation array. Second, the inter-sensor correlation routine may occasionally require new data from another sensor. The generator, in such a case, must prepare a special message to relay this fact to the appropriate sensor. This message is of the same format as a standard report except that the special "help" bit is set to "1" (see Fig. 4-3).

5.0 INTER-SENSOR CORRELATION

Netting commences with a request by one sensor for surveillance reports on a particular aircraft from an adjacent sensor. In general, the sensor receiving this request will contain a track corresponding to that aircraft in its files. The problem for this sensor is to determine which specific track is the one in question. This process of matching a track in one sensor with that in another is known as inter-sensor correlation.

This correlation problem is generally very simple for Mode S or discrete coded ATCRBS aircraft, as the unique code serves as an excellent track discriminant. In the rare cases when two or more tracks exist with the same code, due to reflection false targets or an assignment error, the aircraft position becomes the deciding factor in the selection.

Non-discrete ATCRBS aircraft, particularly those not possessing encoding altimeters, present a considerably more complex inter-sensor correlation problem. It would not be surprising at all for two or more tracks to satisfy all first-order correlation requirements.

This chapter presents correlation algorithms for both the discrete and non-discrete situations. At present, neither algorithm has been tested on real data, so modifications or additions may be required later.

5.1 Correlation Philosophy

An inter-sensor correlation system that will be applicable to all cases must, by nature, be quite complex. It must contain, in addition to the obvious code and position tests, various second-level tie-breaking procedures to resolve ambiguities, as well as likelihood tests to judge the feasibility of a match (particularly when altitude, and thus exact aircraft location, is unknown). As a result, the prospect of an occasional incorrect correlation is a reality that would have to be accepted.

Single sensor surveillance by a Mode S sensor is quite good. Netting would be utilized only to improve its quality in critical situations or to help resolve an occasional uncertainty. Thus the possibility of an incorrect correlation is unsettling. Such an occurrence would convert good surveillance to bad surveillance, and uncertainty to error. No netting algorithm can survive the transmission of data for the wrong aircraft.

Besides degrading performance, incorrect correlations would remove sensor guard actions against error. With uncertainty, the sensor would have taken precautions; once it thinks it knows the situation as a result of netting help, it can follow wrong and possible dangerous actions with impunity. This realization has led to the adoption of the following strategy:

The inter-sensor correlation algorithms shall be simple and straight-forward, with no second-order testing. Any situation too complex to be resolved by such an approach shall be left unresolved, with no netting data passed back to the requesting sensor.

5.2 Discrete Correlation

Tracks for Mode S and discrete ATCRBS aircraft can always be located via their code, as this is a requirement of the sensor implementation. Thus the hardest part of the general correlation process, filtering the track file for possible matches, is trivial in this case. The result of the code search produces none, one, or more than one track with the desired code. If none, inter-sensor correlation has failed, and no netting help can be forthcoming at this time. Since the netting request will be repeated each scan, however, future help is possible should a track later become initiated for the aircraft.

When two or more tracks are identified as code matches, the proper one must be selected via position agreement. Even if only one track exists with the proper code, however, the position test should still be applied. This safety rule prevents the rare system error from producing incorrect correlations.

This position test will be employed either to select from two to more spatially diverse tracks, or to check the validity of the single possible track. Thus it is really only a reasonableness test. For this reason, it need not use mathematically precise procedures. Thus, although the exact coordinate transformation process developed in the next section for non-discrete aircraft would work here as well, the possibility of simpler tests to save processing time should not be ignored.

One particularly simple test is illustrated in Fig. 5-1. If it is assumed that the earth is flat (nearly true over small distances), and that ground range and slant range are equal (nearly true except near a sensor), the primary sensor's azimuth to the aircraft can be computed by multilateration from the ranges in the request message and the track file for the local sensor track in question, and the known inter-sensor distance and angle, d_{12} and ϕ_{12} respectively. Then the comparison of this computed azimuth with the actual azimuth in the request message constitutes the reasonableness check.

The details of this procedure are as follows. First, extrapolate the request message position to the time of validity of the local sensor track position:

$$\hat{\rho}_1 = \rho_1 + \dot{\rho}_1 T$$

$$\hat{\theta}_1 = \theta_1 + \dot{\theta}_1 T$$

where T is the extrapolation interval and the rates are contained in the request message. Then compute the angle ψ_1 within the triangle formed by the inter-sensor line and the two sensor ranges (see the figure):

$$\psi_1 = \cos^{-1} \left[\frac{\hat{\rho}_1^2 + d_{12}^2 - \rho_2^2}{2 \hat{\rho}_1 d_{12}} \right]$$

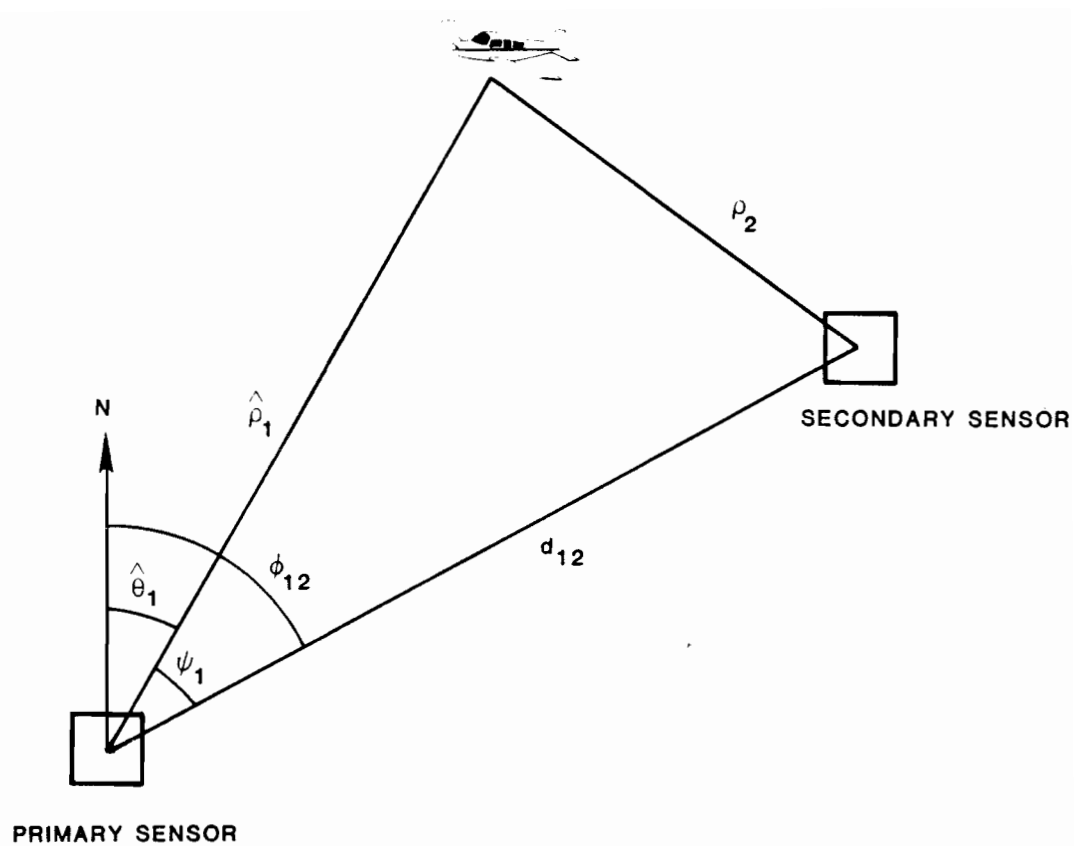


Fig. 5-1. Discrete correlation geometry.

Finally, the track under test is the proper inter-sensor match if:

$$||\hat{\theta}_1 - \phi_{12}| - \psi_1| < \theta_{reas}$$

Whether this simple test works in practice cannot be answered with certainty without real data. It is possible that ground ranges, given by:

$$\rho_g = \sqrt{\rho^2 - z^2}$$

will be required for aircraft near a sensor. Even with this enhancement, the test is far quicker than one using coordinate conversion.

5.3 Non-Discrete Correlation

Non-discrete tracks cannot be accessed by their codes. Even if such access were desired to be added for netting, it would not be feasible. This is because low confidence code bits allow for the possibility of many codes matching a track; in the worst case, any code could form a match. Thus the type of algorithm presented above for discrete tracks, namely track set identification by code followed by a position test, would require a linear search of the track file and a large number of tests.

Fortunately, extensive Mode S sensor testing has shown that a sensor almost always maintains the same track number on an aircraft throughout its time under coverage. This fact can be used to advantage whenever netting is required for the second or subsequent time on an aircraft. The inter-sensor correlation array, indexed by the track number in the new request message, will provide the local track number that formed the match on the previous netting activity. Thus this track can be tested to see if it still represents the aircraft; if so, much time is saved.

First time netting, of course, cannot utilize this shortcut. Instead, some form of area searching of tracks is required. The one selected for inter-sensor correlation takes advantage of the already existing intra-sensor target/track correlation mechanism. In particular, the request message target report is converted to a form that makes it appear to be a locally generated target, and then it is entered into this normal correlation process. The tracks that it associates with then constitute the set of possible matches. If one, and only one, of these tracks meets the inter-sensor match criteria, the search is deemed a success.

The details of this overall inter-sensor correlation algorithm for non-discrete ATCRBS aircraft is outlined in the flowchart of Fig. 5-2. The first step is to transform the request message position report to the position and time coordinates of the local sensor. As detailed in Appendix A, this transformation has two parts. First the position is converted:

$$\rho_1, \theta_1, \dot{\rho}_1, \dot{\theta}_1 \text{ at } T_1 \rightarrow \rho_2, \theta_2, \dot{\rho}_2, \dot{\theta}_2 \text{ at } T_1 \quad (5-1)$$

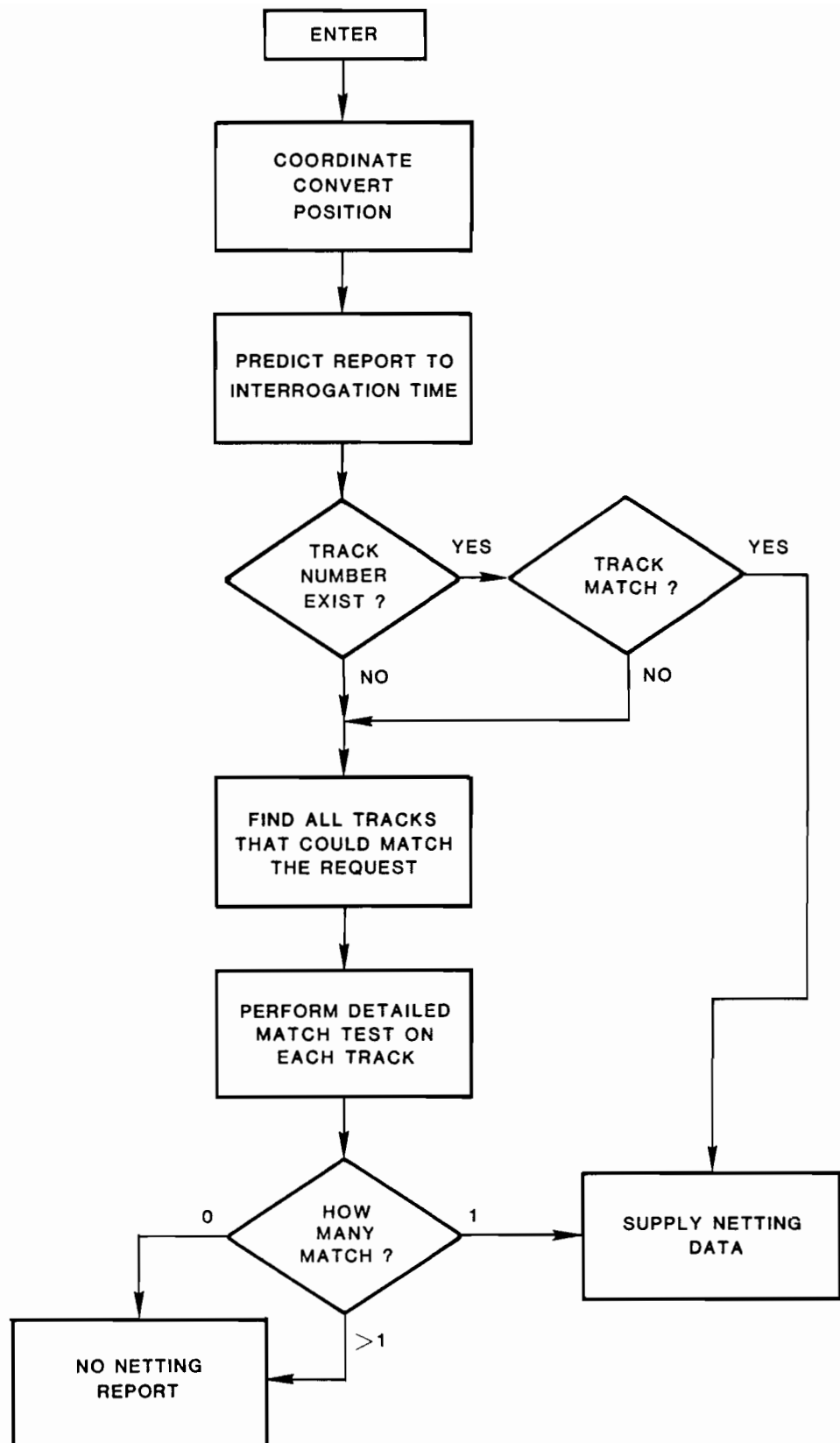


Fig. 5-2. Non-discrete correlation flowchart.

where the hat indicates local sensor coordinates at the requesting sensor viewing time T_1 . Then the report is extrapolated ahead to the time T_2 at which the local sensor would see the aircraft it represents:

$$\hat{\rho}_2, \hat{\theta}_2, \dot{\hat{\rho}}_2, \dot{\hat{\theta}}_2 \text{ at } T_1 \rightarrow \rho_2, \theta_2 \text{ at } T_2 \quad (5-2)$$

The value of T_2 , as shown in the Appendix, is computed by using the report and sensor antenna azimuth rates.

Both of these conversion steps require knowledge of the altitude of the aircraft. Thus, if the request report has no usable altitude, either because it is missing totally or is garbled, a guess of the true altitude must be made. In the former case, a value of 0.5 miles is assumed, as aircraft without altimeters normally fly quite low. In the latter case, unfortunately, any altitude is reasonable. Subject to further testing, a value of 2 miles appears to be a reasonable compromise. In either case, a more accurate conversion will be performed should a match be made with a track having altitude knowledge, as described below.

The next step of the algorithm depends upon whether or not this netting request is a repeat. If so, the track matched last time has its predicted position and codes matched with the converted report. If the Mode A codes and altitudes agree, and the positions satisfy

$$\begin{aligned} |\Delta\rho| &< \Delta\rho_{\text{reas}} \\ |\Delta\theta| &< \Delta\theta_{\text{reas}} \end{aligned} \quad (5-3)$$

the match is retained, and the correlation procedure is terminated.

When the above shortcut is not applicable, either because no previous correlation has occurred or the match has failed (signifying a rare track swap condition), the converted report is placed in a temporary buffer. It is stored there until time T_2 arrives, at which time it is added to the stream of locally generated target reports. It is then automatically processed by the target/track association routine. The track associates found constitute the set of possible correlation matches. If the set is null, of course, inter-sensor correlation has failed.

At this point, the match tests could be made against each potential track. However, a wrong altitude guess (when altitude was not known), or extrapolation or time transformation errors could have produced an incorrect report position with which to match against the tracks. Thus a more accurate procedure has been adopted. First, to generate updated track positions, the local target/track correlation procedure is allowed to complete (minus the extra request reports, of course). Then each potential matching track has its new position interpolated back to time T_1 . If its Mode A code and altitude

agree with the request, and its position difference from $\hat{\rho}_2$ and $\hat{\theta}_2$ of (5-1) satisfies (5-3), a match is recorded. In the event the request report had no known altitude, and thus $\hat{\rho}_2$ and $\hat{\theta}_2$ were generated by an altitude guess, and the track to be matched has a known altitude, the quantities $\hat{\rho}_2$ and $\hat{\theta}_2$ are recomputed using this altitude prior to this difference test.

Inter-sensor correlation will be ruled successful if one, and only one, potential matching track satisfies this test. No attempt is made to arbitrate between multiple matches in accordance with the strategy presented above.

6.0 POSITION AND HEADING IMPROVEMENT

The major emphasis of the surveillance netting project has been the development of algorithms and formulas for improving the tracking accuracy of aircraft by the Mode S sensor. This increased accuracy is derived from two interrelated sources: improved three dimensional position data included in reports produced as surveillance output, and improved smoothing and heading estimations for projecting future positions. The next four chapters present the results of this segment of the project effort, while this chapter serves as an introduction to the various approaches to be described.

6.1 Azimuth Improvement

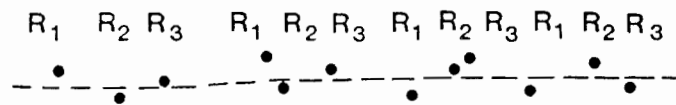
The inaccuracies in sensor position reports are usually confined to the azimuth coordinate. Thus the most widely employed method for positional improvement with multiple sensor data is multilateration, which is the use of two or more range measurements to derive the azimuth of the aircraft. However, a number of competing regimens can be visualized to serve to improve the azimuth accuracy. The three considered in this project were higher data rate tracking, curve fitting, and incremental bilateration. Various combinations of these approaches, where appropriate, were also examined.

Figure 6-1 presents in a simple pictorial manner the three classes of approaches under consideration (incremental bilateration is a special case of multilateration). With higher data rate tracking, the individual sensor reports are entered unmodified into the smoothing filter. Although the reports are no more accurate than with single sensor surveillance, the greater number of them should produce a smoother, and hence more accurate, track. Thus both current position and heading estimates may be improved.

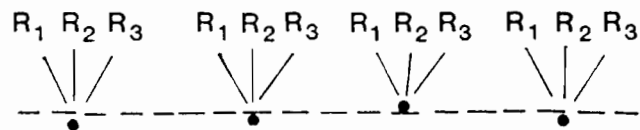
Multilateration employs only the range measurement of each sensor report. Two or three such ranges per scan are used jointly in a formula that determines the true aircraft azimuth. In order to use this approach, of course, all range measurements must be interpolated or extrapolated to a common time. This process will introduce some errors, especially when the aircraft is turning.

Finally, curve fitting is a batch approach to tracking. First a group of consecutive-in-time sensor reports are fit by a linear or quadratic curve. This curve may be taken as the smoothed track. Or, for greater accuracy, the position values at the center of this curve (the most accurate point) can be taken as if they constituted a sensor report and fed into a smoothing filter. In the figure, these reports are represented by the heavy dots. This approach would appear to be more accurate than straightforward high data rate tracking.

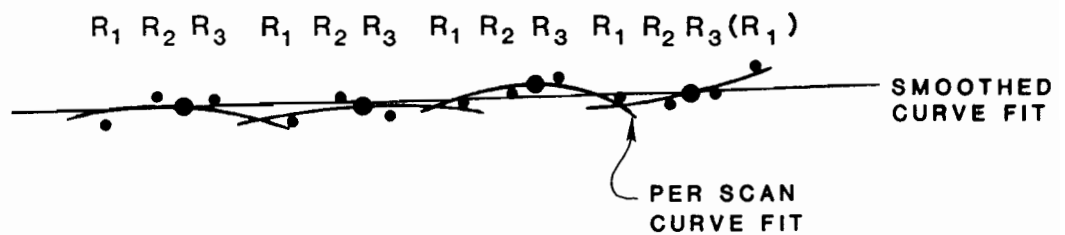
A comparison of the strengths and weaknesses of these competing approaches is outlined in Fig. 6-2. As expressed therein, the high data rate approach is the simplest and most general purpose, and the only one that can utilize directly single sensor tracking algorithms. However, it does not improve the data fed to the tracker. Multilateration, on the other hand, is complex, but is the only method that does not require aircraft altitude, or



HIGH DATA RATE TRACKING



MULTILATERATION



CURVE FITTING

Fig. 6-1. Tracking improvement approaches.

ATTRIBUTE	TRACKER TYPE		
	HIGHER DATA RATE	MULTILATERATION	CURVE FITTING
COMPLEXITY	LOW	HIGH	MEDIUM
DATA IMPROVEMENT	NONE	YES	OPTIONAL
ALTITUDE	REQUIRED	COMPUTABLE	REQUIRED
COORDINATE CONVERSION	REQUIRED	NOT REQUIRED	REQUIRED
AZIMUTH	REQUIRED	COMPUTABLE	REQUIRED
ERROR SENSITIVITY	MODERATE	HIGH	LOW

Fig. 6-2. Tracker comparisons.

any coordinate conversions, and it improves the data quality. Curve fitting, in general, falls midway between these other approaches on almost all criteria.

This and the next several chapters develop all these approaches in detail. At the end of the chapter on smoothing filters, the results of actual comparisons are presented. Not surprisingly, there is no clear cut winner, as different situations affect the approaches differently.

6.2 Overcoming Diffraction

When diffraction zones are present at a sensor, its azimuth measurement will be virtually useless for aircraft flying through them. Thus, netting would be utilized to generate an alternate determination of this measurement. Obviously, netting methods that do not utilize the sensor azimuth would be unaffected by diffraction. Approaches that do utilize it will be degraded to some extent, but may still yield acceptable performance.

Standard multilateration employs only range measurements to compute the aircraft azimuth. Thus this method is ideally suited for overcoming diffraction. However, some of the more complex multilateration formulas presented in Chapter 8, which account for the presence of a transponder bias or the absence of the aircraft altitude, do require one or more azimuth measurements. Data presented in that chapter is used to determine whether these approaches are still feasible in the presence of diffraction.

Incremental bilateration, as presented in Chapter 9, always uses the primary sensor azimuth in its calculations. Thus this method would appear to be unsuitable in diffraction. However, a modification to the general procedure is introduced which is shown by simulation results to handle diffraction zones with negligible loss of accuracy.

Finally, both the higher data rate tracking and curve fitting classes of netting cannot use raw reports from the primary sensor, with diffraction-corrupted azimuth, as inputs to their smoothing filters. Chapter 10 examines whether the approach of omitting primary sensor reports from the filter input stream, and thus using only non-diffracted secondary sensor reports, provides adequate data for good smoothing performance.

6.3 Bias Error Effects

When bias errors are present, either in the sensor system or in the aircraft transponder, all three classes of netting approaches are affected. The degree of resistance to bias may well be the most important selection feature among the approaches, as netting improvement can easily turn to degradation in biased environments.

Higher data rate tracking counts on the introduction of more input data points to aid in trajectory determination. If these additional points are biased, however, they could easily result in noisier, not smoother, tracks. Thus only trackers that are bias-resistant should be employed for this approach.

Multilateration operates under the assumption that the intersection of range arcs from two or three sensors locates the aircraft position. When biases are present, the "range line" from the affected sensor(s) will either undershoot or overshoot the true position. There are three ways in which a new position can be found that satisfies the intersection criterion, as shown in Fig. 6-3.

First, if transponder turnaround delay is a system variable, the lines can all be adjusted in length until an intersection is achieved. If this delay was indeed the major system bias, the proper position will be located by this process. Second, if altitude is not known from an altimeter readout, the lines can be raised or lowered until an intersection is reached. The correct x,y position can thus be achieved at the expense of an altitude estimation error. Finally, if neither transponder delay nor altitude is a variable, the only option is to move the lines laterally until an intersection is achieved. This, unfortunately, translates the entire bias error into an azimuth error. Chapter 8 presents and examines algorithms that implement each of these three approaches.

Incremental bilateration, another form of multilateration, deals with biases by attempting to eliminate any effect of them beyond that seen in single sensor surveillance. Thus consistency, with lower variance, is its goal. Chapter 9 develops an algorithm that successfully meets this goal.

Finally, curve fitting is a natural method for overcoming bias problems. By "averaging" the errors from different sensor reports, truth may be determined. At least consistency, similar to that of incremental bilateration, should be achievable. The only problem arises when more reports are input to the curve fit routine from one sensor than from another. Then the result will be weighted toward that sensor's biases. If the sensor with the most reports varies from scan to scan, inconsistencies can result. Chapter 10 presents results for curve fitting accuracy in the presence of biases.

6.4 Incremental Bilateration

The standard method of envisioning a two-sensor surveillance system is as follows (Fig. 6-4). The locations of the two sensors are known exactly. Each sensor produces a relative position measurement of the aircraft. The two resulting absolute positions are identical, and specify the location of the target.

In reality, as discussed in the previous section, measurement noise and system biases will cause the two positions to differ from each other. Noise, being random, will be seen as a jitter of the two measured positions about the true aircraft location. Biases, on the other hand, will cause the centers of mass of the two sets of jittered measurements to lie at different locations.

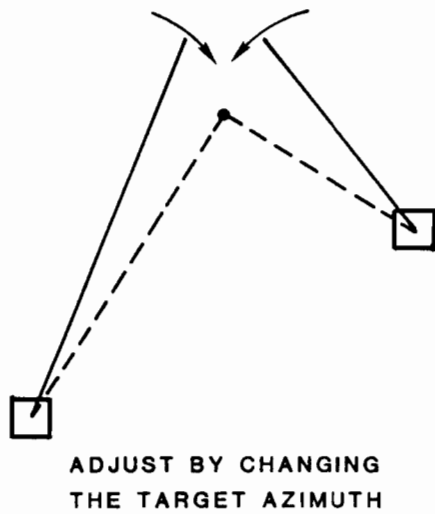
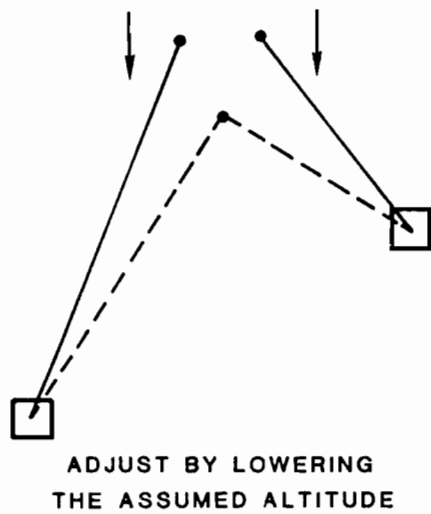
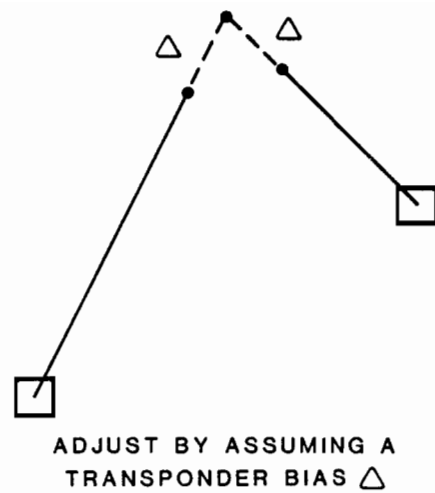
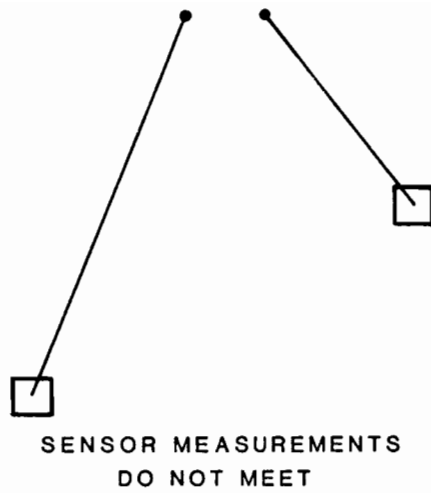


Fig. 6-3. Multilateration bias adjustment.

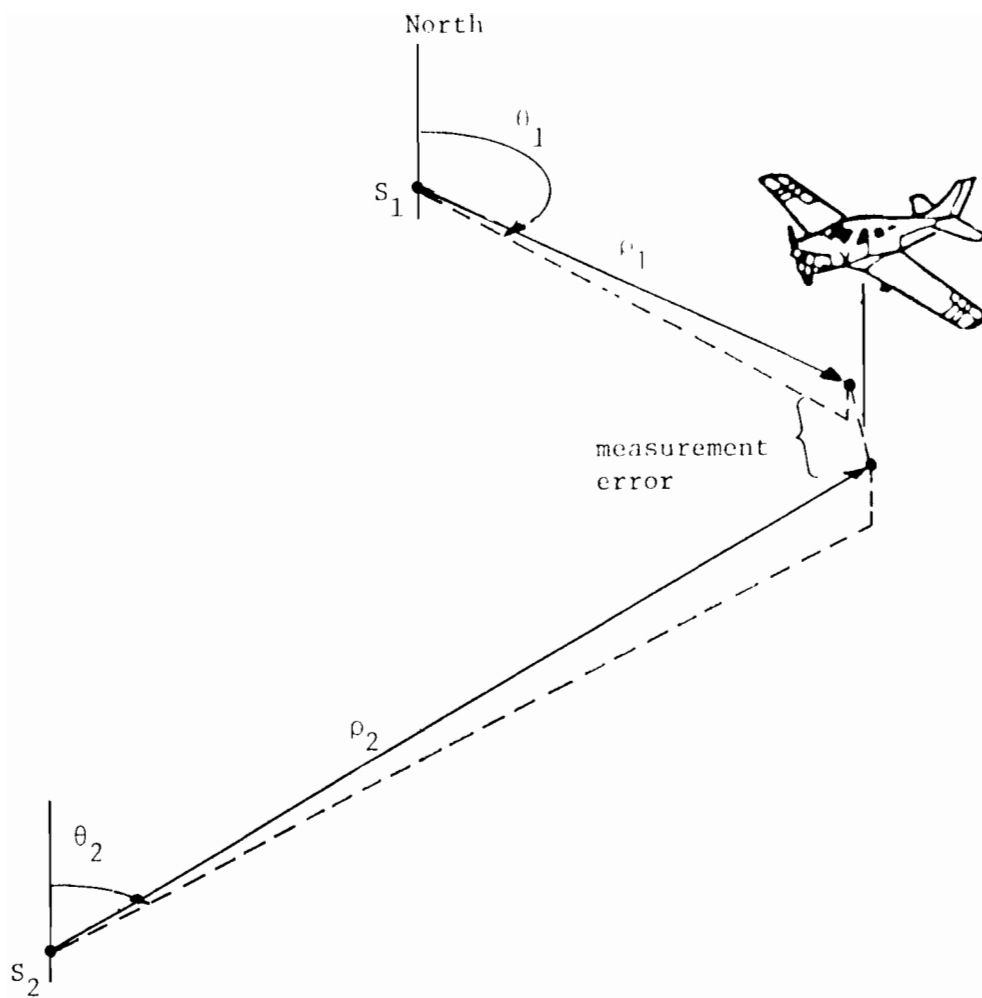


Fig. 6-4. Normal 2-sensor scenario.

Thus, even if the measurements were noise free, biases would still cause the two sensors to disagree on the aircraft location. The multilateration location, obtained by employing range measurements from both sensors, will be at yet a third position. This position, the intersection of the two range arcs, will not be at the average of the other two; in the worst case, it can be many miles from either of them. These observations explain why the jumps illustrated earlier in Fig. 2-5 occur whenever the data source changes from one scan to the next.

An alternative view of this two-sensor system is also possible (Fig. 6-5). Consider only the location of the primary sensor to be known. Its relative position measurement locates the aircraft. Then taking the measurement of the secondary sensor, and "reversing" it, will lead to the apparent location of this sensor. Noise and biases under this view result respectively in jitter and shifts of the secondary sensor location from its true position.

With this view, as long as the secondary sensor is assumed to be located at its bias-calculated position, the two sensors' measurements will be compatible in the absence of noise. Then switching data sources will produce no jumps. It should be noted that this common position of the aircraft will not be correct, only consistent. But, as discussed earlier in section 2.3, this is sufficient for conflict applications that depend upon prediction of future position.

This alternate approach can be viewed as a form of incremental netting (Fig. 6-6). Given that the target and secondary sensor positions on the previous scan are known, the consistent target position for the current scan is determined by bilateration as shown in the figure, using range changes from the previous scan. Since the two sensors agree on the target location, consistent results would also occur if either sensor measurement were absent, and the other sensor measurement were employed alone.

The bias-induced secondary sensor location will be a function of aircraft position. It will be different for each target, as well as changing for any given target over time as shown in Fig. 6-7. Fortunately, empirical evidence has indicated that the apparent scan-to-scan movement of the sensor is quite small. Thus, it is possible to employ this incremental approach, which assumes a constant position over small time intervals. Measurement noise will introduce jitter into the sensor motion that is large compared to this bias component. Thus it is necessary to average the calculated positions of the previous few scans and employ the result on the current scan. This averaging eliminates the measurement noise errors of a single scan's computation.

Although the apparent secondary sensor motion is very small, an example will demonstrate that it cannot be ignored. Figure 6-8 presents an aircraft flight path over a period of 200 scans. Assume that no sensor biases or measurement noise exists, but that the aircraft transponder has a turnaround error of 0.03 miles (well within specification). As a result of the changing geometry of the scenario, the bias-induced position of the secondary sensor will vary. Figure 6-9 in particular presents its distance variation from sensor 1; a similar figure would show the angle variation. Note that the maximum variation is only about 0.035 miles.

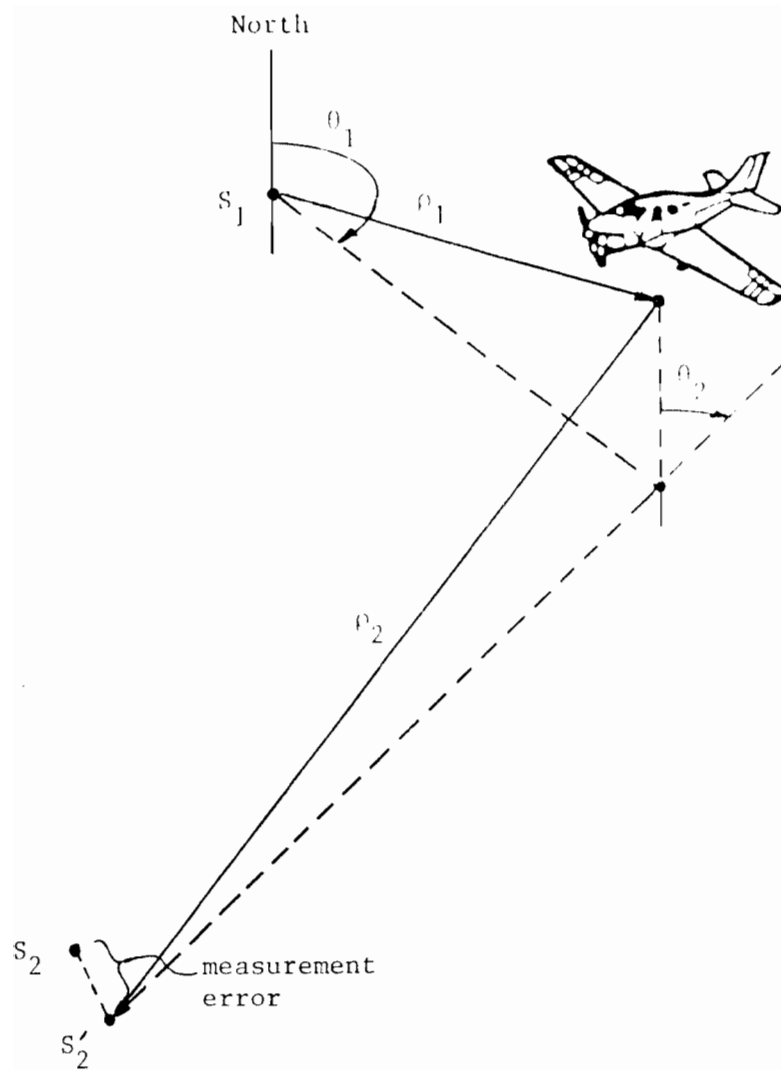


Fig. 6-5. Alternate 2-sensor scenario.

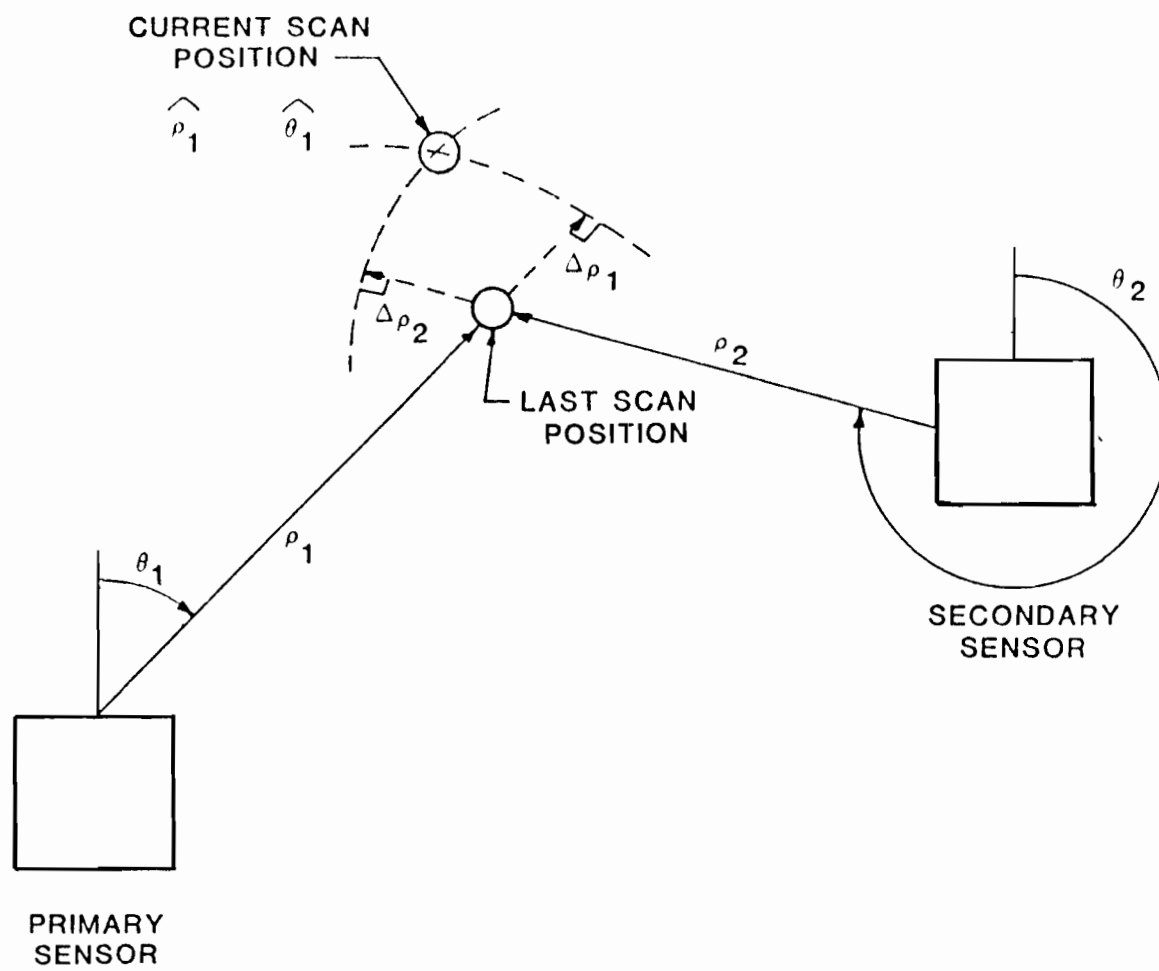


Fig. 6-6. Incremental netting.

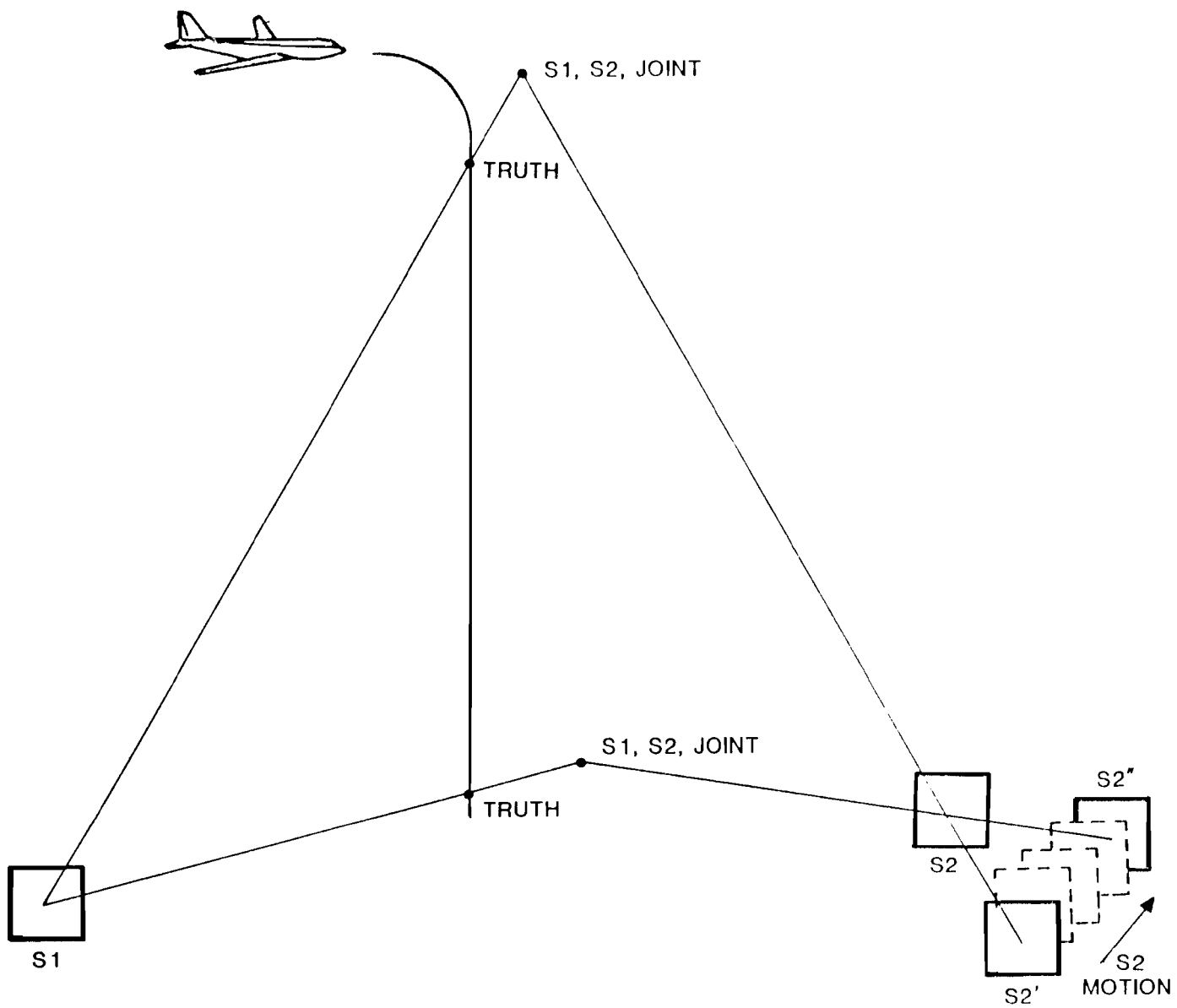


Fig. 6-7. Apparent secondary sensor motion.

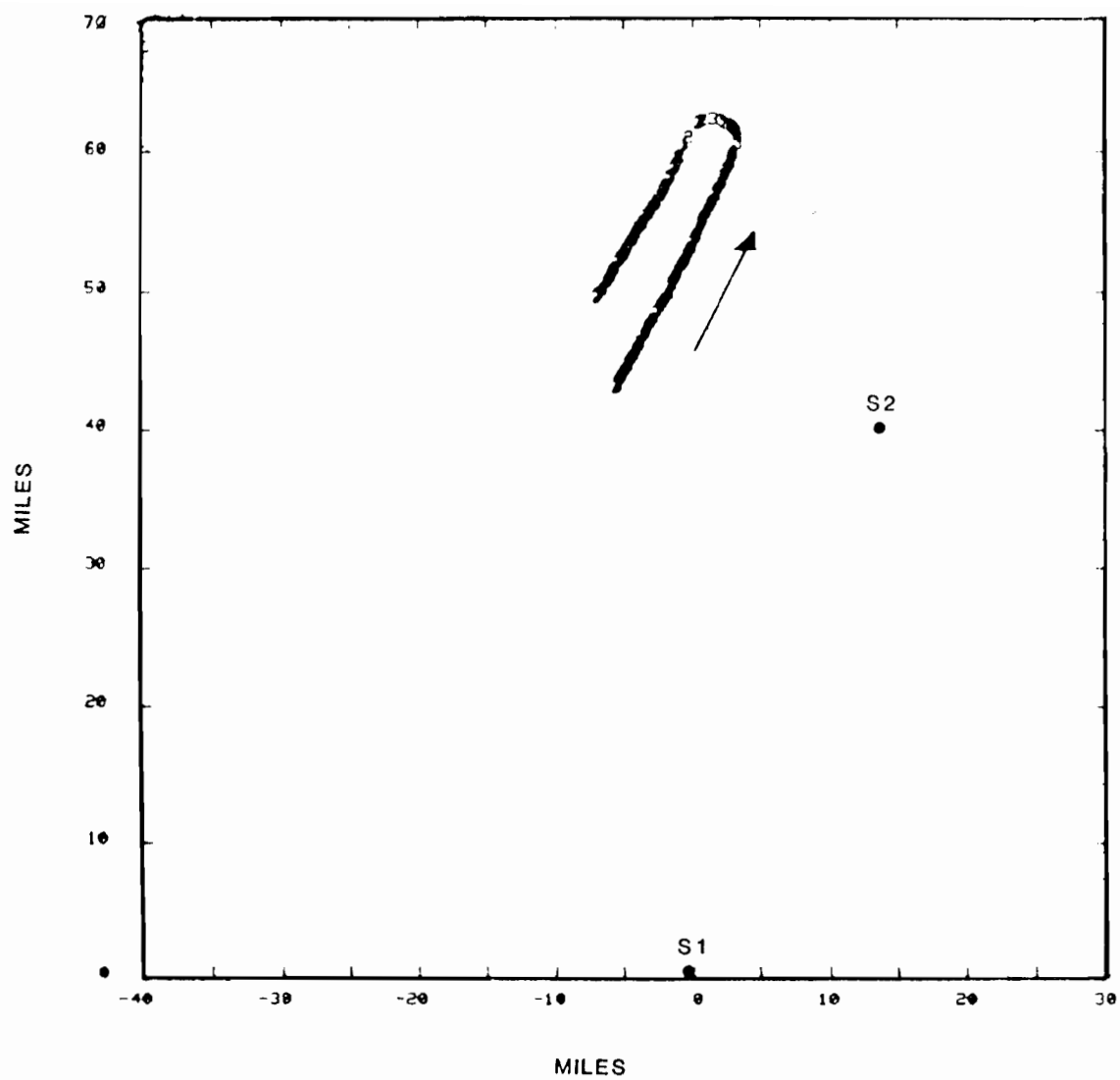
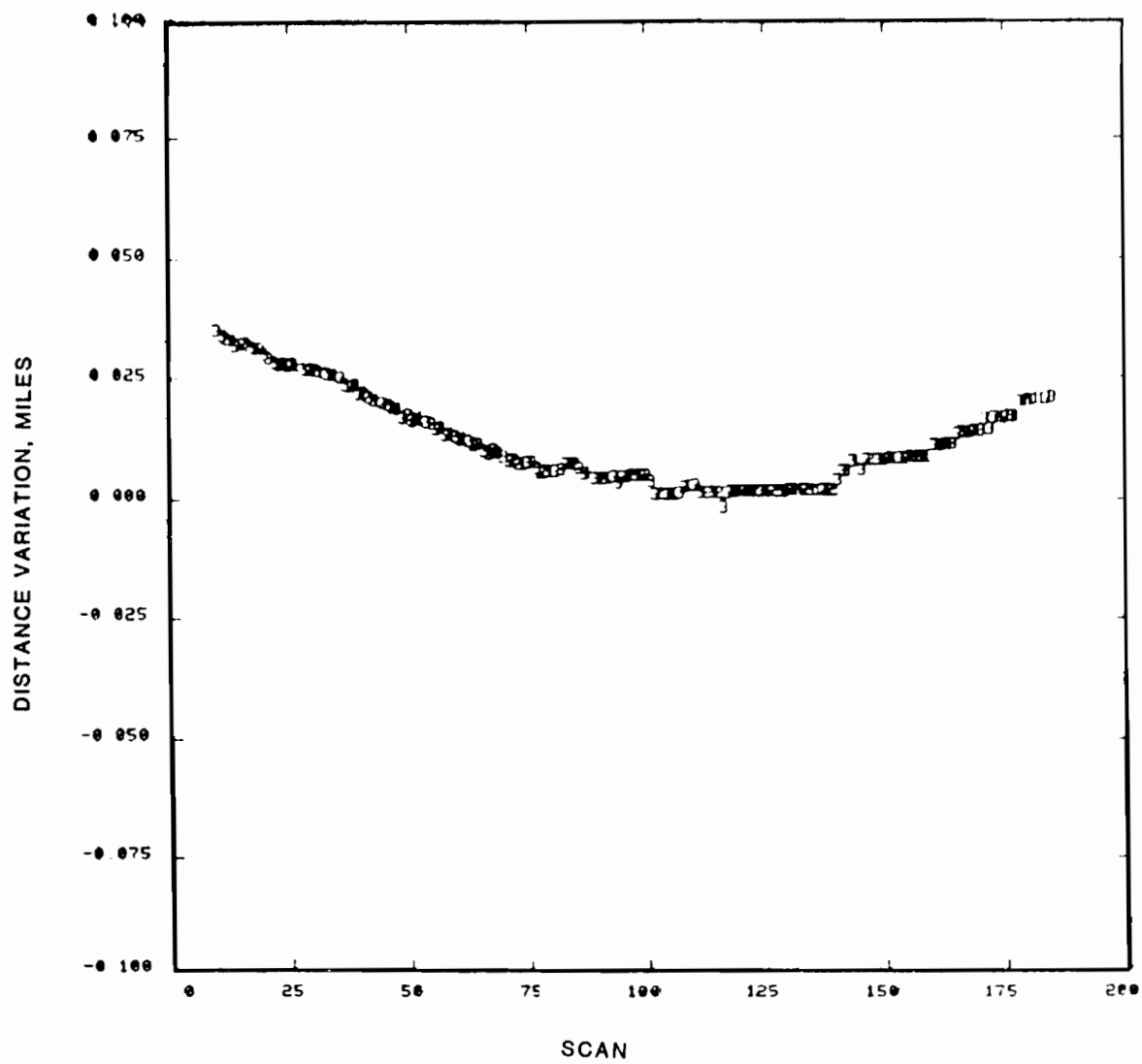


Fig. 6-8. Aircraft test trajectory.



NOTATION: SEE FIG. 2-6.

Fig. 6-9. Secondary sensor distance variation.

Figures 6-10 and 6-11 show the azimuth error versus scan for two methods of incremental bilateration: sensor 2 fixed at its initially calculated offset, and sensor 2 moving over time. The plotted numbers indicate the data source on each scan: sensor 1 only (1), sensor 2 only (2), or bilateration (3). It is clear from Fig. 6-10 that the mean errors for cases 2 and 3 diverge from zero as the geometry changes, and that significant data jumps are once again present for source changes. Figure 6-11, on the other hand, exhibits neither of these problems.

The requirement that the apparent secondary sensor position be computed for each aircraft on each scan implies that, for this algorithm to be feasible, the calculation must be reasonably simple. For the spherical earth model, the "reverse" measurement problem becomes:

Find the latitude λ , longitude ℓ , and height h_{s2} for sensor 2 for which

$$\begin{vmatrix} \sqrt{\rho_2^2 - z_2^2} \sin \theta_2 \\ \sqrt{\rho_2^2 - z_2^2} \cos \theta_2 \\ z_2 \end{vmatrix} = T(\lambda, \ell) \begin{vmatrix} \sqrt{\rho_1^2 - z_1^2} \sin \theta_1 \\ \sqrt{\rho_1^2 - z_1^2} \cos \theta_1 \\ z_1 \end{vmatrix} + U(\lambda, \ell, h_{s2}) \quad (6-1)$$

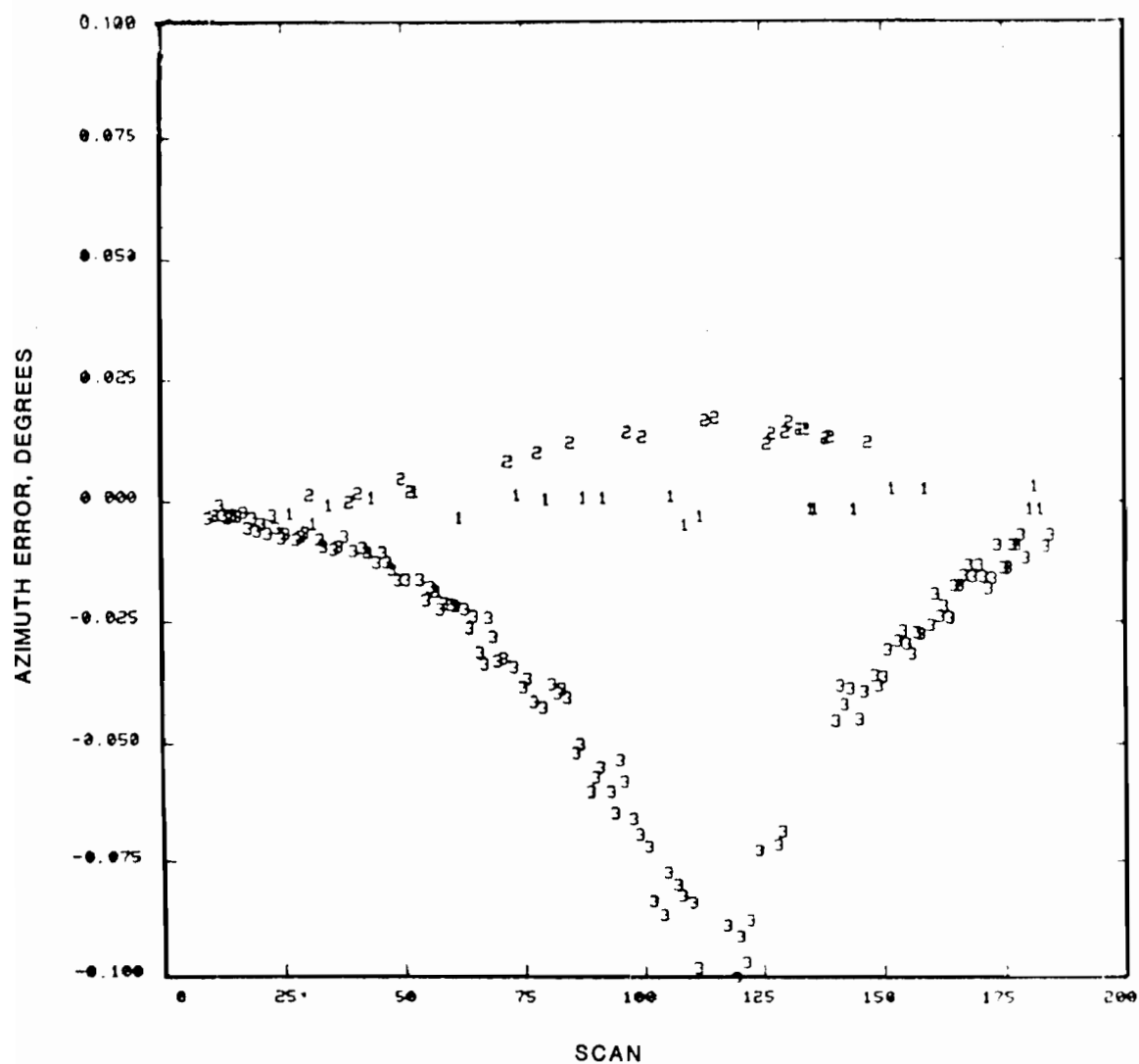
The matrices T and U are both nonlinear in their arguments, so solving this set of simultaneous equations is not possible except through iteration. If the system biases are assumed to be small, T and U can be linearized about the actual sensor 2 location. Then the direct solution of the simultaneous equations (6-1) becomes possible, but still not computationally feasible.

Matters would be much more promising if a flat earth model could be employed. As shown in Fig. 6-12, the geometry of the reverse measurement would then be quite simple. The biased distance D_b and azimuth ψ_{12}^b to the secondary sensor would be given by:

$$D_b^2 = \rho_{1g}^2 + \rho_{2g}^2 - 2\rho_{1g} \rho_{2g} \cos(\theta_1 - \theta_2) \quad (6-2)$$

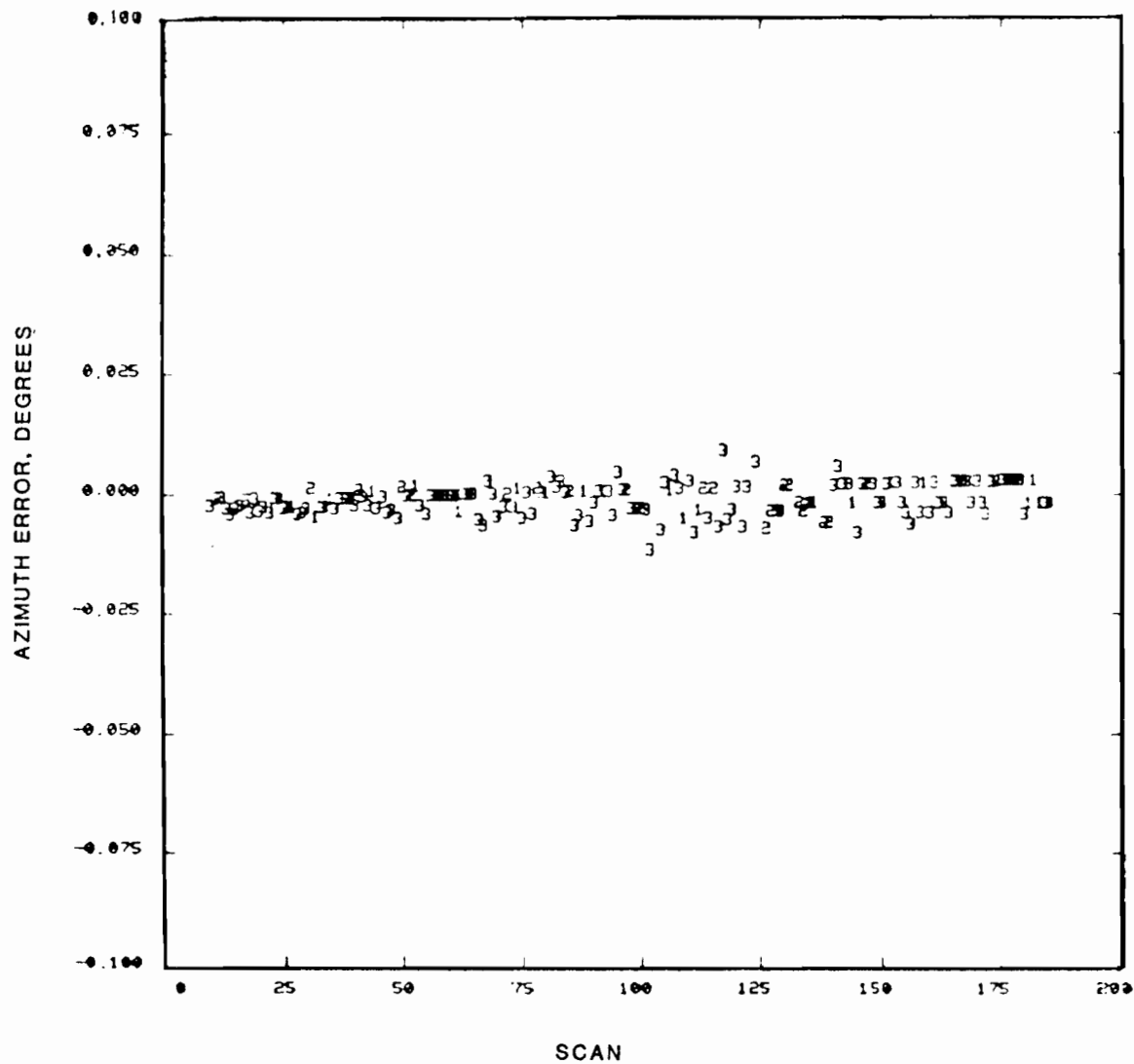
$$\psi_{12}^b = \theta_1 \pm \cos^{-1} \left[\frac{D_b^2 + \rho_{1g}^2 - \rho_{2g}^2}{2 D_b \rho_{1g}} \right] \quad (6-3)$$

where ρ_{ig} is the ground range from sensor i and the correct sign to use in (6-3) is the one producing an answer closer to the unbiased ψ_{12} . Chapter 9 develops a flat earth model that permits this simple approach.



NOTATION: SEE FIG. 2-6.

Fig. 6-10. Azimuth error for fixed offset.



NOTATION: SEE FIG. 2-6.

Fig. 6-11. Azimuth error for moving offset.

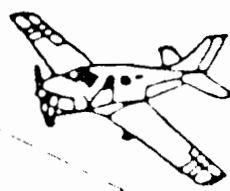


Fig. 6-12. Reverse measurement geometry.

The improved surveillance performance resulting from this algorithm can be seen in Fig. 6-13, where the same case as presented earlier in Fig. 2-6 has been reprocessed. The track is now smooth, with no jumps visible between different data source cases. Further statistical data demonstrating the superiority of bilateration with this bias-resistant extended incremental formulation over the normal spherical earth model is presented later in this report.

6.5 Transponder Turnaround Delay

An ATCRBS transponder is designed to begin transmission of its downlink response exactly 3.0 microseconds after it receives the uplink interrogation. Specifications permit working transponders to vary from this nominal value by up to 0.5 microseconds before requiring repair or recalibration. Since the ground sensor always assumes a 3.0 microsecond turnaround delay in its conversion from received signal time to aircraft range, the permitted variation can introduce a range calculation error of 0.04 miles (250 feet).

Mode S sensors have a measured range error standard deviation of about 25 feet. Thus the transponder delay bias error will be the dominant factor in range inaccuracies. Furthermore, since this bias will differ from aircraft to aircraft, it will affect the assumed relative positions of aircraft in potential conflict situations.

Fortunately, safe aircraft separations are much larger than 250 feet, so this positional error should not be significant. If the bias affected aircraft headings, though, conflict alerts under development could be misread. However, Appendix B demonstrates that heading errors due to transponder turnaround biases are negligible. In conclusion, single sensor surveillance is not compromised by the existence of imperfect transponders.

With multilateration, however, the transponder bias assumes new importance. This range error is now translated into an azimuthal position error for the report. The size of this error, for some geometries, can assume significant dimensions. It can also affect aircraft headings, especially if the data source varies from scan to scan.

The next chapters present algorithms for computing an aircraft's transponder turnaround bias in real-time from the multiple sensor data. Simulation tests then attempt to answer two questions. First, how accurately can this bias be determined, and second, how much improvement (if any) in target azimuth can be achieved via this process.

An idea of the difficulty of estimating this bias is shown by Fig. 6-14, where the scan-to-scan variations in the calculated bias values for one aircraft as produced by two-sensor-multilateration are presented. This example assumed an unbiased sensor system, Mode S data variances, and a transponder bias of 0.03 miles. It is clear that quite a data spread has occurred. Only averaging over several scans could produce a reasonable bias estimate.

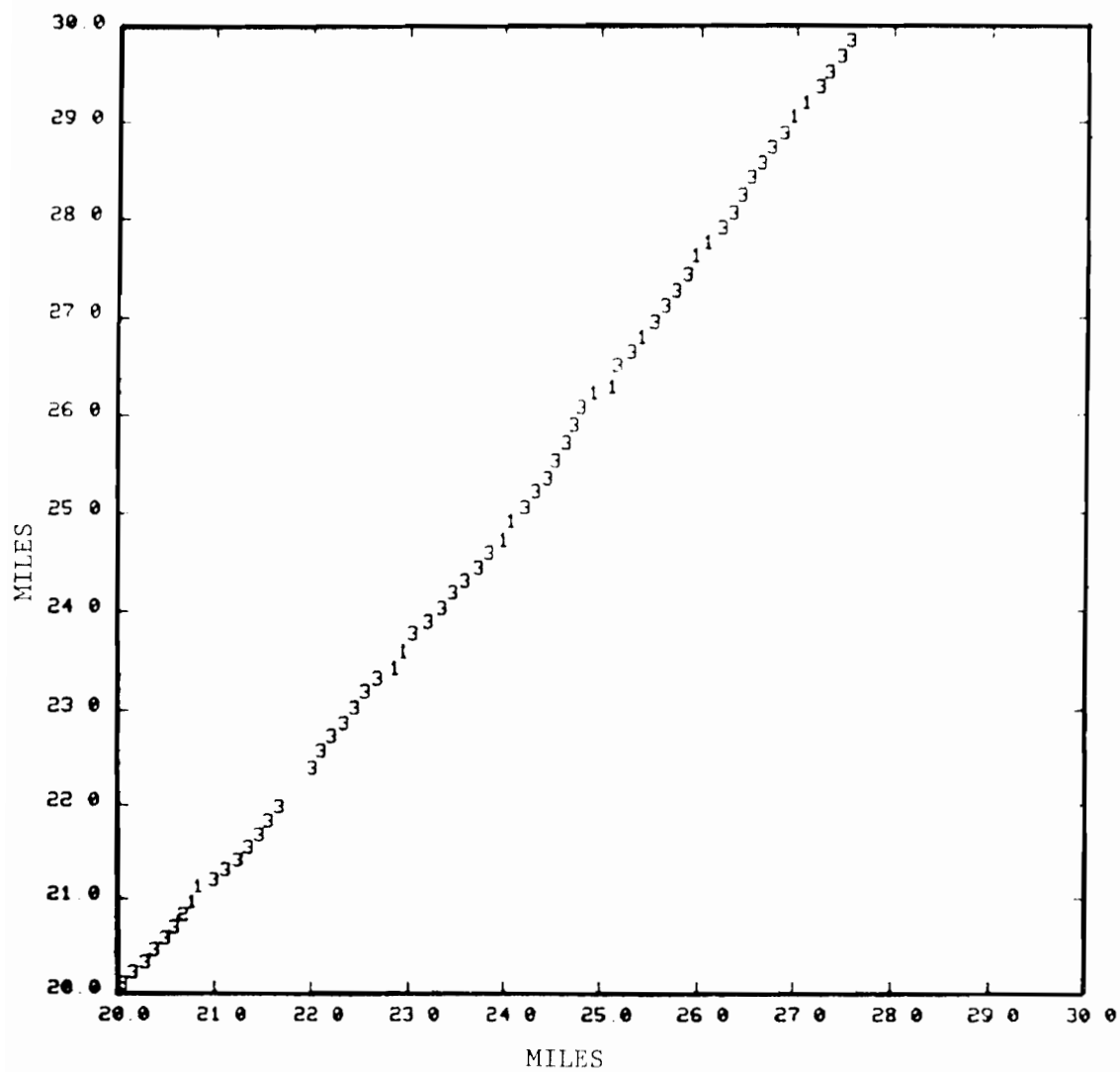


Fig. 6-13. Reprocessed Fig. 2-6 data.

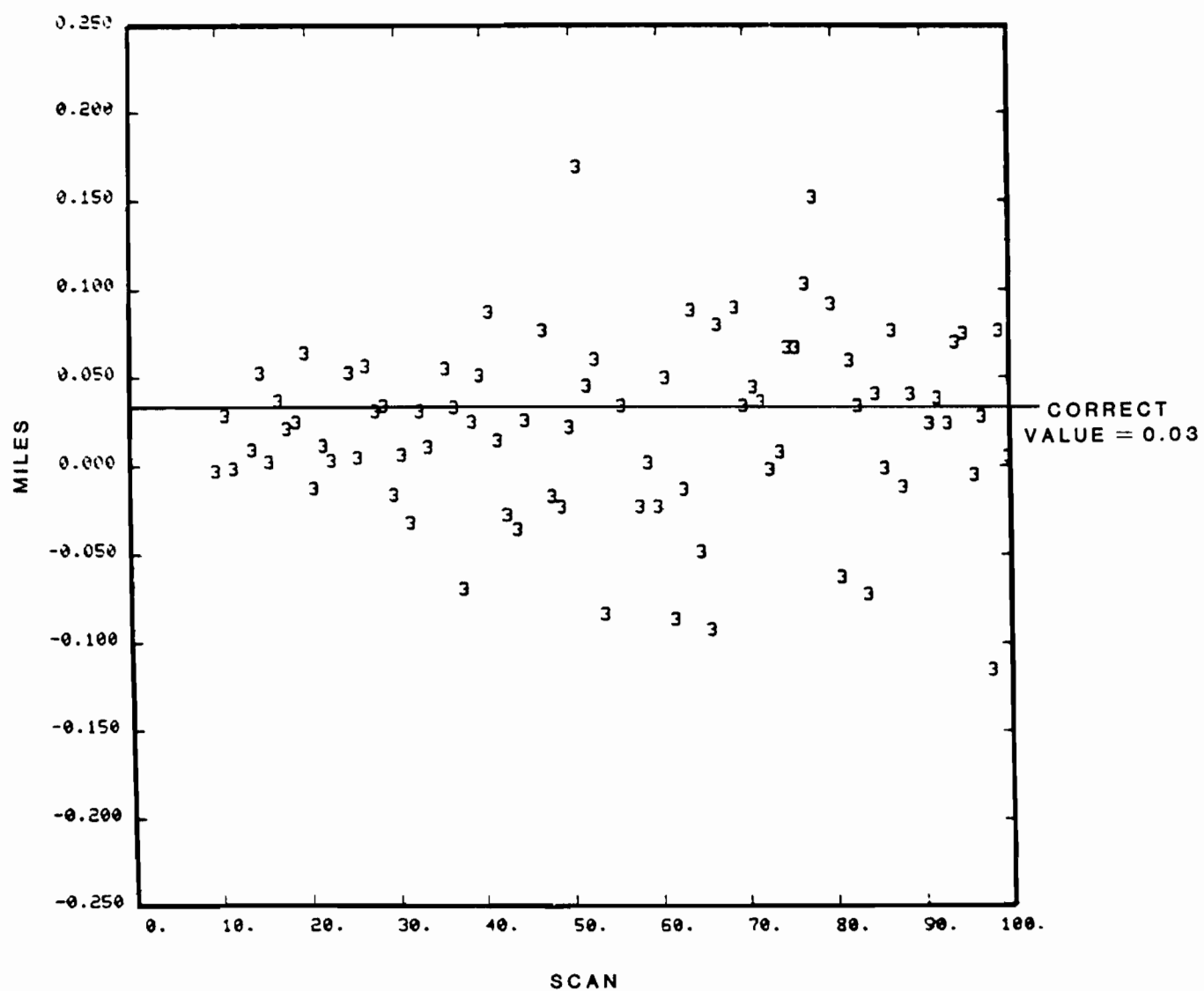


Fig. 6-14. Transponder bias estimates.

6.6 Altitude Estimation

Not all aircraft contain encoding altimeters. Thus there will exist aircraft under sensor coverage for which the altitude will be unknown. This lack of knowledge presents two problems. First, the true aircraft position cannot be determined, as the ground range required for x, y calculations is given by:

$$\rho_g = \sqrt{\rho^2 - z^2}$$

where z is computed from the altitude. Second, many safe aircraft crossing situations will be seen as apparent conflicts due to lack of knowledge of relative altitude separation.

It is not difficult to derive formulas that permit the calculation of aircraft altitudes from netted sensor data. What is difficult, though, is to produce formulas that are sufficiently insensitive to measurement noise and system biases to produce meaningful answers. In particular, any two sensor equations must use an azimuth value.

Fortunately, exact altitude values are not required for conflict alert applications. Knowledge that aircraft are separated by several thousands of feet is sufficient, even with an error margin of 50%. Thus, it is possible to employ altitude estimation formulas that have significant error potential, and to average the values so obtained in successive calculations.

An idea of the accuracy obtainable for altitude estimates with multiple sensor data can be provided very simply. Figure 6-15 presents the geometry that exists for two sensors when a plane is passed through the sensors, perpendicular to the earth's radius at the midpoint of the line connecting them. The "altitude" H of the aircraft relative to this plane will differ from the true altitude h, but its estimation accuracy will be virtually identical. Using the law of cosines, we can solve for H:

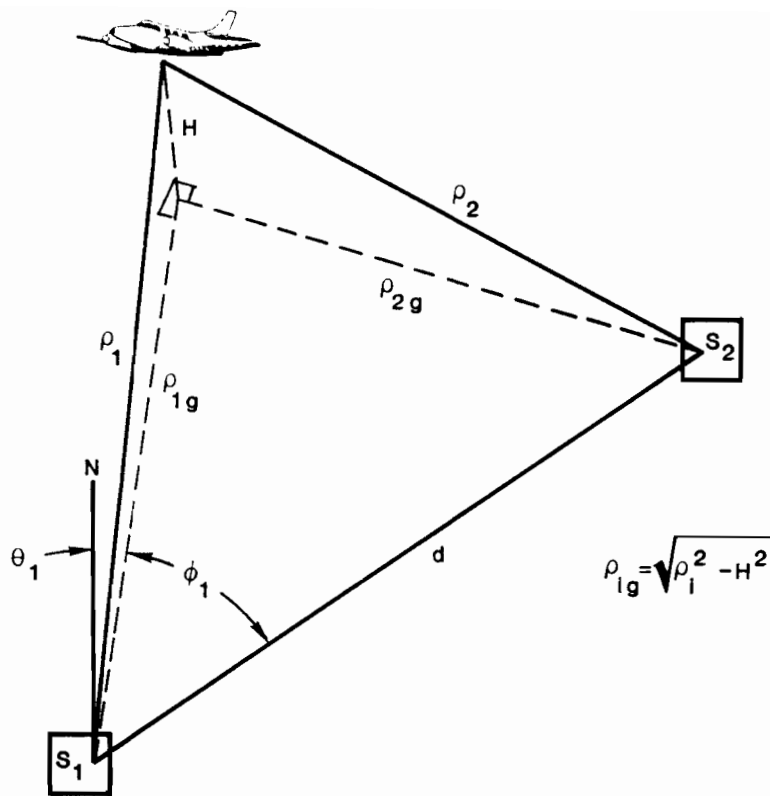
$$\begin{aligned} \rho_2^2 - H^2 &= d^2 + \rho_1^2 - H^2 - 2d \sqrt{\rho_1^2 - H^2} \cos \phi_1 \\ 2d \sqrt{\rho_1^2 - H^2} \cos \phi_1 &= d^2 + \rho_1^2 - \rho_2^2 \\ \rho_1^2 - H^2 &= \frac{(d^2 + \rho_1^2 - \rho_2^2)^2}{4 d^2 \cos^2 \phi_1} \end{aligned} \quad (6-4)$$

$$H = \sqrt{\rho_1^2 - \frac{(d^2 + \rho_1^2 - \rho_2^2)^2}{4 d^2 \cos^2 \phi_1}} \quad (6-5)$$

where, as shown in the figure,

d = distance between the sensors

ϕ_1 = azimuth of the aircraft, in the plane, relative to the inter-sensor line.



H IS THE "ALTITUDE" OF THE AIRCRAFT FROM
THE PLANE PASSING THROUGH THE SENSORS

Fig. 6-15. Altitude estimation geometry.

The major contribution to estimation noise is the jitter of the azimuth measurement that is required for two-sensor altitude determination. Taking the derivative of (6-5) with respect to this azimuth:

$$\frac{\partial H}{\partial \phi_1} = -\frac{1}{2z} \frac{(d^2 + \rho_1^2 - \rho_2^2)^2 \sin \phi_1}{4d^2 \cos^3 \phi_1}$$

Using (6-4):

$$\begin{aligned} \frac{\partial H_1}{\partial \phi_1} &= -\frac{1}{z} (\rho_1^2 - H^2) \tan \phi_1 \\ &= -\frac{\rho_1 g \tan \phi_1}{\tan \psi} \end{aligned} \quad (6-6)$$

where ψ is the elevation angle.

As expected from GDOP (geometric dilution of precision) considerations, the altitude estimate is much better for high-flying aircraft than for low-flying ones. A perhaps unexpected result, however, is that the estimate is better for aircraft flying between the sensors than for ones off to the side. Thus the desirable location for the secondary sensor for azimuth improvement, namely at a position providing a nearly perpendicular aspect angle at the aircraft, is a poor one for altitude determination. This suggests that data from two different secondary sensors may be needed for total position calculation of non-altitude-reporting aircraft. The communications and processing implications of such a strategy tend to make this approach unattractive however.

An error analysis for three sensor altitude estimation can also be made. The solution and error analysis for this case, being more complex, is given in Appendix C. With three sensors, only range measurements are required. As shown in the Appendix, the only major factor relating range jitter to altitude estimation accuracy is the elevation angle:

$$\frac{\partial z}{\partial \rho_1} = \frac{k}{\tan \psi} \quad (6-7)$$

To test the accuracy of altitude estimation procedures, the algorithms developed in Sections 8.4 and 8.5 were applied to the 24 aircraft trajectory data discussed in Chapter 2. Table 6-1 presents, by altitude band, the estimation errors for both two and three sensor cases. Mode S quality data, with no system biases, was assumed in the simulation.

As predicted, the estimation error for the three sensor procedure is nearly monotonically decreasing with altitude. With two sensors, however, this is clearly not the case. Instead, the angle of the aircraft relative to the inter-sensor line, as predicted, plays a prominent role. Thus high flying aircraft off to the side will produce larger errors than low flying ones between the sensors.

TABLE 6-1

ALTITUDE ESTIMATION ERRORS, UNBIASED SYSTEM

Aircraft Altitude	2 Sensors	3 Sensors
0-1 nm	.61 nm	.24 nm
1-2 nm	.46 nm	.16 nm
2-3 nm	.94 nm	.19 nm
3-4 nm	.39 nm	.11 nm
4-5 nm	.51 nm	.11 nm
5-6 nm	1.10 nm	.09 nm
6-7 nm	.26 nm	.05 nm

Analysis of individual data points that entered into the table averages showed that the estimation error for two sensors did indeed improve with altitude for aircraft at the same geometric location. Table 6-2 presents two slices of this finer data, one for constant angle from the inter-sensor line and varying altitude, the other for constant altitude and varying angle. In each case, the estimates react in the manner predicted by (6-6): performance improves with increasing altitude, and degrades as the angle from the inter-sensor line becomes larger (note that 90° is the widest angle, as ϕ_1 and $180^\circ - \phi_1$ are equivalent angles). Also note that in this example, the angle from the inter-sensor line has a much greater effect on accuracy than the aircraft altitude.

In all cases, however, especially when three sensors are employed, the magnitude of the error was sufficiently small to permit differentiation between aircraft in potential conflict and those well separated vertically. Thus netting is indeed valuable for altitude estimation. To insure that this conclusion would stand up when transponder and sensor biases were present, the study was repeated for such a situation. The results, presented in Table 6-3, have clearly been degraded. However, the errors are still sufficiently small to be usable for air traffic control applications. The three sensor results, in particular, are still excellent.

The next several chapters present various approaches to altitude estimation, for both two and three sensor cases. These chapters also investigate the azimuth errors that result when netting data from aircraft without altimeters.

6.7 Smoothing Filters

The method generally employed for determining accurate heading estimates is a smoothing filter. Sensor reports are fed into this filter in real-time, and random errors are averaged and hopefully removed. With single sensor surveillance, bias errors cannot be eliminated by this filter, but fortunately they rarely affect heading calculations.

Netting significantly alters the requirements on the tracking filter. In particular, as opposed to single sensor reports, netting reports:

1. are not homogeneous, so measurement variances differ from one to another
2. have differing biases
3. arrive at non-uniform update times.

Thus more complex filters are required to process this data stream. Failure to provide such a netting-optimized filter may well negate all the position improvement advantages obtained from netting.

Chapter 7 presents a discussion of the report generation timing issue. Then, after Chapters 8 and 9 present algorithms for position improvement, Chapter 10 develops a number of candidate smoothing filters. Simulation results are provided for evaluating the "optimum" tracker.

TABLE 6-2

ALTITUDE ESTIMATE BREAKDOWN

At angles from inter-sensor line between 0° and 20°:

	ALTITUDE						
	0-1 nm	1-2 nm	2-3 nm	3-4 nm	4-5 nm	5-6 nm	6-7 nm
ERROR	.45 nm	.24 nm	.15 nm	.19 nm	no data	.08 nm	.04 nm

At altitudes between 2 nm and 3 nm:

	INTER-SENSOR ANGLE						
	0°-20°	20°-40°	40°-60°	60°-80°	80°-100°	100°-120°	120°-140°
ERROR	.15 nm	.45 nm	.72 nm	1.47 nm	3.30 nm	2.51 nm	2.30 nm

TABLE 6-3

ALTITUDE ESTIMATION ERRORS, BIASED SYSTEM

Aircraft Altitude	2 Sensors	3 Sensors
0-1 nm	.77 nm	.56 nm
1-2 nm	.72 nm	.52 nm
2-3 nm	1.12 nm	.46 nm
3-4 nm	.69 nm	.26 nm
4-5 nm	.85 nm	.23 nm
5-6 nm	1.38 nm	.24 nm
6-7 nm	.43 nm	.20 nm

Assumed Transponder Bias: 400 nanoseconds
 Assumed Sensor Biases: 150 feet in location
 30 feet in range measurement

7.0 NETTING TIMING

With single sensor surveillance, one report per scan is provided to the user for each aircraft. These reports are equally spaced in time and are current (except for a small internal sensor processing delay). This uniformity of data permits very simple trackers to be employed, while the lack of delay provides displays with a real-time picture of the aircraft situation.

When netting is employed, the potential for more complex report timing is introduced. In particular, two major sets of options must be considered:

1. should one report per scan continue to be provided to the user, or should a report be sent corresponding to each observation by a sensor,
2. should data still be provided in real-time or should more accurate, though delayed, data be sent.

This chapter presents and compares the various alternatives that have been examined for these options.

7.1 Data Frequency

Each sensor provides one report per scan per aircraft under netting. In general, these reports will be randomly spaced in time as shown in Fig. 7-1. No matter what netting algorithm is being employed, it is possible to provide the user a report corresponding in time to each of these observations. The alternative is to only generate the one report per scan corresponding to the observation of the primary sensor.

It would appear that, apart from tracker complexity, sending all reports will provide better service to the user. However, there are many cases where this will not be true. First, some of the reports may have greater accuracy than the others due to the characteristics of the netting algorithm in use. Sending the poorer quality ones may then degrade, rather than enhance, the overall tracker performance.

Second, system biases will often affect the data at each sensor differently. Then, as shown in Fig. 7-2, each set of reports will be consistent, but inter-sensor registration errors will exist. In such a case, sending only one report per scan may result in superior tracker operations. These bias effects are studied further in later chapters.

Of course, user tracker complexity should not be ignored. A tracker with uniformly spaced inputs is much simpler to design and implement than one with variably spaced reports. Thus, unless multiple reports per scan can be shown to lead to significantly better performance than a single report, this issue alone would tilt the balance toward single report output operation.

7.2 Interpolation vs. Extrapolation

Most netting algorithms, including all multilateration formulas, require that the data from all sensors represent the identical point in time. Since



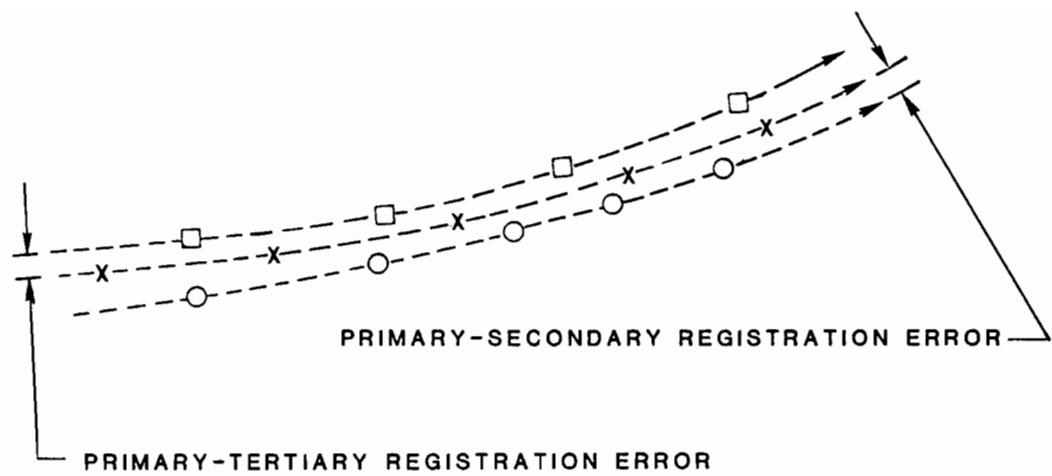
KEY:

X = PRIMARY SENSOR (ASSUMED 4.5 SECOND SCAN RATE)

○ = SECONDARY SENSOR (4.0 SECONDS)

□ = TERTIARY SENSOR (5.0 SECONDS)

Fig. 7-1. Inter-sensor report timing.



KEY: SEE FIG. 7-1.

Fig. 7-2. Inter-sensor registration.

the sensor observations are not naturally time coincident, this fact implies that the measurements from all but one sensor must be time adjusted to the time of the remaining sensor.

Two methods exist for data adjustment: interpolation and extrapolation. Interpolation is defined as estimating the data value between two existing measurements, while extrapolation is defined as projecting data beyond the last known value. In effect, interpolation is equivalent to hindsight, extrapolation to foresight. Thus, it is not surprising that interpolation is generally more accurate.

The worst case error potentials of these two processes are illustrated in Figs. 7-3 and 7-4, where D is the assumed worst case error of a single measurement. For straight flight, the former figure shows that the interpolation error can be no worse than that of a measurement. In fact, the estimation variance is smaller than that of the measurement, being reduced by a factor of 2 at the midpoint. For curved flight, the worst case interpolation error can slightly exceed that of the measurement as shown in the figure. However, the estimation variance is still smaller than that of the data.

Extrapolation, on the other hand, can introduce errors far in excess of the data errors. For straight flight, as shown in the latter figure, a one-scan projection can have an error a factor of 3 greater than that of the measurement, while the potential is even worse for curved flight. These statements apply for the usual two point linear extrapolation. More sophisticated algorithms can reduce the worst case errors. However, the computation and storage requirements of such methods would exceed the abilities of Mode S sensors as currently designed.

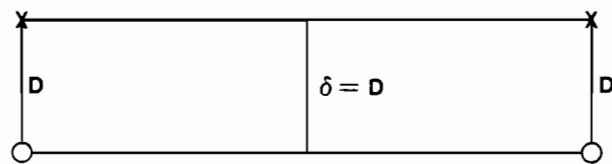
The increased accuracy of interpolation is not achieved without cost. In order to have data available beyond the time at which netting is performed, the common netting time must be behind real time. Thus the data produced is delayed. Since ATC users desire real-time information, this delayed value must be brought forward, by extrapolation.

This extrapolation, however, can use more sophisticated algorithms, as it is not located within the sensor surveillance system. Thus conflict alert, the user most interested in accurate data, can employ Kalman filters if desired for its data projections. The overall combination of interpolation for netting and extrapolation to real time may then be more accurate for netting than extrapolation alone. The last section of this chapter investigates this hypothesis further.

7.3 Timing Alternatives

Six different timing cases were considered in this study. These cases represent all combinations of one or multiple reports output per scan and the three reasonable alternatives for data estimation: extrapolation, interpolation, or a hybrid mixture of the two. Figures 7-5, 7-6, and 7-7 summarize the actions undertaken in each pair of cases.

STRAIGHT FLIGHT WORST ERROR (δ):



X = TRUTH

O = MEASURED

CURVED FLIGHT WORST ERROR:

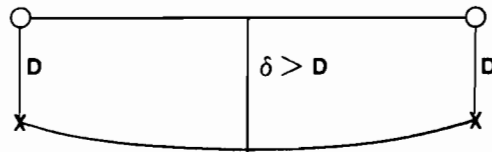
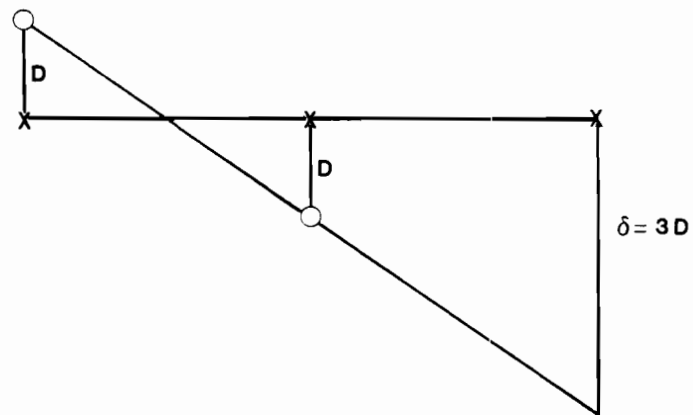


Fig. 7-3. Interpolation error potential.

STRAIGHT FLIGHT WORST ERROR (δ):



X=TRUTH

O=MEASURED

CURVED FLIGHT WORST ERROR:

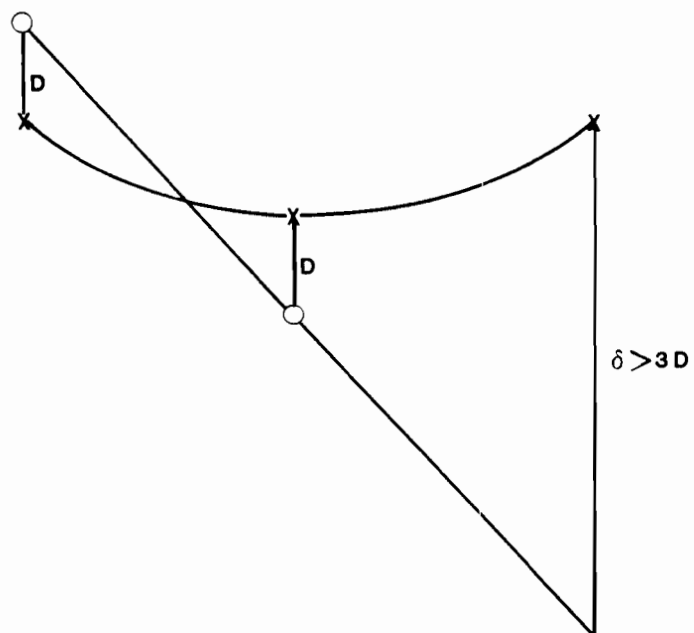
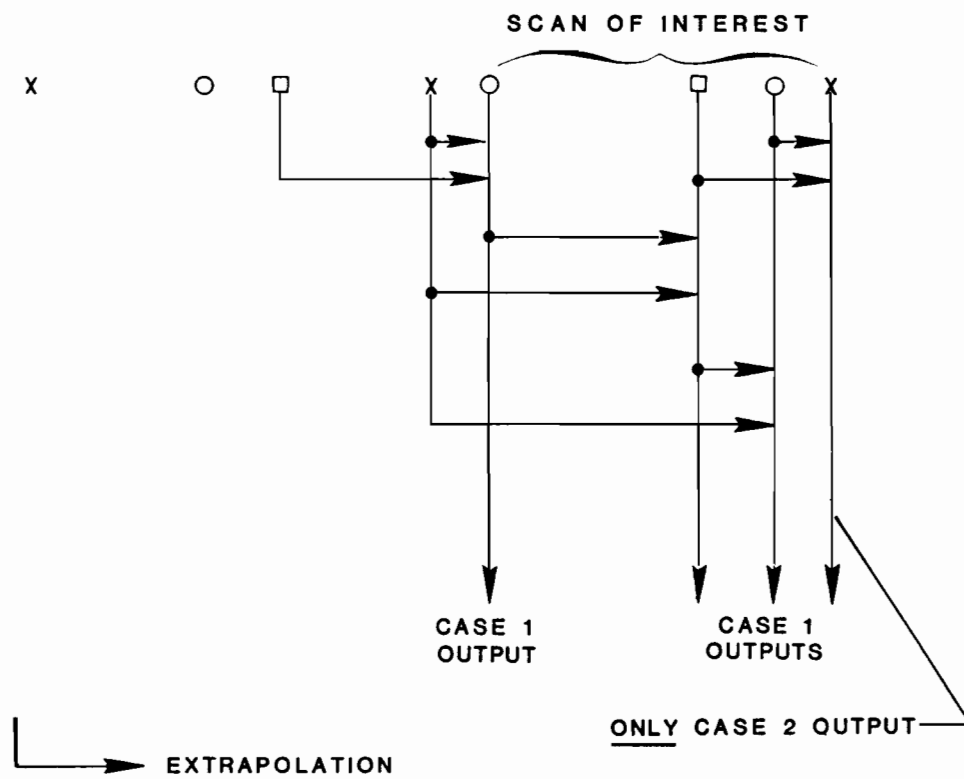


Fig. 7-4. Extrapolation error potential.



KEY: SEE FIG. 7-1.

Fig. 7-5. Extrapolation timing cases.

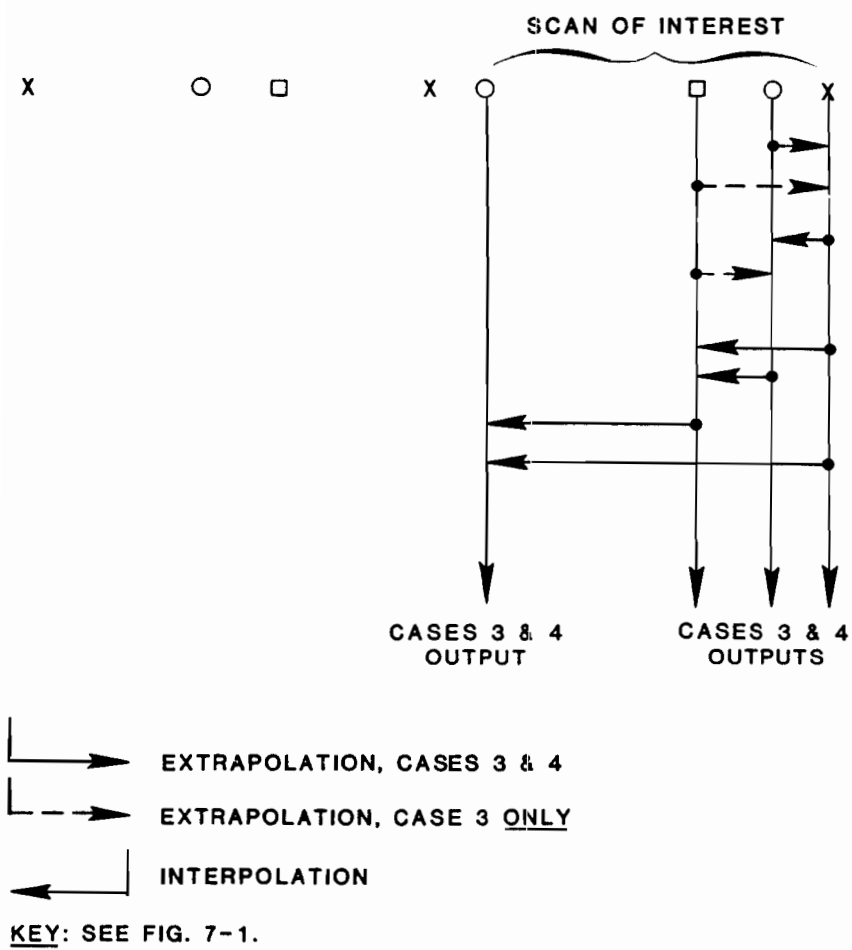


Fig. 7-6. Hybrid timing cases.

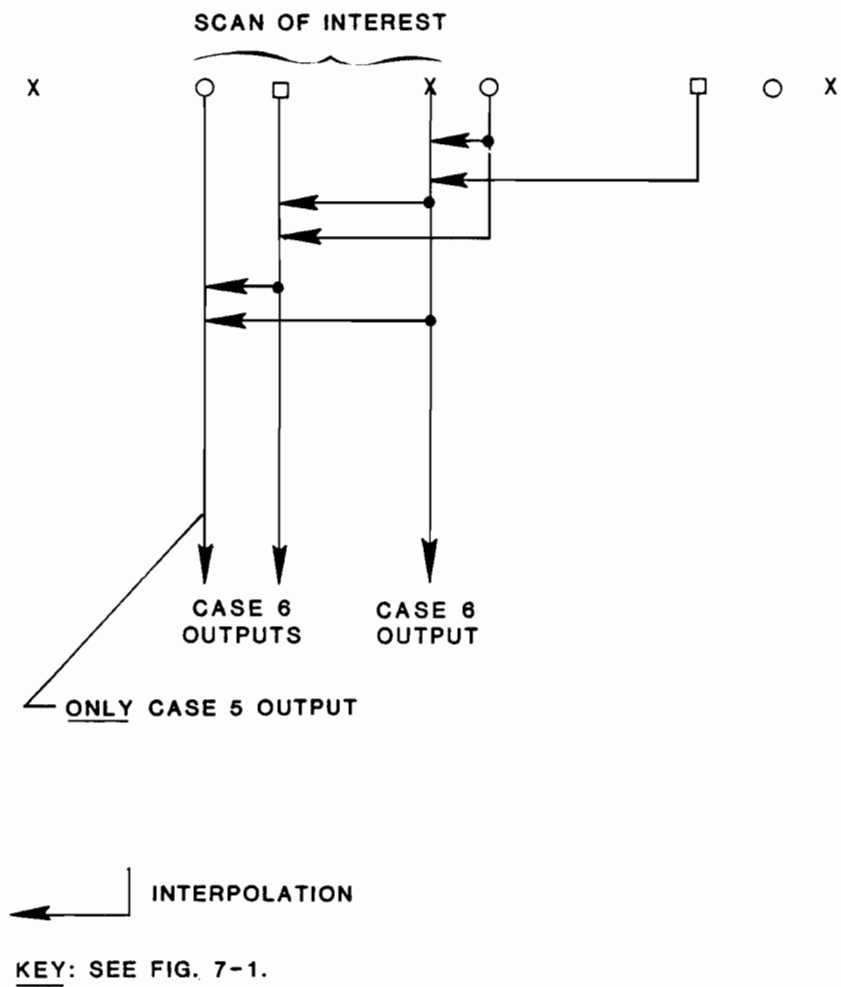


Fig. 7-7. Interpolation timing cases.

For the first two cases, all reports are output at the time of their relevance, that is, in real time. Thus only extrapolation is employed. For case 1, a report is generated each time a sensor views the target. The data for that sensor is used unmodified, while the data for each other sensor covering the target is extrapolated forward from the last sensor observation time to the current time. Then the multilateration (or other netting) algorithm is performed, and a report is output.

Case 2 is the single-report-per-scan analog of case 1. For this case, a report is generated only at the time the primary sensor views the target. Its data is used unmodified, other sensors' data are extrapolated forward. This action is taken every scan, even if the primary sensor missed the target, so that a uniformly spaced stream of reports is provided to the user. Note that with case 1, no report would be generated at the primary sensor time if that sensor failed to form a report.

The next two cases (3 and 4) combine interpolation with extrapolation to provide possibly superior data quality without sacrificing the real-time primary outputs presently expected by users. These cases, like case 1, provide one output report corresponding to each sensor observation, but now all reports are generated when the primary report occurs (or is missed). Thus the reports are output in a batch mode even though they correspond to different times.

The generation of each report is as follows. Assume the report corresponds to a sensor i observation at time t . Then sensor i 's data is used unchanged. For sensor j , $j \neq i$, its data is obtained by interpolation if a sensor j observation occurred later than time t ; by extrapolation if the last one was earlier than time t . Thus, in general, the earliest time report in a scan's batch will have only interpolated inputs, the last one (the primary sensor report) only extrapolated inputs, and the middle ones a combination.

The difference between cases 3 and 4 pertains to the number of sensors' data input to the netting algorithms. Case 3 assumes data from all sensors is always used, while case 4 ignores extrapolated sensor data whenever two or more sensors exist with more recent observations (as shown in the figure). Thus case 4 sacrifices quantity of inputs for quality of inputs. For two sensor systems, of course, these cases are equivalent.

In any of the above four cases, a parameter limits the length of extrapolation permitted for a sensor's data. The typical setting cuts off the input from any sensor that experiences two successive misses on the target. This rule limits the error potential of the data fed to the netting algorithm.

The final two cases, 5 and 6, employ only interpolation of data. By necessity, this causes the issuance of each output report, including that corresponding to the primary sensor observation, to be delayed from real time. Case 6 generates a report corresponding to each sensor's observation. This report, however, is only produced after each other sensor has subsequently viewed the aircraft (whether or not it actually produced a report), permitting interpolation to be employed. No data is input to netting from any sensor that missed the target.

Case 5 generates only one report per scan. This report is the one from the set defined for case 6 that is ready to be output when the primary sensor views the target. In general, as shown in Fig. 7-7, this report will correspond to the observation time of the sensor that saw the target first on the scan (except when two or more reports are produced by a sensor on a scan). Thus, although the output times of case 5 reports will be uniformly spaced, their times of relevance will not be, and the interval between reports can vary considerably from one scan to the next. Case 5 is the single-report-per-scan analog of both cases 4 and 6.

7.4 Timing Performance Comparisons

An exhaustive study of the relative performance of these six cases has not been made to date. However, Table 7-1 presents some data providing an indication of the results that might be obtained. The six timing cases were applied to the 24 aircraft trajectories recorded in the trilateration data base. The input reports were produced by the simulation procedure described in Chapter 2.

For each timing case, the ρ and θ values for each sensor were estimated whenever needed (once or multiple times per scan) by the method specified above for that case (interpolation or extrapolation). The values were then input to the 3-sensor multilateration algorithm presented in Section 8.3 to produce the output reports. The azimuth accuracy of these reports was determined and the results, averaged over all aircraft, are shown in the table.

Each set of output reports produced was then smoothed by the two opposite end of spectrum filters: Kalman and two-point. The resulting track predictions were then sampled two scans beyond the time of the primary sensor observation. The tracker position and heading estimates were compared to the real aircraft position and heading, and the errors noted. The table presents the average position and heading errors over all scans and trajectories to indicate the quality of data available to users.

The first set of results, as expected, shows that the interpolation timing cases provide more accurate reports than the extrapolation cases. The failure of case 4 is probably due to its use of tertiary sensor data rather than the preferred secondary sensor data on many scans. Thus, since it is also the most complex case, it is not considered further.

Since the interpolation cases produce reports behind real time, the trackers must project them a greater distance to reach a common time with the extrapolation cases. Thus it is expected, and verified by the results, that the advantage of the interpolation cases is diminished. In particular, the Kalman filter results for cases 6 and 5 are negligibly better than those for cases 1 and 2 respectively.

It is also seen in the table that one report per scan provides superior results to three reports per scan for both trackers, with a significant difference for the two-point interpolator. This is caused by the magnification of position differences, when expressed in velocity units, for

TABLE 7-1

TIMING TYPE PERFORMANCE COMPARISONS

		Timing Type					
Statistic		1	2	3	4	5	6
Measurement Azimuth σ		.08°	.09°	.08°	.10°	.07°	.07°
8-Second Prediction:	Timing Type Defi- nition	Extrapolate 3/scan	Extrapolate 1/scan	Mixed 3/scan	Mixed 3/scan	Interpolate 1/scan	Interpolate 3/scan
2-Pt Interpolator:							
Position Error		.57 nm	.09 nm	.49 nm	.67 nm	.10 nm	.29 nm
Heading Error		16.0°	6.5°	11.8°	13.3°	6.5°	9.6°
Kalman Filter:							
Position Error		.11 nm	.07 nm	.10 nm	.10 nm	.07 nm	.09 nm
Heading Error		6.1°	5.6°	5.8°	6.2°	5.5°	5.8°

closely spaced-in-time reports. It is quite probable that, for a 12-second sensor, multiple reports per scan would provide superior tracking performance, particularly in turns.

Since case 2, which is by far the simplest, produces as good or better results than any other case for 4-second sensors, this case is used for most testing of algorithms in the next three chapters. Except where specifically noted, it should be assumed for all tables presented there.

8.0 MULTILATERATION WITH TWO OR THREE SENSORS

A common method of utilizing data from two or more sensors is multilateration. Strictly speaking, multilateration is defined as determining an aircraft's position from the range measurements of two or more sensors whose joint overlapping coverage region includes the aircraft. This chapter, however, includes some algorithms that require azimuth data as well.

An aircraft's position, being in 3-dimensional space, requires at least three measurements for its determination. When altitude information is reported in the downlink message, only two surveillance measurements are needed. Thus two sensor multilateration is sufficient. In the absence of such knowledge, such as for non-altitude-reporting aircraft, three sensors are needed for normal multilateration. When only two sensors report on the aircraft, an azimuth measurement must be employed. This chapter develops the optimum method of utilizing this less accurate piece of data.

Whenever the aircraft transponder turnaround delay is not perfectly calibrated, all sensor range measurements contain a bias Δ . Thus, to employ multilateration, Δ must be computed and subtracted from each range. Since four quantities must now be determined (x, y, z, Δ), azimuth data must be utilized in many more cases: for two sensors at all times, and for three sensors whenever altitude information is absent. All these situations are examined here. The general multilateration diagram, with the notation used in this chapter, is presented in Fig. 8-1.

Multilateration can also be utilized when four or more sensors supply data on the aircraft. In particular, when both Δ and altitude are considered to be unknown, four ranges are required. In this case, the procedure becomes identical to hyperbolic multilateration, as unknown Δ is mathematically equivalent to absence of knowledge of the interrogation transmission time. In other cases, the problem becomes overspecified (more measurements than variables). Various least squares techniques exist for solving this problem. This report does not deal with any of these issues, as it is felt that data from four sensors would be quite rare. Whenever it does exist, using only the best three sensors, based on distance and geometry, will generally yield nearly optimum results.

It should be noted that multilateration works best when the sensors view the target from different directions, so that one sensor's range measurement supplies the other sensor's azimuth coordinate. Optimum netting occurs with a perpendicular aspect angle, while poor results appear when the target passes between the sensors. In the worst case of co-linear target and sensors, the azimuth error variance becomes infinite.

8.1 Earth Models

The earth's surface is of approximately spherical, nearly ellipsoidal shape. Many studies have been made to determine the "exact" shape of the earth. The various measurements of the international ellipsoid, and a discussion of local surface variations, are contained in [5].

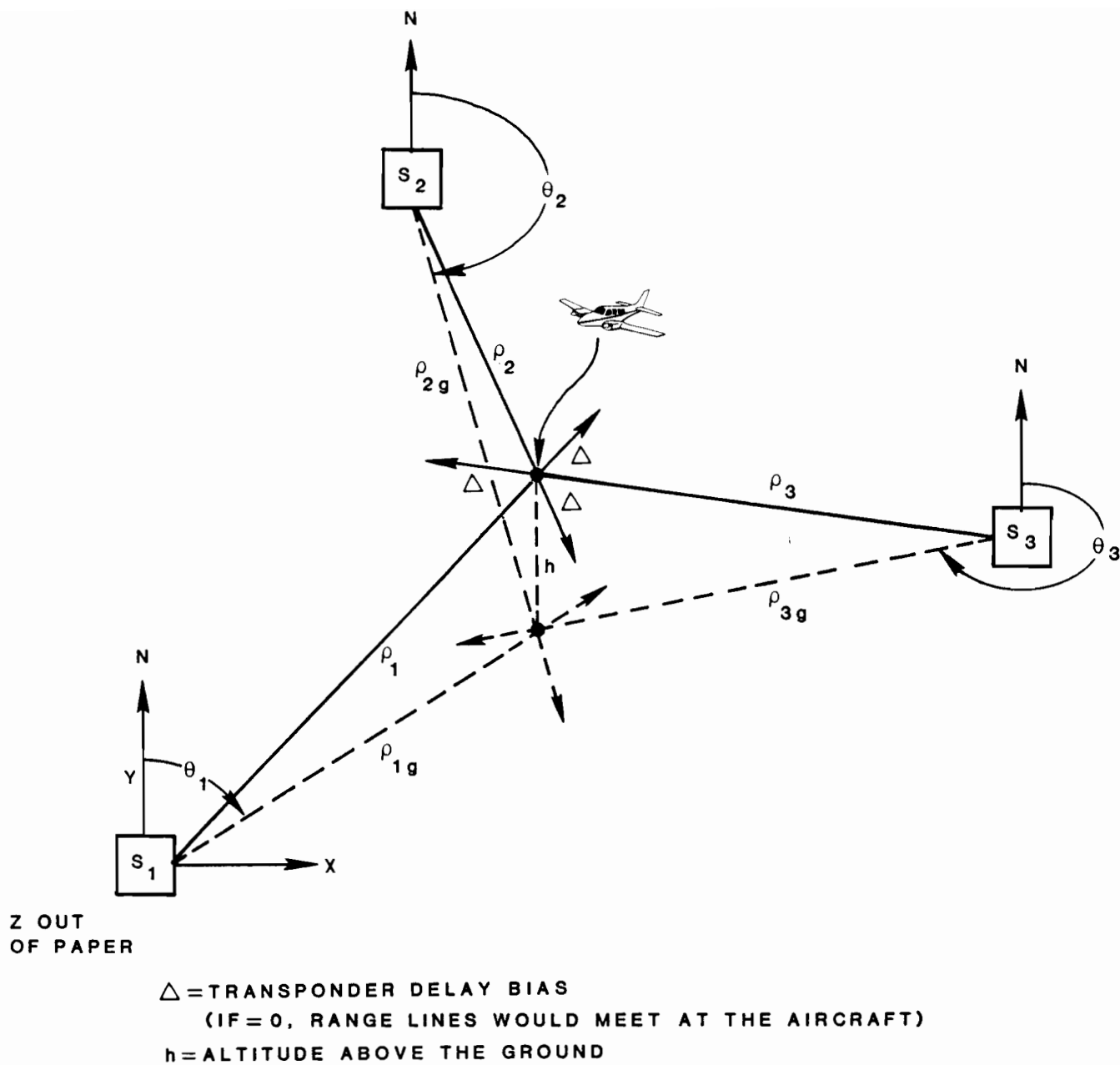


Fig. 8-1. Multilateration diagram.

A common problem in systems employing two or more sensors for surveillance is the transformation of measurements from one sensor's coordinate system to that of another. This is especially true in localized systems, in which surveillance is performed at the sensor. Then secondary sensor data is used by the primary sensor as a performance aid. If this data is to fill in gaps, or confirm extraneous reports, it must be converted:

$$\begin{array}{ccccc} & (a) & & (b) & & (c) \\ (\rho_2, \theta_2, h) & \rightarrow & (x_2, y_2, z_2) & \rightarrow & (x_1, y_1, z_1) & \rightarrow & (\rho_1, \theta_1, h) \end{array}$$

For calculation (b), very precise models of the earth may be employed, as the position of the aircraft is known prior to the transformation. The degree of exactness is determined by the quality of the sensor data, as this step attempts only to maintain the data accuracy already in existence.

The bilateration problem, when data from primary and secondary sensors is employed jointly, requires a considerably different form of coordinate transformation:

$$(\rho_1, \rho_2, h) \rightarrow (\rho_1, \theta_1, h)$$

In this application, the position of the aircraft on the earth becomes accurately known only after the transformation is performed. Thus the use of an exact earth model would require an iterative solution. To avoid the great complexity that would result from such an approach, the assumption of a locally-spherical earth has generally been employed for bilateration. By "locally-spherical" we mean assuming the coverage region is part of a sphere, whose radius is that of the true ellipsoidal earth at the center of the region.

Additional mathematical simplicity could be attained if the earth model could be further simplified to a locally flat surface. Unfortunately, the bilateration errors that would result from this approach would be far greater than those from a single sensor. Thus, instead of improving surveillance, a degradation would occur. For this reason, the spherical earth approach has been the traditional one for bilateration.

There are, though, accurate methods of mapping the earth onto a flat plane. For example, stereographic projection is commonly employed to transform measurements from several sensors onto a common plane [6]. This approach is useful for centralized surveillance systems. This type of projection method is unsuitable for the bilateration problem, however. First, it would be iterative, as again the position of the target is required for the transformation. Second, and more serious, the transformation equations are more complex than those for spherical bilateration, so the resulting flat earth model would add to and not reduce the mathematical complexity.

Chapter 9 of this paper introduces a new approach to using a flat earth model for bilateration. This model, although not a true representation of the earth's surface, produces surveillance results that are as accurate as the spherical model.

8.2 Two Sensors, Altitude Known, $\Delta = 0$

The simplest case of multilateration exists when an aircraft reports its altitude, and its transponder is assumed to be calibrated. The equations to be used in this case are:

$$x^2 + y^2 + z^2 = \rho_1^2 \quad (8-1)$$

$$(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2 = \rho_2^2 \quad (8-2)$$

where

ρ_i is the range measurement of sensor i , $i = 1, 2$

x_2, y_2, z_2 is the position of sensor 2 in sensor 1 coordinates

x, y, z is the aircraft position in sensor 1 coordinates

Then the azimuth of the aircraft is given by:

$$\theta = \tan^{-1} \left(\frac{x}{y} \right) \quad (8-3)$$

Its range, of course, is simply ρ_1 .

The position of sensor 2 (x_2, y_2, z_2) must be precomputed and stored in sensor 1. The relevant coordinate transformation equations are given simply by the U matrix in Appendix A, as a sensor is at $x=y=z=0$ in its own coordinate system. Thus:

$$\begin{aligned} x_2 &= (E+h_{s2})\cos\lambda_2\sin(\gamma_2-\gamma_1) \\ y_2 &= -(E+h_{s2})[\cos\lambda_2\sin\lambda_1(\cos\gamma_2-\gamma_1)-\sin\lambda_2\cos\lambda_1] \\ z_2 &= (E+h_{s2})[\cos\lambda_2\cos\lambda_1\cos(\gamma_2-\gamma_1)+\sin\lambda_2\sin\lambda_1] - (E+h_{s1}) \end{aligned} \quad (8-4)$$

where

E is the radius of the earth

h_{si} is the height of sensor i above sea level

λ_i is the latitude of sensor i

γ_i is the longitude of sensor i

Also, before (8-1) and (8-2) can be solved, the coordinate z must be computed from the reported aircraft altitude h :

$$z = \frac{(h-h_{s1})^2 + 2(h-h_{s1})(E+h_{s1}) - \rho_1^2}{2(E+h_{s1})} \quad (8-5)$$

The solution of the simultaneous equations (8-1) and (8-2) is straightforward. Subtracting the first from the second yields the linear equation:

$$-2xx_2 - 2yy_2 - 2zz_2 = \rho_2^2 - \rho_1^2 - d_2^2 \quad (8-6)$$

where

d_2 = distance of sensor 2 from sensor 1

$$= x_2^2 + y_2^2 + z_2^2$$

Solving for x in terms of y:

$$x = \frac{d_2^2 - \rho_2^2 + \rho_1^2 - 2zz_2}{2x_2} - \frac{y_2}{x_2} y \quad (8-7)$$

Finally, substituting (8-7) into (8-1):

$$\begin{aligned} y^2 \left[1 + \frac{y_2^2}{x_2^2} \right] + y \left[\frac{y_2}{x_2^2} (2zz_2 - d_2^2 + \rho_2^2 - \rho_1^2) \right] \\ + \left[\left(\frac{d_2^2 - \rho_2^2 + \rho_1^2 - 2zz_2}{2x_2} \right)^2 + z^2 - \rho_1^2 \right] = 0 \end{aligned} \quad (8-8)$$

The quadratic (8-8) can be solved for two values of y by the quadratic formula, the results used in (8-7) to produce the corresponding values of x, and the two possible azimuths found as in (8-3). The one closer to the measured θ_1 is the correct answer.

8.3 Three Sensors, Altitude Known, $\Delta=0$

When data from three sensors is available, a choice exists as to the secondary sensor to employ (as least squares techniques are not being considered). Of course, as discussed earlier, any sensor providing a bad aspect angle relative to the primary sensor must be avoided. If we assume that every area of coverage has a preferred secondary sensor, then that one will be chosen unless other considerations override this selection. One such consideration is which sensor most recently saw the target, so as to minimize extrapolation errors. This issue was discussed under netting timing.

Of course, if the only two sensors to view the target on a scan were the two non-primary sensors (the primary one having a miss), the equation (8-1) no longer applies. Instead, it must be replaced by

$$(x-x_3)^2 + (y-y_3)^2 + (z-z_3)^2 = \rho_3^2 \quad (8-9)$$

where the same definitions and precomputation notes apply as before. The equation (8-5) for z still applies, although since sensor 1 has no data the value of ρ_1 must be found via extrapolation.

The solution to (8-2) and (8-9) proceeds the same as before. The revised relationships become:

$$x = \frac{d_2^2 - d_3^2 - \rho_2^2 + \rho_3^2 - 2z(z_2 - z_3)}{2(x_2 - x_3)} - \frac{(y_2 - y_3)}{(x_2 - x_3)} y \quad (8-10)$$

$$\equiv A - By$$

for x in terms of y, and using (8-2):

$$y^2 [1+B^2] + y [2Bx_2 - 2AB - 2y_2] + [A^2 - 2Ax_2 + d_2^2 + z^2 - 2zz_2 - \rho_2^2] = 0 \quad (8-11)$$

for the quadratic to solve. The final answers for the target position are then:

$$\rho_1 = \sqrt{x^2 + y^2 + z^2}$$

$$\theta_1 = \tan^{-1}\left(\frac{x}{y}\right) \quad (8-12)$$

A caution should be noted when using three sensors. If any biases exist in the system, this approach will degrade more than the previous two sensor method, particularly when the two secondary sensors are the source of the data. As shown in (8-12), both the range and azimuth are now determined by netting, whereas before ρ_1 was always a measured value. Thus jumps in both coordinates can occur from scan to scan.

8.4 Two Sensors, Altitude Unknown, $\Delta=0$

When the aircraft altitude is unknown, the multilateration must determine all three position coordinates. Since the sensors essentially lie in a plane normal to the z coordinate, GDOP (geometrical dilution of precision) arguments imply that accurate determination of altitude is improbable. Any measurement noise is amplified substantially into altitude estimation errors.

For this reason, aircraft altitude will be determined by smoothing estimates over several successive scans. Since altitude can change over time, a moving average has been chosen:

$$h_n = \frac{w \cdot \hat{h}_{n-1} + h_n}{w+1} \quad (8-13)$$

where $\hat{h}_n$ and h_n are the smoothed and raw altitude estimates respectively on scan n , and w is a weighting factor (typically $w=4$).

To determine three position coordinates, three measurement variables are required: two ranges and an azimuth. To minimize extrapolation errors, the azimuth employed will always be that of the sensor whose measurement corresponds to the netting time. To simplify the mathematics, its coordinate system will be used for the equations. Thus:

$$x^2 + y^2 + z^2 = \rho_a^2 \quad (8-14)$$

$$(x-x_b)^2 + (y-y_b)^2 + (z-z_b)^2 = \rho_b^2 \quad (8-15)$$

$$x = y \tan \theta_a \quad (8-16)$$

where

ρ_a, θ_a are the measurements of the time coincident sensor
 ρ_b is the range of the second sensor
 x_b, y_b, z_b are the location of sensor b relative to sensor a

Subtracting (8-14) from (8-15), and using (8-16) for x , provides a relationship for y in terms of z :

$$y = \frac{-z_b z}{y_b + x_b \tan \theta_a} + \frac{d_b^2 - \rho_b^2 + \rho_a^2}{2y_b + 2x_b \tan \theta_a} \quad (8-17)$$

Using this result, plus (8-16), and substituting into (8-14), provides the following quadratic for z :

$$z^2 \left[1 + \frac{z_b^2 (1 + \tan^2 \theta_a)}{(y_b + x_b \tan \theta_a)^2} \right] + z \left[\frac{-z_b (d_b^2 - \rho_b^2 + \rho_a^2) (1 + \tan^2 \theta_a)}{(y_b + x_b \tan \theta_a)^2} \right] + \left[\frac{(d_b^2 - \rho_b^2 + \rho_a^2)^2 (1 + \tan^2 \theta_a)}{4(y_b + x_b \tan \theta_a)^2} - \rho_a^2 \right] = 0 \quad (8-18)$$

Of the two values of z determined by solving this quadratic, the more positive is the one corresponding to the aircraft altitude. Finally,

$$\hat{h} = E \left[-1 + \sqrt{1 + \frac{h_{sa}^2 + \rho_a^2 - 2zh_{sa}}{E^2} + \frac{2h_{sa} + 2z}{E}} \right] \quad (8-19)$$

The equations (8-14) through (8-16) could be solved further to generate x and y . However, these values would have poor accuracy, as well as possibly being in the wrong coordinate system. A more accurate method of position determination is as follows:

1. solve for $\hat{h}$ as above in (8-19)
2. smooth as in (8-13) to obtain h
3. employ the 2 sensor, altitude known, approach of (8-1) through (8-8) to obtain the desired θ_1

8.5 Three Sensors, Altitude Unknown, $\Delta=0$

When data from three sensors is available, only range measurements need be used to estimate the aircraft altitude. The equations to be solved in this case are:

$$x^2 + y^2 + z^2 = \rho_1^2 \quad (8-20)$$

$$(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2 = \rho_2^2 \quad (8-21)$$

$$(x-x_3)^2 + (y-y_3)^2 + (z-z_3)^2 = \rho_3^2 \quad (8-22)$$

Subtracting (8-20) from (8-21) and solving for x yields:

$$x = -\frac{z_2}{x_2} z - \frac{y_2}{x_2} y + \frac{d_2^2 - \rho_2^2 + \rho_1^2}{2x_2} \quad (8-23)$$

Next subtracting (8-20) from (8-22), using (8-23), and solving for y :

$$y = \frac{\frac{z_2 x_3}{x_2} - z_3}{y_3 - \frac{y_3 x_3}{x_2}} z + \frac{d_3^2 - \frac{x_3^2}{x_2^2} (d_2^2 - \rho_2^2 + \rho_1^2) - \rho_3^2 + \rho_1^2}{2(y_3 - \frac{y_3 x_3}{x_2})} \quad (8-24)$$

$$\equiv \frac{Y_1}{Y_3} z + \frac{Y_2}{2Y_3}$$

Using this to reduce (8-23):

$$x = - \left[\frac{z_2}{x_2} + \frac{y_2}{x_2} \frac{Y_1}{Y_3} \right] z + \left[\frac{d_2^2 - \rho_2^2 + \rho_1^2}{2x_2} - \frac{y_2}{2x_2} \frac{Y_2}{Y_3} \right] \quad (8-25)$$

$$\equiv -X_1 z + X_2$$

Finally, substituting (8-24) and (8-25) into (8-20) yields the quadratic for z :

$$z^2 \left[1 + \frac{Y_1^2}{Y_3^2} + X_1^2 \right] + z \left[\frac{Y_1 Y_2}{Y_3^2} - 2 X_1 X_2 \right] + \left[\frac{Y_2^2}{4Y_3^2} + X_2^2 - \rho_1^2 \right] = 0 \quad (8-26)$$

The more positive solution of this quadratic is the correct value for z .

Then, as above, $\hat{h}$ is found from z via (8-19).

Once the smoothed value of h is computed, this case reduces to the 3 sensor altitude known case presented in 8.3. The aircraft position is determined as described there.

8.6 Two Sensors, Altitude Known, Δ Unknown

Few if any aircraft have their transponder turnaround delay perfectly calibrated. Whenever this delay error is non-negligible, even though within specifications, ignoring it will cause the multilateration to introduce errors into the azimuth estimation. This section and the next three present methods for including Δ as another variable to be determined from the multiple sensor data.

When altitude is known, three position coordinates must be determined: x , y , and Δ . Thus, three measurement variables are required, two ranges and an azimuth. As above in section 8.4, extrapolation errors can be minimized by employing the azimuth and coordinate system of the sensor whose measurement corresponds to the netting time. Using the notation of that section, the three equations defining the current situation are:

$$x^2 + y^2 + z^2 = (\rho_a - \Delta)^2 \quad (8-27)$$

$$(x - x_b)^2 + (y - y_b)^2 + (z - z_b)^2 = (\rho_b - \Delta)^2 \quad (8-28)$$

$$x = y \tan \theta_a \quad (8-29)$$

Subtracting (8-27) from (8-28), and using (8-29) for x, permits the determination of y in terms of Δ :

$$y = \Delta \left[\frac{\rho_b - \rho_a}{y_b + x_b \tan \theta_a} \right] + \left[\frac{d_b^2 - 2zz_b - \rho_b^2 + \rho_a^2}{2y_b + 2x_b \tan \theta_a} \right] \quad (8-30)$$

Using this result in (8-29) to obtain x, and then substituting for x and y in (8-27), the following quadratic equation for Δ is obtained:

$$\begin{aligned} & \Delta^2 \left[\left(\frac{\rho_b - \rho_a}{y_b + x_b \tan \theta_a} \right)^2 (\tan^2 \theta_a + 1) - 1 \right] \\ & + \Delta \left[\frac{(\rho_b - \rho_a) (d_b^2 - 2zz_b - \rho_b^2 + \rho_a^2)}{(y_b + x_b \tan \theta_a)^2} (\tan^2 \theta_a + 1) + 2\rho_a \right] \\ & + \left[\frac{(d_b^2 - 2zz_b - \rho_b^2 + \rho_a^2) (\tan^2 \theta_a + 1)}{4 (y_b + x_b \tan \theta_a)^2} + z^2 - \rho_a^2 \right] = 0 \end{aligned} \quad (8-31)$$

Two values of Δ are the solutions to this quadratic; the one with the smaller magnitude is the valid answer.

The per scan values of Δ obtained in this manner will have wide variations due to the sensitivity of the calculation to measurement errors. Figure (6-14) illustrated this assertion by plotting Δ values determined on several successive scans for a single aircraft. Thus, averaging of values from many scans is required to obtain a reasonable estimate of the true aircraft transponder bias. The recommended procedure is as follows:

1. solve for Δ_n as above
2. limit Δ_n to prevent unreasonable values from affecting the averaging:

$$\Delta_n = \begin{cases} .07 & \Delta_n > .07 \\ -.07 & \Delta_n < -.07 \end{cases}$$

where .04 miles is the specification limit.

3. compute a new running average:

$$\hat{\Delta}_n = \frac{m * \hat{\Delta}_{n-1} + \Delta_n}{m+1}$$

where m is the number of previous samples of Δ .

As with the unknown altitude cases, the solution of equations (8-27) through (8-29) should not be completed to determine x and y. Instead, the value of $\hat{\Delta}_n$ should be used to adjust the range measurements:

$$\hat{\rho}_1 = \rho_1 - \hat{\Delta}_n$$

$$\hat{\rho}_2 = \rho_2 - \hat{\Delta}_n$$

and these adjusted values used in the equations (8-1) and (8-2) to produce the aircraft position.

8.7 Three Sensors, Altitude Known, Δ Unknown

The presence of a third sensor removes the need to use an azimuth measurement in the equations. Instead, the three ranges provide a sufficient set of variables:

$$x^2 + y^2 + z^2 = (\rho_1 - \Delta)^2 \quad (8-32)$$

$$(x-x_2)^2 + (y-y_2)^2 + (z-z_2)^2 = (\rho_2 - \Delta)^2 \quad (8-33)$$

$$(x-x_3)^2 + (y-y_3)^2 + (z-z_3)^2 = (\rho_3 - \Delta)^2 \quad (8-34)$$

These equations are solved in a manner similar to that used above. First, subtract (8-32) from (8-33) and solve for x:

$$x = \Delta \left[\frac{\rho_2 - \rho_1}{x_2} \right] - y \left[\frac{y_2}{x_2} \right] + \left[\frac{z_2^2 - 2zz_2 - \rho_2^2 + \rho_1^2}{2x_2} \right] \quad (8-35)$$

Then subtract (8-32) from (8-34), and use (8-35), to produce:

$$y = \Delta \left[\frac{\rho_1 - \rho_3 + \frac{x_3}{x_2} (\rho_2 - \rho_1)}{\frac{x_3}{x_2} y_2 - y_3} \right] + \left[\frac{2zz_3 + \frac{x_3}{x_2} (d_2^2 - \rho_2^2 + \rho_1^2 - 2zz_2) - (d_3^2 - \rho_3^2 + \rho_1^2)}{2 \frac{x_3}{x_2} y_2 - 2y_3} \right] \quad (8-36)$$

$$\equiv Y_1 \Delta + Y_2$$

Substituting this result back into (8-35):

$$x = \Delta \left[\frac{\rho_2 - \rho_1}{x_2} - \frac{y_2}{x_2} Y_1 \right] + \left[\frac{d_2^2 - \rho_2^2 + \rho_1^2 - 2zz_2}{2x_2} - \frac{y_2}{x_2} Y_2 \right] \quad (8-37)$$

$$\equiv X_1 \Delta + X_2$$

Finally, using the results (8-36) and (8-37) in (8-32) produces the desired quadratic:

$$\Delta^2 [X_1^2 + Y_1^2 - 1] + \Delta [2X_1X_2 + 2Y_1Y_2 + 2\rho_1] + [X_2^2 + Y_2^2 + z^2 - \rho_1^2] = 0 \quad (8-38)$$

The smaller magnitude solution is the valid transponder bias.

As in the previous section, the values of Δ must be averaged over many scans to produce a reasonably accurate result. Then the solution of x and y is obtained via the method described in section 8.3, using

$$\hat{\rho}_i = \rho_i - \hat{\Delta}_n \quad i = 1, 2, 3$$

8.8 Two Sensors, Altitude and Δ Both Unknown

The last possible situation occurs when the aircraft altitude is unknown and the transponder turnaround bias is thought to be non-negligible. In this case, all of the values x , y , z , and Δ must be determined from the data. Although theoretically possible to accomplish, the resulting values of z and Δ will almost always be inaccurate as well as showing wide variations from scan to scan. In particular, altitude and transponder delay tend to be highly correlated since both serve as range adjustments:

$$x^2 + y^2 = \rho^2 - (z^2 + 2\rho\Delta)$$

Thus a calculation error in one produces a corresponding error in the other.

This section and the next present the methods for solving for all four position variables from the data. However, it is felt unlikely that these approaches would ever be used in practice. Only the extended flat earth algorithm to be described in the next chapter has been shown capable of producing accurate azimuth estimates in this situation.

With two sensors, both measurement ranges and both measurement azimuths must be employed in the equations. Since the sensor 2 azimuth is defined in the "wrong" coordinate system, a transformation must be utilized. Converting x and y from sensor 1 to sensor 2 coordinates is accomplished as follows:

$$\begin{aligned} x^{(2)} &= [\cos(\gamma_1 - \gamma_2)] x - [\sin\lambda_1 \sin(\gamma_1 - \gamma_2)] y \\ &\quad + [\cos\lambda_1 \sin(\gamma_1 - \gamma_2)] z + [(E + h_{s1}) \cos\lambda_1 \sin(\gamma_1 - \gamma_2)] \quad (8-39) \\ &\equiv T_{x1} x + T_{x2} y + T_{x3} z + U_x \end{aligned}$$

$$\begin{aligned} y^{(2)} &= [\sin\lambda_2 \sin(\gamma_1 - \gamma_2)] x + [\sin\lambda_1 \sin\lambda_2 \cos(\gamma_1 - \gamma_2) + \cos\lambda_1 \cos\lambda_2] y \\ &\quad - [(\cos\lambda_1 \sin\lambda_2 \cos(\gamma_1 - \gamma_2) + \sin\lambda_1 \cos\lambda_2)] z \\ &\quad - [(E + h_{s1}) \{\cos\lambda_1 \sin\lambda_2 \cos(\gamma_1 - \gamma_2) - \sin\lambda_1 \cos\lambda_2\}] \quad (8-40) \\ &\equiv T_{y1} x + T_{y2} y + T_{y3} z + U_y \end{aligned}$$

Then the equations to be solved become:

$$x^2 + y^2 + z^2 = (\rho_1 - \Delta)^2 \quad (8-41)$$

$$(x - x_2)^2 + (y - y_2)^2 + (z - z_2)^2 = (\rho_2 - \Delta)^2 \quad (8-42)$$

$$x = y \tan\theta_1 \quad (8-43)$$

$$x^{(2)} = y^{(2)} \tan\theta_2 \quad (8-44)$$

The first step of the solution is to solve (8-44) for z in terms of y, using (8-39) and (8-40) to define $x^{(2)}$ and $y^{(2)}$ and (8-43) to eliminate x:

$$\begin{aligned} z &= y \left[\frac{(T_{y1} \tan\theta_2 - T_{x1}) \tan\theta_1 + (T_{y2} \tan\theta_2 - T_{x2})}{T_{x3} - T_{y3} \tan\theta_2} \right] + \frac{U_y \tan\theta_2 - U_x}{T_{x3} - T_{y3} \tan\theta_2} \quad (8-45) \\ &\equiv \frac{Z_1}{Z_3} y + \frac{Z_2}{Z_3} \end{aligned}$$

Then subtract (8-41) from (8-42) and solve for y, using (8-45) and (8-43) for z and x respectively, to obtain:

$$y = \Delta \left[\frac{\rho_2 - \rho_1}{\tan\theta_1 x_2 + y_2 + z_2 \frac{Z_1}{Z_3}} \right] + \frac{d_2^2 - 2z_2 \frac{Z_2}{Z_3} - \rho_2^2 + \rho_1^2}{2 \tan\theta_1 x_2 + 2y_2 + 2z_2 \frac{Z_1}{Z_3}} \quad (8-46)$$

$$\equiv \frac{D_1}{D_3} \Delta + \frac{D_2}{2D_3}$$

Finally, substitute (8-46) into (8-45) and (8-43) to obtain respectively z and x in terms of Δ . Then (8-41) yields a quadratic for Δ when these results and (8-46) are employed:

$$\Delta^2 \left[\frac{D_1^2}{D_3^2} (1 + \tan^2\theta_1) + \frac{Z_1^2 D_1^2}{Z_3^2 D_3^2} - 1 \right]$$

$$+ \Delta \left[\frac{D_1 D_2}{D_3^2} (1 + \tan^2\theta_1) + \frac{2Z_1 D_1}{Z_3^2 D_3^2} \left(\frac{Z_1 D_2}{2D_3 Z_3} + \frac{Z_2}{Z_3} \right) + 2\rho \right]$$

$$+ \left[\frac{D_2^2}{4D_3^2} (1 + \tan^2\theta_1) + \left(\frac{Z_1 D_2}{2Z_3 D_3} + \frac{Z_2^2}{Z_3} \right) - \rho_1^2 \right] = 0 \quad (8-47)$$

The smaller magnitude solution of this quadratic is the valid transponder delay bias.

As in the previous two sections, the per scan values of Δ must be averaged to obtain $\hat{\Delta}_n$. This value can then be used to modify the ranges:

$$\hat{\rho}_1 = \rho_1 - \hat{\Delta}_n$$

$$\hat{\rho}_2 = \rho_2 - \hat{\Delta}_n$$

and the method of section 8.4 employed to obtain the position values x, y and z. That is, after a smoothed Δ is known, an estimate of z can be found. This

estimate is entered into the moving average to obtain $\hat{z}$. Finally, this value is used in (8-1) and (8-2) to produce the x, y position.

8.9 Three Sensors, Altitude and Δ Both Unknown

With three sensor data available, only one azimuth measurement need be employed for computing x , y , z and Δ . As before, the azimuth to be used is that of the sensor whose measurement corresponds to the netting time. Also, that sensor's coordinate system is used for simplicity. The four equations thus become:

$$x^2 + y^2 + z^2 = (\rho_a - \Delta)^2 \quad (8-48)$$

$$(x - x_b)^2 + (y - y_b)^2 + (z - z_b)^2 = (\rho_b - \Delta)^2 \quad (8-49)$$

$$(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 = (\rho_c - \Delta)^2 \quad (8-50)$$

$$x = y \tan \theta_a \quad (8-51)$$

where

sensor a is the master sensor

x_b , y_b , z_b are the sensor b coordinates in the sensor a system

x_c , y_c , z_c are the sensor c coordinates in the sensor a system

These coordinates must be precomputed and stored in each sensor's computer for each other possible overlapping sensor. The equations that define them were given previously in (8-4).

The solution to these equations starts by subtracting (8-48) from (8-49), and using (8-51), to produce:

$$z = -y \left[\frac{x_b \tan \theta_a + y_b}{z_b} \right] + \Delta \left[\frac{\rho_b - \rho_a}{z_b} \right] + \left[\frac{d_b^2 - \rho_b^2 + \rho_a^2}{2z_b} \right] \quad (8-52)$$

Then subtract (8-48) from (8-50), and use (8-51) and (8-52) to eliminate x and z , to obtain:

$$y = \Delta \left[\frac{\rho_c - \rho_a - \frac{z_c}{z_b} (\rho_b - \rho_a)}{y_c + x_c \tan \theta_a - \frac{z_c}{z_b} (y_b + x_b \tan \theta_a)} \right] + 1/2 \left[\frac{d_c^2 - \rho_c^2 + \rho_a^2 - \frac{z_c}{z_b} (d_b^2 - \rho_b^2 + \rho_a^2)}{y_c + x_c \tan \theta_a - \frac{z_c}{z_b} (y_b + x_b \tan \theta_a)} \right] \quad (8-53)$$

$$\equiv \frac{Y_1}{Y_3} \tan \theta_a \Delta + \frac{Y_2}{2Y_3} \tan \theta_a$$

Substituting this into (8-52) yields:

$$z = \Delta \left[\frac{\rho_b - \rho_a}{z_b} - \frac{x_b \tan \theta_a + y_b}{z_b} \frac{Y_1}{Y_3} \right] + \left[\frac{d_b^2 - \rho_b^2 + \rho_a^2}{2z_b} - \frac{Y_2}{2Y_3} \left(\frac{x_b \tan \theta_a + y_b}{z_b} \right) \right] \quad (8-54)$$

$$\equiv Z_1 \Delta + Z_2$$

Finally, substituting the results (8-51), (8-53), and (8-54) into (8-48) produces a quadratic for Δ :

$$\Delta^2 \left[\frac{Y_1^2}{Y_3^2} (\tan^2 \theta_a + 1) + Z_1^2 - 1 \right] + \Delta \left[2\rho_1 + \frac{Y_1 Y_2}{Y_3^2} (\tan^2 \theta_a + 1) + 2 Z_1 Z_2 \right] + \left[\frac{Y_2^2}{4Y_3^2} (\tan^2 \theta_a + 1) + Z_2^2 - \rho_1^2 \right] = 0 \quad (8-55)$$

The smaller magnitude value of Δ is the valid solution.

Once Δ is obtained, the procedure of the last section is pursued. That is, find $\hat{\Delta}$ by smoothing, then compute z by section 8.5, then estimate $\hat{z}$, and finally solve for x and y via section 8.3.

8.10 Simulation Results

The various forms of multilateration presented in this chapter were tested against the 24 aircraft trajectory database described in Chapter 2. Since biases play a significant role in judging system performance, four separate runs, differing in their bias assumptions, were made. The actual bias values used in each run, along with the settings of other system parameters, are listed in Table 8-1. The performance results for the various tests are then presented in Tables 8-2 through 8-5.

Four performance statistics are used to compare the systems. The first two are the mean and standard deviation of the azimuth estimate produced by the multilateration algorithm. For a single, standalone sensor, the values that existed were

$$\theta_{\text{mean}} = -.001^\circ$$

$$\sigma_\theta = .058^\circ \text{ (1 milliradian)}$$

TABLE 8-1
SIMULATION RUN PARAMETERS

Parameter	Table 8-2	Table 8-3	Table 8-4	Table 8-5
Transponder Delay Bias	0 nm	.03 nm	0 nm	.03 nm
Altimeter Bias	0 ft.	0 ft.	300 ft.	300 ft.
Sensor 2 Position Bias	0 ft.	0 ft.	170 ft.	170 ft.
Sensor 1 Range Bias	0 ft.	0 ft.	30 ft.	30 ft.
σ_p	30 feet			
σ_θ	1 milliradian			
Sensor Blip/scan	.9			
Aspect Angle Cutoffs (8.12)	18°-162°			

TABLE 8-2
MULTILATERATION ACCURACY, NO BIASES

No. of Sensors	Alt. Known	Δ Computed	Azimuth		Scan-to-Scan Change	
			Mean	σ	Velocity	Heading
2	Yes	No	-.000°	.038°	29.2 kn	8.3°
		Yes	-.001°	.040°	30.4 kn	8.6°
	No	No	-.001°	.054°	32.2 kn	8.0°
		Yes	-.001°	.056°	33.4 kn	9.3°
3	Yes	No	-.000°	.048°	27.1 kn	7.1°
		Yes	.000°	.049°	27.6 kn	7.1°
	No	No	.001°	.056°	33.2 kn	7.7°
		Yes	-.006°	.063°	38.7 kn	8.9°
Incremental Bilateralation, Altitude Known			-.001°	.044°	31.1 kn	8.9°
Incremental Bilateralation, Altitude Unknown			-.000°	.042°	32.1 kn	11.2°

TABLE 8-3
MULTILATERATION ACCURACY, TRANSPONDER BIAS

No. of Sensors	Alt. Known	Δ Computed	Azimuth		Scan-to-Scan Change	
			Mean	σ	Velocity	Heading
2	Yes	No	$-.005^\circ$	$.070^\circ$	35.0 kn	9.4°
		Yes	$-.002^\circ$	$.040^\circ$	30.6 kn	8.5°
	No	No	$-.003^\circ$	$.050^\circ$	31.9 kn	8.9°
		Yes	$-.001^\circ$	$.057^\circ$	33.8 kn	9.3°
3	Yes	No	$-.013^\circ$	$.078^\circ$	42.7 kn	11.1°
		Yes	$-.003^\circ$	$.050^\circ$	27.7 kn	7.0°
	No	No	$-.015^\circ$	$.056^\circ$	33.2 kn	7.7°
		Yes	$-.012^\circ$	$.063^\circ$	38.8 kn	9.0°
Incremental Bilateration, Altitude Known			$-.001^\circ$	$.044^\circ$	31.2 kn	8.9°
Incremental Bilateration, Altitude Unknown			$-.000^\circ$	$.042^\circ$	32.3 kn	11.3°

TABLE 8-4

MULTILATERATION ACCURACY, SENSOR BIASES

No. of Sensors	Alt. Known	Δ Computed	Azimuth		Scan-to-Scan Change	
			Mean	σ	Velocity	Heading
2	Yes	No	$-.024^\circ$	$.063^\circ$	34.9 kn	9.4°
		Yes	$-.010^\circ$	$.043^\circ$	31.5 kn	8.9°
	No	No	$-.007^\circ$	$.054^\circ$	33.2 kn	9.2°
		Yes	$-.005^\circ$	$.059^\circ$	34.4 kn	9.5°
3	Yes	No	$-.025^\circ$	$.078^\circ$	38.7 kn	9.6°
		Yes	$.006^\circ$	$.068^\circ$	30.8 kn	7.9°
	No	No	$-.010^\circ$	$.061^\circ$	34.0 kn	8.0°
		Yes	$-.013^\circ$	$.066^\circ$	39.0 kn	9.0°
Incremental Bilateralation, Altitude Known			$-.001^\circ$	$.044^\circ$	31.3 kn	9.1°
Incremental Bilateralation, Altitude Unknown			$-.000^\circ$	$.042^\circ$	32.2 kn	11.2°

TABLE 8-5
MULTILATERATION ACCURACY, COMBINED BIASES

No. of Sensors	Alt. Known	Δ Computed	Azimuth		Scan-to-Scan Change	
			Mean	σ	Velocity	Heading
2	Yes	No	-.030°	.117°	44.6 kn	11.9°
		Yes	-.012°	.044°	31.9 kn	8.9°
	No	No	-.008°	.059°	34.7 kn	9.5°
		Yes	-.006°	.064°	34.8 kn	9.6°
3	Yes	No	-.038°	.125°	55.6 kn	14.4°
		Yes	-.005°	.066°	31.6 kn	8.1°
	No	No	-.026°	.064°	34.4 kn	8.0°
		Yes	-.020°	.070°	39.3 kn	9.2°
Incremental Bilateralation, Altitude Known			-.001°	.045°	31.5 kn	8.9°
Incremental Bilateralation, Altitude Unknown			-.000°	.043°	32.3 kn	11.3°

Thus, a netting algorithm has improved surveillance accuracy if it betters these numbers. In actuality, small mean errors are of no consequence to any ATC function, so that the standard deviation is the usual measure of data quality.

The second two statistics presented are the average deviations in the scan-to-scan measure of velocity and heading. That is, the velocity (or heading) determined by the newest and previous report positions is measured and compared with that determined by the previous two reports. Thus, these statistics measure the scan-to-scan jitter in positional estimates, and indicate the difficulty a tracker would have in producing smooth data. For reference, the single sensor values for these quantities were:

$$V_{dev} = 46.8 \text{ knots}$$

$$h_{dev} = 13.5^\circ$$

Again, netting should improve upon these numbers.

Each multilateration algorithm is applied on every scan in which two or more sensors supply raw reports. When only a single sensor's data is available on a scan, that raw report is coordinate converted to the primary sensor coordinates and used directly. Thus, the statistics in each row are for the total sequence of reports that would be output if the named multilateration algorithm were being employed. In particular, if the netted reports were biased differently from single sensor reports, the scan-to-scan statistics would be adversely affected.

Each table also includes the results for incremental bilateration. Although this algorithm is not presented until the next chapter, the data is included here to permit a comparison of its performance with multilateration. A more detailed discussion of these results is found in section 9.4.

8.10.1 No Biases

Table 8-2 presents the results obtained when both the aircraft transponder and sensor systems are assumed to contain no biases. As expected, all algorithms perform well, with only the azimuth statistics for the most complex algorithm not improved over single sensor data, and then only insignificantly worse.

The best data values are provided by the "normal" multilateration algorithms, in which altitude is known and biases are assumed to not exist. Since these conditions match reality, their superiority is no surprise. Clearly, the algorithms that attempt to compute the altitude, or the transponder delay, or both, will not be as accurate, as no estimate can match perfect knowledge. Within each group of algorithms, the performance ordering is as expected: altitude known superior to altitude estimation, transponder delay ignored superior to transponder delay computation.

The slight superiority of two sensor algorithms over three sensor ones is also expected. Since the secondary sensor is chosen to be located more advantageously than the tertiary sensor, its data will be more helpful for azimuth netting. Three sensor estimates of altitude, however, were previously shown to be far superior.

It is important to note that when altitude knowledge is assumed to be absent for the aircraft, multilateration still provides excellent azimuth accuracy. Thus netting can provide improved surveillance data on all aircraft, altitude reporting or not.

8.10.2 Transponder Delay Bias

For the next test, an assumed aircraft transponder delay bias of 0.03 miles was added to all sensor raw reports. Table 8-3 presents the revised results for each of the multilateration algorithms. As expected, those algorithms that specifically consider this bias to be present, and solve for it, produced results statistically unchanged from the previous test. To them, a value 0.03 is no more challenging than one of 0.

The "normal" multilateration formulas, on the other hand, were significantly degraded by the presence of this delay. For both two and three sensor cases, they performed worse than any other algorithms.

Perhaps surprisingly, the algorithms that assume altitude knowledge is lacking were all unaffected by the presence of this delay bias. The reason for this effect, explained in section 6.3, is that these algorithms can absorb the bias by adjusting the computed aircraft altitude.

Finally, neither of the incremental bilateration algorithms was affected by the bias, thereby providing more evidence to support the bias-resistant claims made for this approach.

8.10.3 Sensor Biases

The third test removed the transponder bias, but replaced it with various sensor position and measurement biases. Table 8-4 presents the performance results for this situation. In this case, no multilateration algorithm matches the real system, so it is not surprising that all were degraded from the no bias system of Table 8-2.

The major result of adding these sensor biases is the introduction of mean azimuth errors for all the multilateration algorithms. Further testing showed that this mean error was proportional to the size of the biases; doubling the biases doubled the mean error. The mean error, because of the section 6.3 argument, was worst for the "normal" multilateration formulas.

The azimuth standard deviation was also increased for all multilateration algorithms. This statistic for the "normal" formulas was also found to be proportional to the size of the biases. The other formulas, however, had only

a very slow increase with increasing biases. The reason, again, is their ability to absorb these biases into altitude or transponder bias calculations, instead of fully in the azimuth estimate.

The incremental bilateration formulas were again totally unaffected by the presence of the system biases. No mean azimuth error was produced, and the standard deviation was unchanged.

8.10.4 Combined Biases

By returning the transponder delay bias, producing a system with both aircraft and sensor biases, the multilateration algorithms were further degraded, as shown in Table 8-5. The "normal" formulas now produced a significant mean azimuth error, along with an azimuth standard deviation and scan-to-scan performance worse than that of single sensor surveillance. The other formulas were only degraded to a minor degree.

Incremental bilateration, as expected by now, was unaffected.

8.10.5 Results Summary

Two major results concerning the accuracy of multilateration have been uncovered by these simulation tests. First, the "normal" multilateration equations produce acceptable performance only in perfect systems, namely those containing no aircraft or sensor biases. In any other case, some "give" must be built into the formulas. Since the transponder delay bias is the major bias encountered in practice, and since incorporating it as a variable in the formulas provides this necessary accommodation, the " Δ computed" formulas are recommended. This recommendation holds whether aircraft altitude is known or unknown.

Second, incremental bilateration is superior to any of the forms of multilateration investigated.

8.11 Multilateration in Diffraction Zones

Normal multilateration formulas don't employ a sensor azimuth measurement. Thus, the presence of diffraction will not affect their accuracy at all. When the aircraft altitude or transponder delay bias must be estimated, however, use of one or more azimuth measurements could be required. Thus, some of the multilateration algorithms of this chapter may be affected when the aircraft passes through a diffraction zone. Also, incremental bilateration always uses the primary sensor azimuth for its computations, so it would appear to be most susceptible to diffraction.

To examine the effects of diffraction on the various algorithms, the no bias and full bias tests were repeated, with the σ_θ of the primary sensor increased to 5 milliradians as a model of the severe azimuth noise found in diffraction zones. As described in section 2.4, the simulated diffraction tests are conducted by preceding each diffraction zone segment (assumed to be

20 scans in length) by a normal data initialization segment (assumed to be 5 scans in length). This initialization period permits algorithms that estimate transponder delay (Δ) or altitude (or both) to produce values approaching steady state before encountering the diffraction conditions. This allowance matches expected real conditions, as netting is designed to be initiated whenever an aircraft nears a known diffraction zone.

With case 2 timing, the apparent case of choice, used for netting, altitude and transponder delay estimates require the primary sensor azimuth. Thus new estimate values cannot be generated in the diffraction zone. One method of proceeding would be to maintain unchanged the smoothed values existing upon entry to the zone. This would produce smooth scan-to-scan surveillance. Unfortunately, the altitude value would lose accuracy, thereby affecting azimuth performance, if the aircraft were changing altitude. Also, even though the true Δ would not change, additional samples tend to produce a more precise estimate.

Thus, continuing to produce altitude and transponder delay estimates in the zone is the preferable method of operation. This can be accomplished by changing to case 1 timing for estimation purposes while in the diffraction zone. This permits estimate values to be generated corresponding to each non-primary sensor report, utilizing that sensor's non-diffraction-corrupted azimuth. Position determination, however, will still be made only at primary sensor times.

The results of the diffraction simulation tests are presented in Tables 8-6 and 8-7. These results should be compared to those in Tables 8-2 and 8-5 respectively to determine the effects of diffraction. One obvious, and perhaps surprising, result is that scan-to-scan consistency has improved for every multilateration algorithm. The reason for this improvement is that, since primary sensor raw reports cannot be used for output due to their corrupted azimuths, a coast occurs for any scan on which secondary sensor data is missing. This eliminates the jumps between raw primary and netted data that has been observed before. Of course, whenever a target is maneuvering, this coast will lead to a major tracker error, and thus it is not a preferred mode of operation.

The other main result is that the only multilateration algorithms affected adversely by diffraction are the 2-sensor altitude-estimation ones. This indicates that accuracy is sacrificed when the estimates are made from a secondary sensor point-of-view. This follows from the expectation that the primary sensor is closest to the target, and thus has the largest elevation angle. Even these algorithms still perform quite well however, so multilateration is clearly a feasible approach to overcoming surveillance problems in diffraction zones.

The incremental bilateration algorithms, as seen, are virtually unaffected by diffraction. The reasons for this result are discussed in the next chapter.

TABLE 8-6

MULTILATERATION ACCURACY, DIFFRACTION, NO BIASES

No. of Sensors	Alt. Known	Δ Computed	Azimuth		Scan-to-Scan Change	
			Mean	σ	Velocity	Heading
2	Yes	No	-.000°	.027°	20.8 kn	6.3°
		Yes	-.000°	.030°	22.4 kn	6.5°
	No	No	-.007°	.099°	25.8 kn	7.3°
		Yes	-.005°	.099°	25.8 kn	7.1°
3	Yes	No	.000°	.048°	25.6 kn	6.8°
		Yes	.001	.049°	25.8 kn	6.8°
	No	No	-.001°	.062°	30.5 kn	7.1°
		Yes	-.008°	.060°	33.8 kn	7.4°
Incremental Bilateralation, Altitude Known			-.002°	.047°	21.5 kn	6.5°
Incremental Bilateralation, Altitude Unknown			.001°	.064°	24.1 kn	8.3°

TABLE 8-7

MULTILATERATION ACCURACY, DIFFRACTION, COMBINED BIASES

No. of Sensors	Alt. Known	Δ Computed	Azimuth		Scan-to-Scan Change	
			Mean	σ	Velocity	Heading
2	Yes	No	$-.039^\circ$	$.127^\circ$	29.0 kn	8.2°
		Yes	$-.027^\circ$	$.037^\circ$	23.0 kn	6.6°
	No	No	$-.043^\circ$	$.154$	27.7 kn	7.7°
		Yes	$-.038^\circ$	$.125^\circ$	26.3 kn	7.5°
3	Yes	No	$-.040^\circ$	$.126^\circ$	53.6 kn	14.0°
		Yes	$-.004^\circ$	$.065^\circ$	29.5 kn	7.6°
	No	No	$-.032^\circ$	$.090$	33.9 kn	7.7°
		Yes	$-.027^\circ$	$.084$	36.2 kn	8.1°
Incremental Bilateralation, Altitude Known			$.001^\circ$	$.059^\circ$	22.9 kn	6.5°
Incremental Bilateralation, Altitude Unknown			$.003^\circ$	$.068^\circ$	24.3 kn	8.3°

8.12 Effect of Aspect Angle

The azimuth estimation accuracy of 2-sensor multilateration, more than that for any other form of netting, is determined to a large degree by the aspect angle of the sensors relative to the target. This angle, defined by Fig. 8-2, is the difference in target viewing direction of the two sensors. Multilateration, in its simplest concept, uses the range of the second sensor as a substitute for the azimuth of the first. Thus, the closer the aspect angle is to 90° , the closer this secondary range measurement is to coordinate alignment with the primary azimuth. In the limit, as shown earlier in Fig. 1-9, the azimuth estimation error becomes identically equal to that of the secondary sensor range.

For aspect angles not exactly 90° , the azimuth error is a function of the range errors of both sensors. Furthermore, the magnification factor for these errors grow rapidly as the aspect angle nears 180° (or equivalently, nears 0°). In the worst case, at exactly 180° (or 0°), the derivatives of azimuth error to range errors become infinite. Conceptually, at these aspect angles, neither sensor range has a component in the azimuth direction, so no estimate is possible.

To quantify the effect of aspect angle on 2-sensor multilateration accuracy, Table 8-8 presents the multilateration azimuth standard deviation as a fraction of that of the primary sensor azimuth measurement. Thus, values < 1.0 signify improvement. This data is for the standard 24 aircraft trajectories, with no biases. As seen, for any given range band, the netting improvement tends to drop off as the aspect angle diverges from 90° .

In addition, the netting data, as expected, is best at longer ranges, since the azimuth error is determined as:

$$\theta_{\text{err}} = \frac{\text{cross-range error}}{\rho}$$

and the cross-range coordinate is the quantity actually calculated by multilateration. In general, the cross-range error with multilateration is nearly independent of range.

This table indicates why multilateration is not recommended for targets near the primary sensor, and should only be used when secondary sensors are located off to the side of the target. In particular, all multilateration results presented in this paper have been generated by screening from consideration any target reports satisfying either:

(a) $\rho < 25 \text{ nm}$, or

(b) $|\phi_{\text{aspect}} - 90^\circ| > 72^\circ$

Without this filter, netting could have appeared to have produced worse azimuth accuracy than single sensor data. In fact, as shown in numerous tables, netting, when appropriate geometry exists, is a definite surveillance improvement technique.

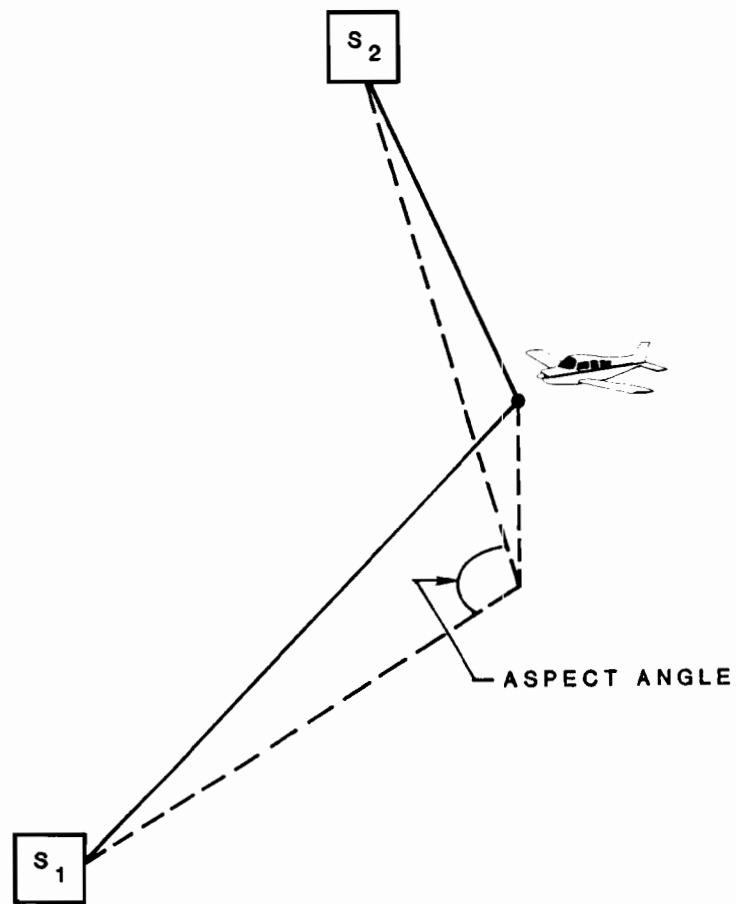


Fig. 8-2. Aspect angle definition.

TABLE 8-8

MULTILATERATION ACCURACY VS. GEOMETRY

Aspect Angle	Range			
	0-25 nm	25-50 nm	50-75 nm	75-100 nm
80°-100°	1.04	.64	-	-
60°-80° 100°-120°	1.25	.69	.65	-
40°-60° 120°-140°	1.19	.77	.70	-
20°-40° 140°-160°	3.11	1.02	.83	.76
0°-20° 160°-180°	6.56	2.13	.77	.71

$$\text{Entry} = \frac{\sigma_{\theta} \text{ netting}}{\sigma_{\theta} \text{ sensor}}$$

9.0 FLAT EARTH INCREMENTAL BILATERATION

Standard two-sensor multilateration, as just seen, requires four different algorithms to handle the ensemble of aircraft characteristics expected to be encountered by a Mode S sensor. Furthermore, none of these algorithms is designed to handle possible sensor biases; all are degraded by their presence. This chapter describes a single algorithm based on the incremental tracking discussions presented earlier that is as accurate as any of these approaches in a "perfect" system, and is capable of maintaining this good performance irrespective of aircraft or system biases.

The key to this use of incremental bilateration is the development of a flat earth model equivalent to the standard spherical earth one. This chapter develops this model in detail. It then presents the method of extending this model to produce the bias-resistant algorithm being sought. Finally, the specific algorithms to be used for surveillance netting are provided, including the method of operating in diffraction zones.

As part of this development, the issues of transponder turnaround bias and altitude estimation are discussed. Both are possible with this model, although accurate surveillance is shown to occur if either or both is simply treated as an unknown bias. Results on the netting database are presented to support the conclusions of this chapter.

9.1 Spherical-Equivalent Flat Earth

Mathematically, planar formulas are simpler to use than spherical ones. However, if the sensor measurements must be transformed for a planar representation, this simplicity is lost. Thus, a model that permits planar mathematics on the raw sensor data is desired. The particular equation we desire to employ, as shown in Fig. 9-1, is as follows:

$$\rho_{2g}^2 = \rho_{1g}^2 + d^2 - 2\rho_{1g} d \cos \phi_1 \quad (9-1)$$

where

$$\rho_{1g} = \sqrt{\rho_1^2 - z_1^2}$$

z_1 = altitude above sensor 1

d = distance in the plane between the sensors

ϕ_1 = angle between inter-sensor line and sensor 1 measurement

From this, the azimuth to be determined from the bilateration is given by:

$$\theta_1 = \psi_{12} \pm \cos^{-1} \left[\frac{\rho_{1g}^2 + d^2 - \rho_{2g}^2}{2\rho_{1g} d} \right] \quad (9-2)$$

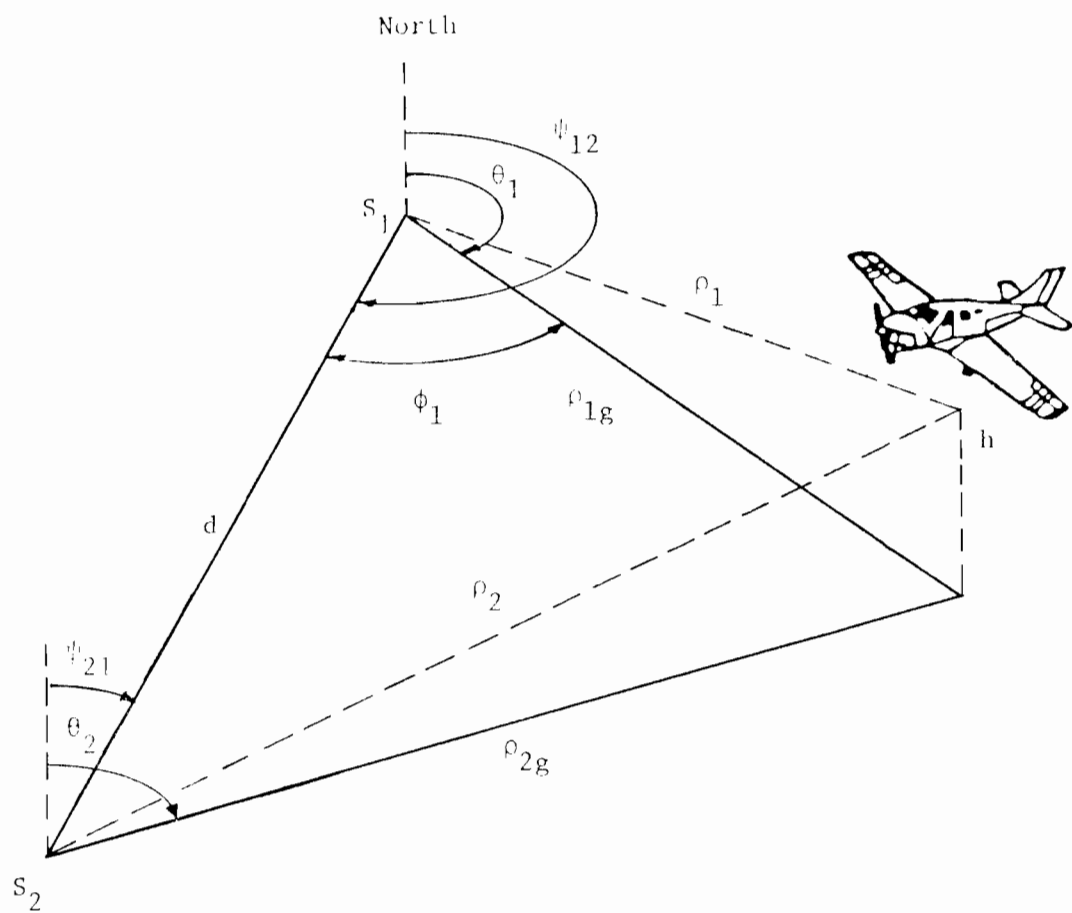


Fig. 9-1. Planar mathematics diagram.

where ψ_{12} is the sensor 1 azimuth of the inter-sensor line, and the correct sign is a function of the sensor and aircraft geometry. Note that (9-1) and (9-2) will apply for sensors at any height above the plane.

The property of a planar system that supports equations (9-1) and (9-2) is the alignment of the local x, y coordinate systems of the two sensors. One way to express this alignment is that, for any target location:

$$\begin{aligned} |x_1 - x_2| &= d_x = x \text{ component of } d \\ |y_1 - y_2| &= d_y = y \text{ component of } d \end{aligned} \quad (9-3)$$

where $x_i = \rho_i \sin \theta_i$

$$y_i = \rho_i \cos \theta_i$$

This section now presents a method of introducing property (9-3) to a spherical earth, allowing formula (9-2) to be used.

To simplify the discussion, with no loss of generality, assume as shown in Fig. 9-2 that sensor 1 is directly north on the sphere from sensor 2. This condition can always be produced by rotating the sensor 1 coordinate system by $\pi + \psi_{12}$ and the sensor 2 coordinate system by ψ_{21} , where ψ_{ij} is the azimuth of sensor j in sensor i's coordinate system. Note that $\psi_{21} \neq \pi + \psi_{12}$ because the earth is not flat.

Now place a hypothetical sensor 0 on the earth's surface midway between the real sensors and use its local coordinate system as the master. The transformed master coordinates x_0 , y_0 , and z_0 of the target at the two real sensors are then (see Fig. 9-3):

$$\begin{aligned} x_1 &= x_0 \\ y_1 &= y_0 \cos \frac{\Phi}{2} - z_0 \sin \frac{\Phi}{2} - E \sin \frac{\Phi}{2} \\ z_1 &= y_0 \sin \frac{\Phi}{2} + z_0 \cos \frac{\Phi}{2} + E \cos \frac{\Phi}{2} - (E+h_{s1}) \\ x_2 &= x_0 \\ y_2 &= y_0 \cos \frac{\Phi}{2} + z_0 \sin \frac{\Phi}{2} + E \sin \frac{\Phi}{2} \\ z_2 &= -y_0 \sin \frac{\Phi}{2} + z_0 \cos \frac{\Phi}{2} + E \cos \frac{\Phi}{2} - (E+h_{s2}) \end{aligned} \quad (9-4)$$

where Φ , as shown in the figure, is the angle between the two sensors. A derivation of the general local coordinate system to local coordinate system

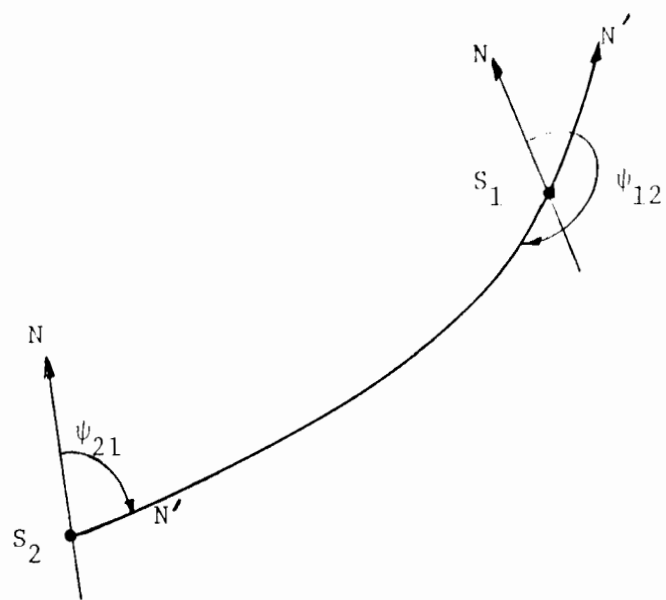


Fig. 9-2. Sensor alignment rotation.

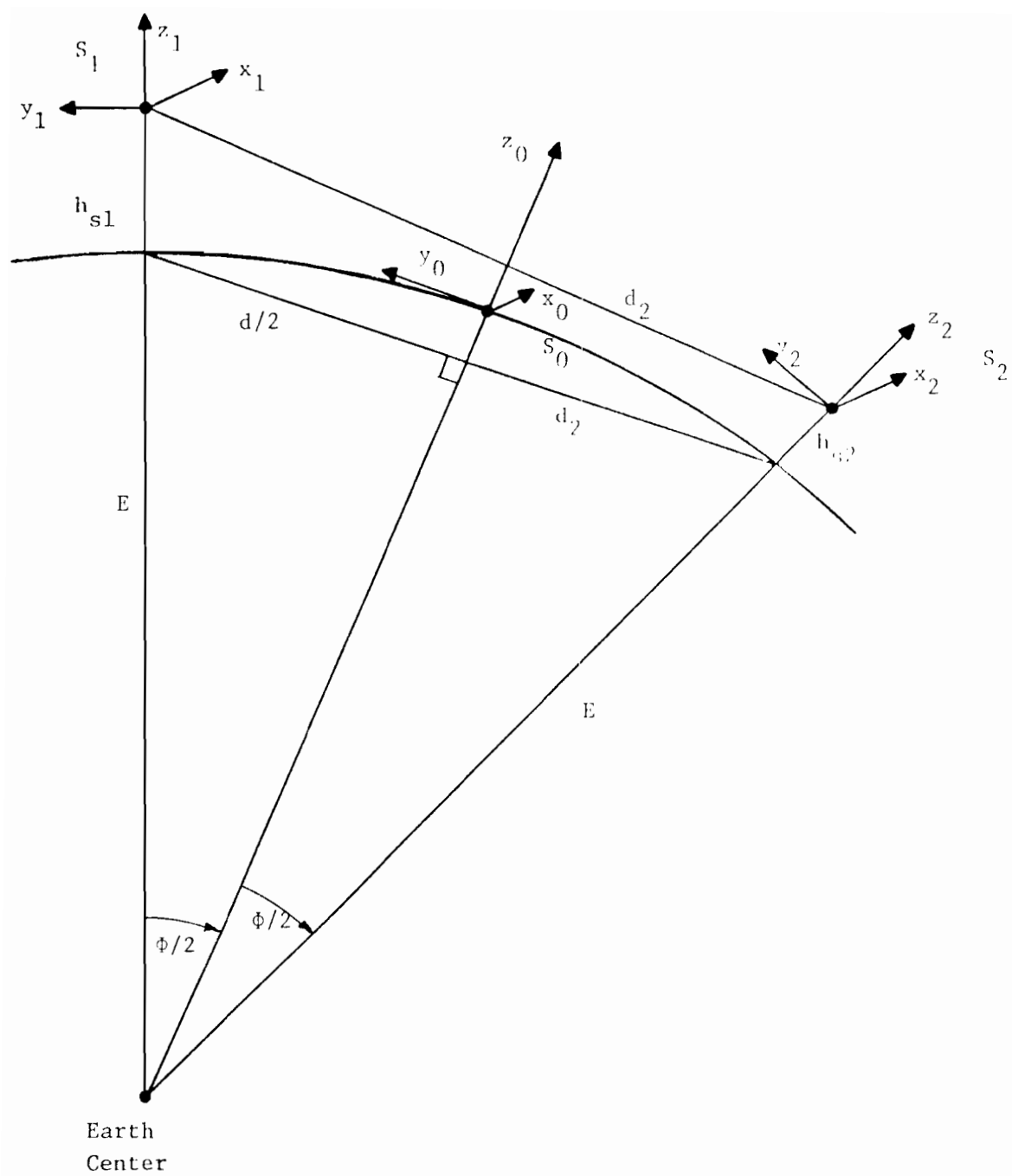


Fig. 9-3. Transformed sensor coordinates.

transformation is given in [7]; (9-4) is the simplification that results from our directly-north assumption.

Taking the x and y differences yields:

$$|x_1 - x_2| = 0$$

$$|y_1 - y_2| = 2z_0 \sin \frac{\phi}{2} + 2E \sin \frac{\phi}{2}$$

$$= 2z_0 \frac{d/2}{E} + 2E \frac{d/2}{E}$$

$$D = d(1 + \frac{z_0}{E}) \quad (9-5)$$

where, as shown in Fig. 9-3, d is the distance between the earth surface locations of the two sensors. Appendix D shows how this distance d can be expressed in terms of the measured distance d₂ between the sensors. Thus, property (9-3) is realized if the secondary sensor is assumed to be located, not at its true distance of d₂ from the primary sensor, but at the expanded distance D given by (9-5).

This section's discussion serves as an informal proof of the following theorem:

Spherical-Equivalent Flat Earth Theorem

Given: a two sensor system defined by:

(a) a spherical earth model

(b) sensor 1 located at latitude λ_1 , longitude ℓ_1 , and height h_{s1}

(c) sensor 1 yielding measurements ρ_1 , θ_1 , and

$$z_1 = \frac{(h-h_{s1})^2 + 2(h-h_{s1})(E+h_{s1}) - \rho_1^2}{2(E+h_{s1})}$$

(d) sensor 2 located at latitude λ_2 , longitude ℓ_2 , and height h_{s2} ,

being at a position at range d₂ and azimuth ψ_{12} relative to sensor 1

(e) sensor 2 yielding measurements ρ_2 , θ_2 , and

$$z_2 = \frac{(h-h_{s2})^2 + 2(h-h_{s2})(E+h_{s2}) - \rho_2^2}{2(E+h_{s2})}$$

Then: an equivalent two sensor system can be defined by:

- (f) a flat earth model
- (g) sensor 1 located at $x_{s1} = y_{s1} = z_{s1} = 0$
- (h) sensor 1 yielding the same measurements ρ_1 , θ_1 , and z_1 as in (c)
- (i) sensor 2 located at x_{s2} , y_{s2} , and $z_{s2} = 0$, with x_{s2} and y_{s2} being the values that locate sensor 2 relative to sensor 1 at adjusted range

$$D = \left(1 + \frac{z_0}{E}\right) \left[d_2 - \frac{(h_{s1} - h_{s2})^2}{2d_2} - \frac{(h_{s1} + h_{s2})d_2}{2E} + \frac{(h_{s1} + h_{s2})(h_{s1} - h_{s2})^2}{2d_2 E} \right]$$

and same azimuth ψ_{12} as in (d), where

$$z_0 = h + \frac{h^2 - \rho_0^2}{2E}$$

and ρ_0 is the slant range of the target from the location given by latitude $(\lambda_1 + \lambda_2)/2$, longitude $(\ell_1 + \ell_2)/2$, and height $h=0$ of the original spherical earth model

- (j) sensor 2 yielding the same measurements ρ_2 , θ_2 , and z_2 as in (e).

This equivalence applies even if the aircraft altitude h is unknown. Using any estimate for h , the spherical and spherical-equivalent flat earth models will produce identical results. Since some assumption on h is required to use this approach, a value of 0.5 miles has been chosen. Section 9.6 discusses this choice in more detail.

The apparent problem with this theorem is the need to determine ρ_0 . A precise calculation of ρ_0 requires spherical earth mathematics, which we are trying to avoid, and knowledge of θ_1 , which we are trying to calculate. Fortunately, ρ_0 need only be known approximately. A simple derivative on the z_0 equation shows that:

$$\left| \frac{\partial z_0}{\partial \rho_0} \right| = \frac{\rho_0}{E}$$

Altitudes in beacon systems are quantized to 100 foot intervals. To remain within this accuracy, ρ_0 need only be good to one-third mile for targets anywhere within the maximum 200 mile coverage region.

Thus, a simple calculation that approximates ρ_0 to within this interval can be made by assuming that the earth is flat. As shown in Fig. 9-4:

$$\rho_0^2 = \rho_1^2 + \left(\frac{d}{2}\right)^2 - 2\rho_1 \left(\frac{d}{2}\right) \cos(\theta_1 - \psi_{12}) \quad (9-6)$$

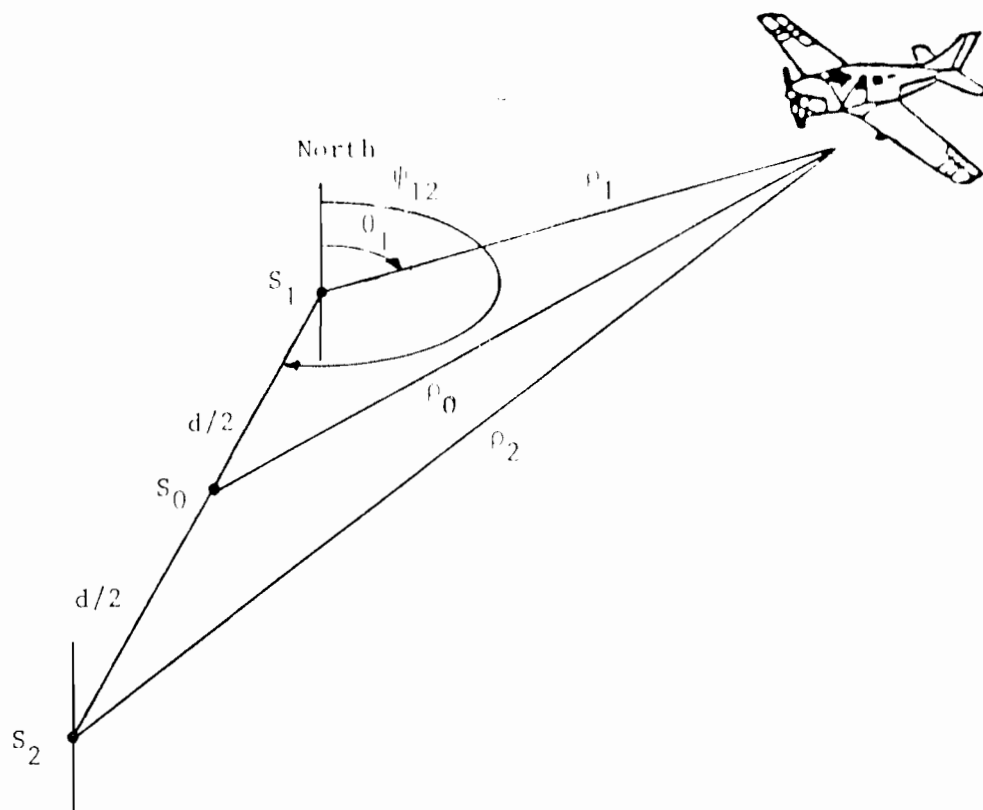


Fig. 9-4. Simplified ρ_0 calculation.

Then since

$$\left| \frac{\partial \rho_0}{\partial \theta_1} \right| < \frac{\rho_1 d}{2\rho_0}$$

the required accuracy of ρ_0 can generally tolerate a θ_1 error of as much as a full degree. Sensor azimuth errors are certainly within this tolerance, even in diffraction zones.

A summary of the two approaches to bilateration, spherical earth and flat earth, are presented in Figs. 9-5 and 9-6. Optimally coded, and realizing that the required accuracy of the cosine function can be achieved by a lookup table of 160 entries (spaced .01 radians over a quadrant), the flat earth approach requires slightly less computation time. However, the true power of the approach is that it can easily be extended to incorporate the extended incremental algorithms for overcoming bias errors, whereas the traditional approach can not.

9.2 Extended Flat Earth Model

Now that a flat earth model has been developed, the bias-resistant approach developed earlier in Section 6.4 can be implemented. The effect of the apparent change in the secondary sensor location required in this approach is a modification of the quantities D and ψ_{12} used in Fig. 9-6. With this change, the bilateration algorithm will be referred to as the extended flat earth model.

Each scan, a biased distance D_b and azimuth ψ_{12}^b to the secondary sensor can be computed as shown earlier in (6-2) and (6-3). To eliminate the effects of measurement noise, the actual values used in the bilateration are smoothed over a few scans:

$$D_{new} = \frac{n * D^{old} + D^j}{n+1} \quad (9-7)$$

$$\psi_{12}^{new} = \frac{n * \psi_{12}^{old} + \psi_{12}^j}{n+1}$$

where superscript old means the value used on scan $j-1$, and superscript j means the values computed with the current scan sensor measurements. These values are then used, as shown in Fig. 9-6, to compute the aircraft position.

Should one or the other sensor not have a report on the current scan, of course, the values of D and ψ_{12} are left unchanged and no bilateration is performed. If sensor 1 was the only reporting sensor, its ρ_1 and θ_1 are used directly; if sensor 2 supplied the report, its data is transformed using the existing D and ψ_{12} values and the flat earth model to provide ρ_1 and θ_1 as follows (see Fig. 9-1):

Precomputed:

$x_2, y_2, z_2, d_2, \psi_{12}$ of sensor 2 relative to sensor 1

$$A = \left(\frac{x_2^2}{y_2^2} \right) + 1$$

Per Scan Computations:

$$z = \frac{(h-h_{s1})^2 + 2(h-h_{s1})(E+h_{s1}) - \rho_1^2}{2(E+h_{s1})}$$

$$D_2 = d_2^2 + \rho_1^2 - \rho_2^2 - 2zz_2$$

$$B = \frac{-x_2}{y_2^2} D_2$$

$$C = \left(\frac{D_2^2}{2y_2^2} \right) + z^2 - \rho_1^2$$

$$R = \sqrt{B^2 - 4AC}$$

$$x_+ = \frac{-B + R}{2A}$$

$$y_+ = \frac{-x_2}{y_2} x_+ + \frac{D_2}{2y_2}$$

$$\theta_{1+} = \tan^{-1} \left(\frac{x_+}{y_+} \right)$$

$$\theta_{1-} = 2\psi_{12} - \theta_{1+}$$

choose from θ_{1+} and θ_{1-} the one closer to the measured θ from sensor 1.

Fig. 9-5. Spherical earth calculations.

Precomputed:

d_2 and ψ_{12} of sensor 2 relative to sensor 1

$$d = d_2 - \frac{(h_{s1} - h_{s2})^2}{2d_2} - \frac{(h_{s1} + h_{s2})d_2}{E} + \frac{(h_{s1} - h_{s2})^2 (h_{s1} + h_{s2})}{2d_2 E}$$

Per Scan Computations:

$$z_0 = h + \frac{h^2 - \rho_1^2 - (d/2)^2 + \rho_1 d \cos(\theta_1 - \psi_{12})}{2E}$$

$$D = d \left(1 + \frac{z_0}{E}\right)$$

$$z_i = \frac{(h - h_{si})^2 + 2(h - h_{si})(E + h_{si}) - \rho_i^2}{2(E + h_{si})} \quad i=1, 2$$

$$\rho_{lg} = \sqrt{\rho_1^2 - z_1^2}$$

$$C = \cos^{-1} \left[\frac{\rho_{lg}^2 + D^2 - \rho_2^2 + z_2^2}{2\rho_{lg}D} \right]$$

$$\theta_{1+} = \psi_{12} + C$$

$$\theta_{1-} = \psi_{12} - C$$

choose from θ_{1+} and θ_{1-} the one closer to the measured θ from sensor 1.

Fig. 9-6 Flat earth calculations.

$$\rho_{1g} \approx \sqrt{D^2 + \rho_{2g}^2 + 2D\rho_{2g} \cos(\theta_2 - \psi_{12})} \quad (\psi_{21} \approx \psi_{12} - \pi)$$

$$z_1 = \frac{(h-h_{s1})^2 + 2(h-h_{s1})(E+h_{s1}) - \rho_{1g}^2}{2(E+h_{s1})} \quad (\rho_1 \approx \rho_{1g})$$

$$\rho_1 = \sqrt{\rho_{1g}^2 + z_1^2} \quad (9-8)$$

$$\theta_1 = \psi_{12} \pm \cos^{-1} \left[\frac{\rho_{1g}^2 + D^2 - \rho_{2g}^2}{2\rho_{1g}D} \right] \quad (9-9)$$

The positional error produced by these approximations is negligible.

The key parameter of the model is the number of scans over which to average the apparent secondary sensor position. A smaller value of n permits better tracking of the apparent secondary sensor motion caused by the system biases. On the other hand, a larger value of n provides less susceptibility to measurement noise. This tradeoff is reflected in the data produced by the bilateration: smaller n provides better azimuth accuracy, while larger n produces less scan-to-scan jitter. In general, accuracy is preferred, as the tracker fed the bilateration result should be responsible for smoothing the data. Thus, we have found that best overall surveillance is produced by using an n of 2 or 3, with the value of 2 used with low variance sensors (such as Mode S) and 3 used with current ATCRBS sensors.

It should be pointed out that the smoothing of secondary sensor location discussed here is quite different in effect from the smoothing of aircraft motion in a surveillance tracker. In a tracker, the smoothing prevents immediate following of aircraft maneuvers; the perceived aircraft motion itself is being averaged. In this netting algorithm, the smoothing is of an essentially stationary object, the secondary sensor; the raw data (the two ranges) is employed as measured, and aircraft turns are acquired immediately. Thus, no loss of responsiveness results from the extended flat earth approach.

9.3 Diffraction Zone Algorithm

When an aircraft enters a diffraction zone, the sensor azimuth becomes extremely noisy. Thus, if this measurement were to be used in computing the D^j and ψ_{12}^j values, the D^{new} and ψ_{12}^{new} values produced by (9-7) after averaging would still possess extreme jitter. To prevent the resulting poor bilateration performance, an algorithm modification is required for diffraction zones.

First, even in clear airspace regions, the normal jitter can be reduced while improving accuracy at the same time. This is done by noting that the apparent sensor motion has two components: one due to changes in z_0 , the other to system biases. Since z_0 can be computed each scan from the reported altitude, the first component can be eliminated. Thus the revised averaging procedure becomes:

$$D^{\text{new off}} = \frac{n * D^{\text{old off}} + D^{\text{j off}}}{n+1}$$

$$D^{\text{new}} = D^{\text{no bias}} + D^{\text{new off}}$$

$$\psi_{12}^{\text{new off}} = \frac{n * \psi_{12}^{\text{old off}} + \psi_{12}^{\text{j off}}}{n+1} \quad (9-10)$$

$$\psi_{12}^{\text{new}} = \psi_{12}^{\text{no bias}} + \psi_{12}^{\text{new off}}$$

where the superscript "no bias" means the value computed each scan, using z_0 , according to the Spherical Equivalent Flat Earth Theorem, and the superscript "j off" is the offset from that expected value found on the current scan:

$$D^{\text{j off}} = D^{\text{j}} - D^{\text{no bias}}$$

$$\psi_{12}^{\text{j off}} = \psi_{12}^{\text{j}} - \psi_{12}^{\text{no bias}}$$

This revised procedure becomes particularly useful when an aircraft is passing through a diffraction zone. At such a time, the measured azimuth is especially inaccurate, as illustrated earlier. Thus, the per scan diffraction-caused jitter of the secondary sensor location will far exceed the motion due to system biases. As a result, the best procedure in diffraction zones is given by:

$$D^{\text{new}} = D^{\text{no bias}} + D^{\text{old off}}$$

$$\psi_{12}^{\text{new}} = \psi_{12}^{\text{no bias}} + \psi_{12}^{\text{old off}} \quad (9-11)$$

That is, the average offset values existing upon entry to the zone are maintained without update throughout the zone.

This procedure will produce accurate surveillance in diffraction zones provided first, the zone traversal distance is reasonably short, and second, the system biases are small to moderate. Otherwise, "drift" can occur in the positions produced by the bilateration.

Data presented last chapter in Tables 8-6 and 8-7 illustrated that with this procedure, surveillance in diffraction is nearly as accurate as surveillance in normal regions. This result held for both perfect and typically biased systems. The number of scans assumed for the bias zone traversal, namely twenty, provided for greater aircraft movement in diffraction than would be possible in most real world situations, and thus the conclusion is justified.

When the two conditions above are violated, that is when an aircraft travels a great distance in a diffraction zone in a system with major sensor biases, it is possible that substantial azimuth errors can result from this approach. This will occur when the geometry at zone exit is sufficiently different from that at zone entrance that the biases will have produced a significant shift in the apparent secondary sensor location. An alternate procedure that can overcome this problem is, instead of maintaining constant offset values throughout the zone, computing new ones each scan using the netting-derived primary sensor azimuth. That is, after calculating $\hat{\theta}_1$ by bilateration, use it, instead of the measured value θ_1 , in the formulas (6-2) and (6-3) to produce values of D_j^j and ψ_{12}^j for the current scan. Then the usual averaging procedure (9-10) can be employed to update the offset values.

The one potential problem with this procedure is positive feedback. That is, if $\hat{\theta}_1$ has an error, it will produce errors in the offset values, which can cause a larger error in the next scan's $\hat{\theta}_1$, and so forth. As a result, drift can occur in this method as well. Simulation tests of the two approaches, for 100-scan diffraction zones and larger system biases, have shown that this revised method is reasonably stable, and slightly superior to the constant offset one. The improvement, however, was not worth the extra effort in general.

A different conclusion was reached, though, for aircraft whose altitude was unknown. By assuming an altitude of 0.5 miles, a large bias will exist for aircraft substantially above this level. Also, when aircraft climb or descend in the diffraction zone, this bias will change dramatically. Thus, it is quite possible that a combination of these two facts will cause an assumption of constant offset values throughout the zone to produce large error drifts. The revised procedure, in this case, has been found via simulation tests to be far superior. Thus it is recommended for non-altitude-reporting aircraft, and was the procedure used to generate the results in the last row of Tables 8-6 and 8-7.

9.4 Performance of the Extended Flat Earth Model

The extended flat earth model just presented has been tested on both real and simulated data. The model has uniformly outperformed the usual bilateration techniques whenever biases have existed in the system. These results have been previously documented in Tables 8-1 through 8-7. Two more examples of the performance of the extended flat earth approach, with different types of input data, are presented in Table 9-1.

This table first compares the spherical multilateration and extended flat earth algorithms when system measurements from actual operational enroute (NAS) air traffic control sensors were used. This data had unknown biases, although great care had been taken to attempt to remove them. The statistics chosen for exposition were the average and variance of the azimuth error, and the average scan-to-scan heading and velocity errors. As before, the first pair indicates the accuracy and consistency of the bilateration estimates, while the second pair provides an estimate of the difficulty a tracker would experience in smoothing the data. The results clearly indicate the superiority of the extended flat earth approach.

The second half of the table presents results for the standard 24 simulated aircraft trajectories with the biases specified in Fig. 8-1 included. In addition, a primary sensor azimuth bias of one milliradian was added. Since the extended flat earth is an incremental algorithm, this bias is maintained in its data. The standard multilateration algorithm is unaffected by this azimuth bias. However, the sensitivity of this model to the other system biases has in fact resulted in an average azimuth error that is a significant fraction of the bias, as well as producing a far larger variance than the extended flat earth model.

Furthermore, since the netted data with multilateration has a different north reference than the primary data, the very troublesome data jumps depicted in Fig. 9-7 occur whenever secondary sensor data is missing and primary data must be used directly. With the incremental extended flat earth approach, both netted and primary data have the same reference. Thus, as shown in Fig. 9-8, no data jumps occur in this case.

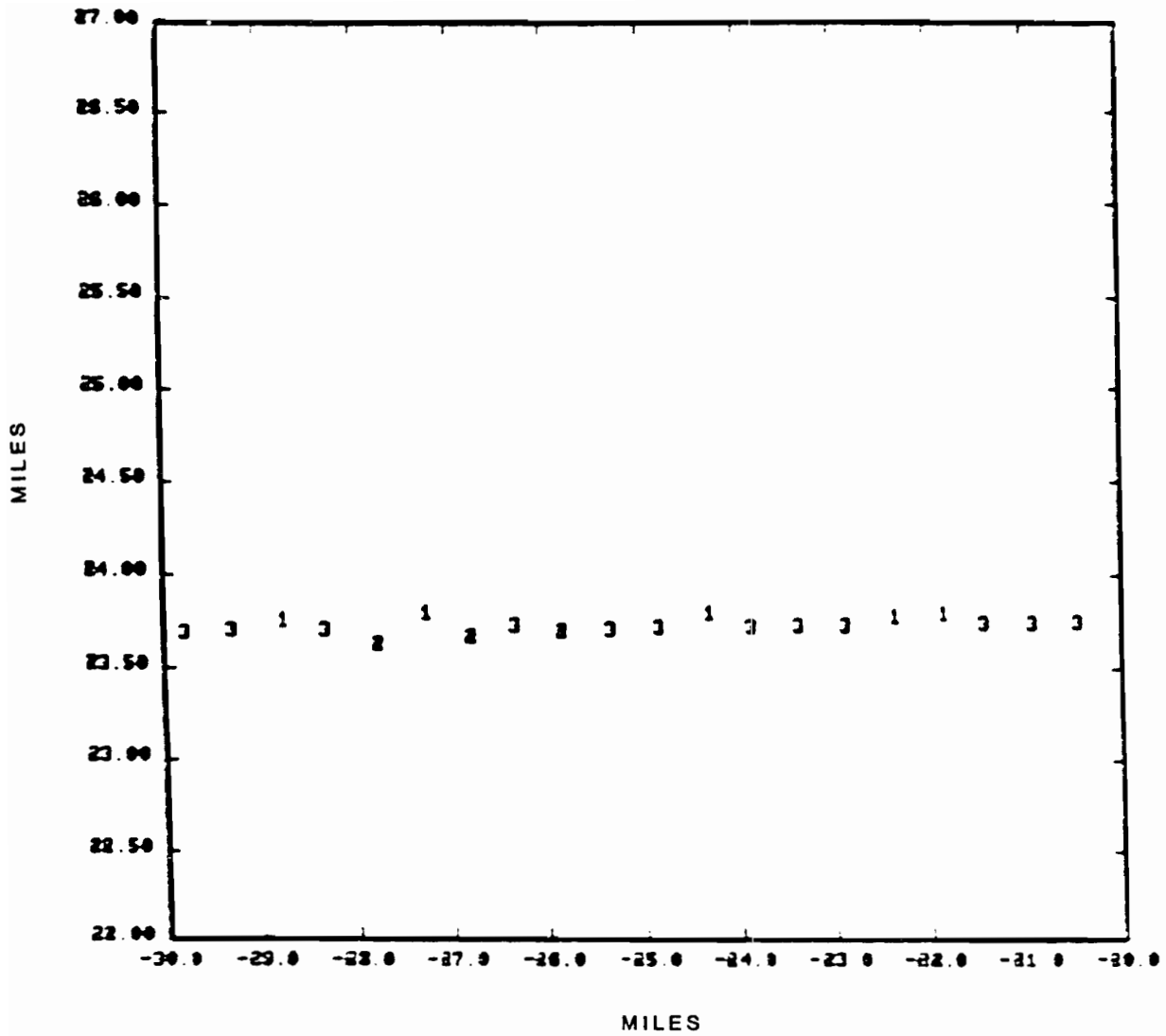
9.5 Transponder Bias Estimation

The major aircraft bias, namely the transponder turnaround error (Δ), can be handled in two different ways by the extended flat earth approach. The first is to compute Δ in real-time using the sensor measurements, while the second is to ignore Δ by treating it as an unknown bias. Only the first approach was applicable to the spherical multilateration method; ignoring Δ led to azimuth errors.

TABLE 9-1.
SYSTEM COMPARISONS

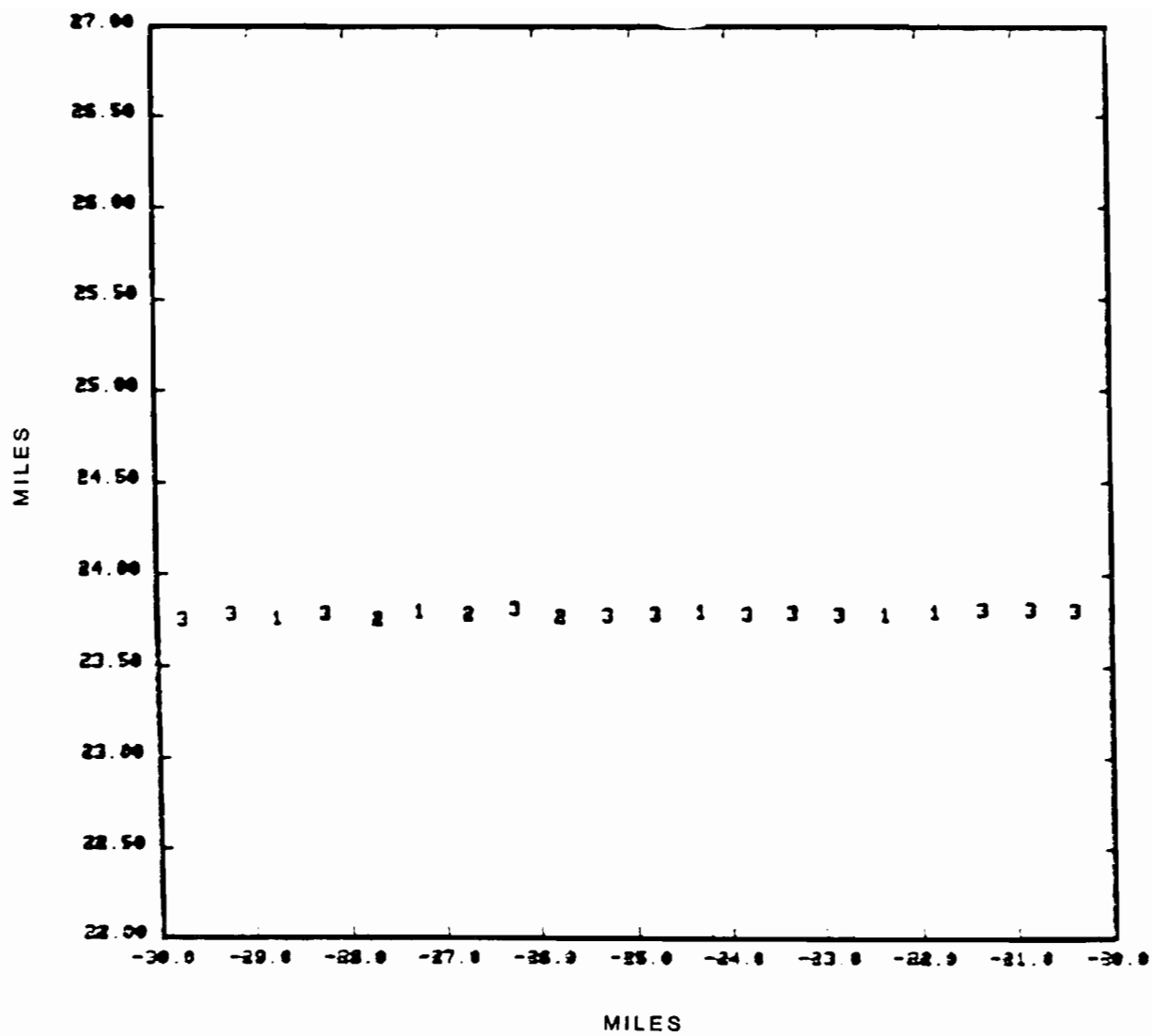
Statistic	Live Traffic System		Simulated Traffic System	
	Spherical Earth	Extended Flat Earth	Spherical Earth	Extended Flat Earth
Average Azimuth Error	.179°	.003°	.018°	.056° (see note)
Azimuth Standard Deviation	.125°	.069°	.123°	.044°
Average Heading Error	6.91°	5.66°	7.57°	5.26°
Average Velocity Error	20.2 knots	18.2 knots	30.9 knots	19.4 knots

Note: Simulated System had θ_1 bias of .057°



NOTATION: SEE FIG. 2-6.

Fig. 9-7. Multilateration data jumps.



NOTATION: SEE FIG. 2-6.

Fig. 9-8. Flat earth improved data.

If the spherical-equivalent flat earth model is being employed, Δ can be computed. For this model, Δ will act as a bias, moving the apparent position of the secondary sensor. Thus we can write:

$$D_{\text{scan}} = D_{\text{no bias}} + \text{term due to } \Delta$$

or as shown in Fig. 9-9:

$$\begin{aligned} D_{\text{scan}} &\approx \sqrt{(\rho_1 + \Delta)^2 - z_1^2} \cos(\theta_1 - \psi_{12}) + \sqrt{(\rho_2 + \Delta)^2 - z_2^2} \cos(\theta_2 - \psi_{21}) \\ &\approx \sqrt{\rho_1^2 - z_1^2} \sqrt{1 + \frac{2\Delta\rho_1}{\rho_1^2 - z_1^2}} \cos(\theta_1 - \psi_{12}) + \sqrt{\rho_2^2 - z_2^2} \sqrt{1 + \frac{2\Delta\rho_2}{\rho_2^2 - z_2^2}} \cos(\theta_2 - \psi_{21}) \\ &\approx D + \Delta \left[\frac{\rho_1}{\rho_{1g}} \cos(\theta_1 - \psi_{12}) + \frac{\rho_2}{\rho_{2g}} \cos(\theta_2 - \psi_{21}) \right] \end{aligned}$$

where

$$D = \rho_{1g} \cos(\theta_1 - \psi_{12}) + \rho_{2g} \cos(\theta_2 - \psi_{21}) = \text{normal unbiased } D$$

Thus finally:

$$\Delta = \frac{D_{\text{scan}} - D}{\frac{\rho_1}{\rho_{1g}} \cos(\theta_1 - \psi_{12}) + \frac{\rho_2}{\rho_{2g}} \cos(\theta_2 - \psi_{21})} \quad (9-12)$$

Although this formulation is not exact and uses both sensors' azimuth values, it has actually been found to produce no more scan-to-scan variation than the spherical earth formulation. This conclusion is supported by the Δ estimates shown in Fig. 9-10, in which the same input data used for Fig. 6-14 was employed. As seen, the data spread is very similar for the two methods. However, as seen above, the flat earth approach requires much less computation.

As always, the per-scan values of Δ must be averaged over many scans to produce a reasonable estimate of the true transponder bias error. This average value is then used in the normal position determination formulas (Fig. 9-6) as a correction to the ground ranges:

$$\rho_{1g} = \sqrt{(\rho_1 - \Delta)^2 - z_1^2}$$

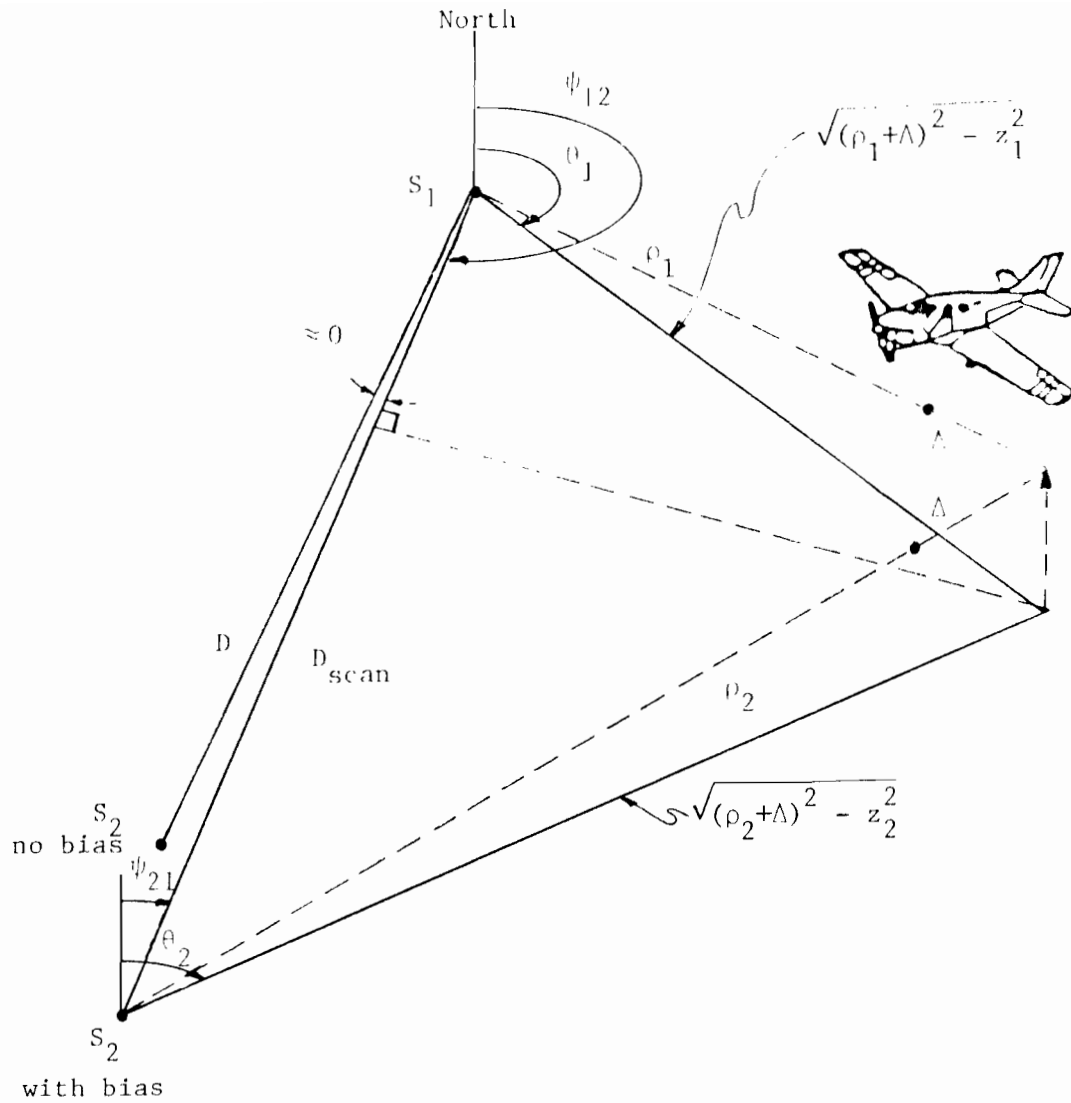


Fig. 9-9. Flat earth Δ computation.

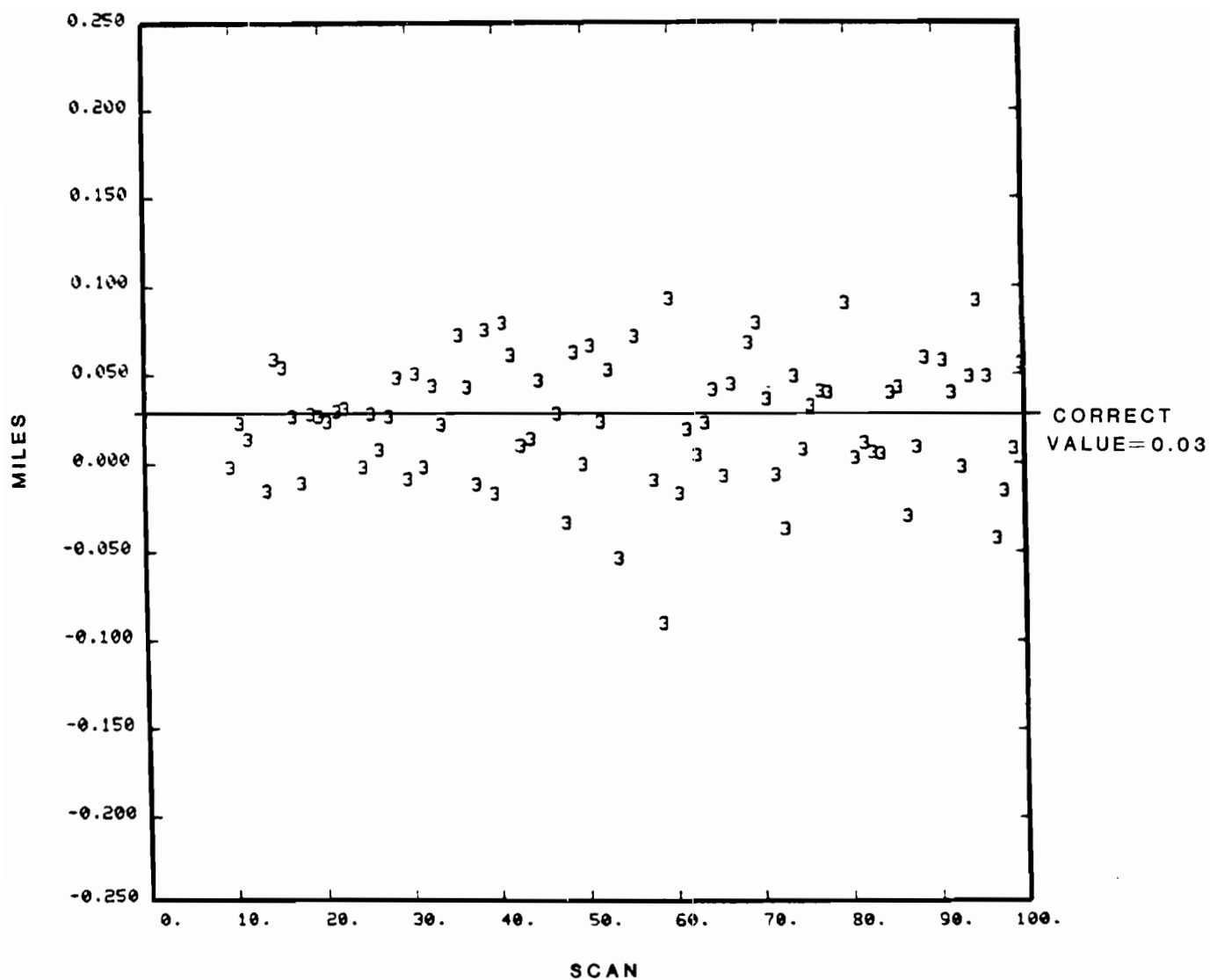


Fig. 9-10. Flat earth Δ estimates.

Alternatively, the transponder bias can be handled implicitly, without extra computation, by the extended flat earth model. Since this bias does not affect the azimuth reported by sensor 1, and since the model maintains the single sensor accuracy, this approach should provide both accuracy and consistency of reported azimuth data. The range data, however, will remain in error by the small amount of the bias. Fortunately for conflict alert algorithms, which depend on heading information, the heading accuracy is virtually unaffected by a transponder bias. This assertion is demonstrated in Appendix B. Thus it would appear that the extended flat earth approach, certainly the easiest method of dealing with the transponder bias, may also be the best way to deal with Δ among the techniques investigated.

The various approaches to dealing with the transponder bias were tested on the standard 24 simulated aircraft trajectories, all assuming a bias value of $\Delta = 0.03$ miles. Two cases were considered, one with no other system biases, and the other with typical sensor biases. The results, reported in Table 9-2, provide the average azimuth error produced by each approach. Only scans on which all sensors data was available, and thus on which a netting azimuth estimate was made, were considered. Note that the statistic is a combination of the mean and standard deviation of the errors, and thus differs from previous tables. This was done to provide a single number for comparison of methods.

The results of this table can be summarized as follows:

1. ignoring Δ leads to significant bilateration azimuth errors for both two and three sensor spherical earth models, whether or not other biases exist
2. estimating Δ , even approximately, improves the performance of these models, especially when other biases are present
3. using three sensors provides no more accurate performance
4. using the extended flat earth model, and not being concerned with Δ , provides, with less computation, nearly as accurate an azimuth determination as any estimation procedure when no other biases exist, and becomes the best approach when they do

9.6 Altitude Estimation

The absence of a reported altitude can also be handled in two different ways by the extended flat earth approach. First, a formula can be developed to estimate in real time the true aircraft altitude. Second, the altitude can be assumed to be any desired value, and the error from the value treated as

TABLE 9-2
EFFECTS OF Δ BIAS

		Transponder Delay (Δ) Assumption	
Sensor System	Biases (other than Δ)	Δ Ignored ($\Delta = 0$ assumed)	Δ Computed from Data
2 Sensors, Spherical Earth	No	.049°	.020°
	Yes	.090°	.029°
3 Sensors, Spherical Earth	No	.050°	.018°
	Yes	.089°	.028°
Extended Flat Earth	No	.025°	.018°
	Yes	.025°	.034°

Notes:

1. Statistic is average azimuth error = $-\frac{1}{n} \sum_{i=1}^n \theta_i - \theta_{\text{actual}}$
2. $\Delta = 0.03$ nautical miles

just another unknown system bias. Since most aircraft without encoding altimeters fly at low levels, a value of 0.5 miles (3000 feet) will be employed when this second approach is considered.

Formulas can be written that appear to permit the flat earth model to be used to estimate h . However, since the equivalence theorem depends upon h , using the model for this purpose introduces serious questions of model validity. Of the various equations for h that can be employed, the simplest is:

$$\sqrt{\rho_1^2 - z_1^2} \sin(\theta_1 - \psi_{12}) = \sqrt{\rho_2^2 - z_2^2} \sin(\theta_2 - \psi_{21}) \quad (9-13)$$

where the z_i 's are given in terms h by (1-1). Testing of this relationship, plus other candidates, has failed to yield one that possesses acceptable performance. None, in particular, can match the ones presented in Chapter 8.

However, considering the unknown altitude to be a bias, instead of trying to compute it, permits the extended flat earth model to be used directly. Since the effect of an altitude error, as shown earlier, is similar to that of a transponder bias, once again this approach yields azimuth and heading accuracy (except at very short ranges), although range data, and hence position, will be slightly in error.

The standard 24 simulated trajectories were used to test the azimuth accuracy of these various approaches to the unknown altitude problem. However, only scans on which the aircraft altitude was 3 miles (18,000 feet) or less were used in the results. Higher flying aircraft are virtually always altimeter equipped. The results presented in Tables 8-1 through 8-7 showed how accurate was the extended flat earth approach. Further data, showing a comparison of the various models and approaches, is shown in Table 9-3. These results provide the average azimuth error for each model for three different altitude assumptions: altitude known, altitude assumed to be 0.5 miles, and altitude computed from the data. The error statistic is the same as that defined in the last section. The table also presents the average altitude error for each approach for an aircraft near the upper flight limit.

These results serve as the basis for the following conclusions:

1. assuming an altitude of 0.5 miles leads to serious azimuth errors for higher flying aircraft for both the two and three sensor spherical earth models, while not compromising the extended flat earth model
2. estimating h works quite well for both spherical earth models, but poorly for the extended flat earth model
3. three sensor estimates of altitude are significantly superior to those using two sensors, while two sensor extended flat earth estimates are much poorer than those for the corresponding spherical earth case

TABLE 9-3
EFFECTS OF UNKNOWN h.

Sensor System	Aircraft Altitude (h) Assumption		
	h known	h assumed to be 0.5 nm	h Computed from Data
2 Sensors, Spherical Earth	.013°	.145°	.024°
	0.0 nm	2.0 nm	0.95 nm
3 Sensors, Spherical Earth	.015°	.142°	.018°
	0.0 nm	2.0 nm	0.19 nm
Extended Flat Earth	.024°	.026°	.045°
	0.0 nm	2.0 nm	1.39 nm

Notes:

1. Top statistic is average azimuth error = $-\frac{1}{n} \sum_{i=1}^n \theta_i - \theta_{\text{actual}}$
2. Bottom statistic is average altitude error at h=2.5 miles
3. h varies from 0 to 3 nautical miles.

4. using the extended flat earth model, and assuming the true altitude to be a bias from 0.5 miles, provides azimuth as accurate as the h estimated spherical earth cases, with much less computation, even for aircraft flying at considerably higher levels.

As shown in Chapter 8, adding typical system biases to the simulation tends to degrade the altitude estimation results and make the extended flat earth result appear comparatively better.

Of course, conflict alert algorithms require knowledge of aircraft altitude for detecting potential aircraft conflicts. Any estimate of altitude, however biased, from a beacon transponder or height finder radar will be sufficiently accurate for this application. If no altitude information exists, one of the estimation algorithms presented in the last chapter must be employed. The aircraft azimuth, in this case, would still be most accurately obtained via the extended flat earth approach.

10.0 PREDICTION FILTERS

The preceding chapters have presented algorithms for improving the accuracy of surveillance reports. These superior quality reports should produce more accurate predictions of future aircraft position when input to smoothing filters. Although the option of producing one netting report per scan as occurs with single sensor surveillance will allow the same tracker to be employed, the additional options open to the system suggest that a different tracker may produce superior overall performance. Also, the different character of netted data, particularly when system or aircraft biases exist and successive reports can arise from different sensor combinations, can tend to alter the relative performance of competing trackers.

This chapter presents a number of possible smoothing algorithms for raw and netted data, both for single and multiple reports per scan. The range of complexity extends over a wide spectrum. By testing the prediction ability of each tracker with each class of inputs, it is hoped that the most cost effective combination of netting and smoothing algorithms can be produced. (Note that overall performance, and not intermediate optimization, is the goal). Although test data is presented at the end of the chapter, this study was not yet completed when the netting project ended.

When multilateration reports are used as tracker inputs, the reports are by construction already in primary sensor coordinates. However, when raw sensor reports are input, those from secondary sensors must first be coordinate converted:

$$\rho_2, \theta_2 \rightarrow \rho_1, \theta_1$$

The mathematics of this process is presented in the first half of Appendix A.

A study of trackers is necessary even though the Kalman filter is known to be optimum, because of the following two points. First, the Kalman filter is costly in time and storage. If a simpler tracker performs nearly as well with the superior netting inputs, it should be preferred. Second, two major characteristics of netted aircraft data violate standard Kalman filter assumptions:

1. aircraft turn and maneuver
2. the input data contains biases that differ from one report to another.

Thus, the best filter to use in this application is not necessarily a Kalman filter.

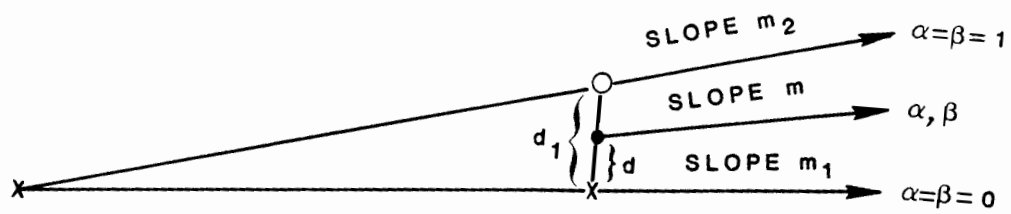
10.1 Alpha-Beta Filters

The simplest smoothing filters are those of the α, β class. The equations that define their operation are (Fig. 10-1):

$$\rho_s = \rho_p + \alpha (\rho_m - \rho_p)$$

$$\theta_s = \theta_p + \alpha (\theta_m - \theta_p)$$

X = PREDICTED
O = MEASURED



$$d = \alpha d_1$$

$$m = m_1 + \beta (m_2 - m_1)$$

Fig. 10-1. α, β filter.

$$\dot{\rho}_s = \dot{\rho}_p + \beta/T (\rho_m - \rho_p) \quad (10-1)$$

$$\dot{\theta}_s = \dot{\theta}_p + \beta/T (\theta_m - \theta_p)$$

where s, p, and m refer to smoothed, predicted and measured values respectively, and T is the time since the last update. The predicted values are given by:

$$\begin{aligned} \rho_p &= \rho_s + \dot{\rho}_s T \\ \theta_p &= \theta_s + \dot{\theta}_s T \\ \dot{\rho}_p &= \dot{\rho}_s \\ \dot{\theta}_p &= \dot{\theta}_s \end{aligned} \quad (10-2)$$

For single report per scan inputs, T will be essentially constant at T_{scan} ; for multiple report input streams, it will vary from 0 to T_{scan} .

The values of α and β can vary between 0 and 1. For single report per scan applications, the typical rules for selecting the proper values are:

1. increase α and β for more responsive tracking, decrease for smoother tracking
2. increase α and β as the track becomes firmer, decrease when uncertainty is present.

Various guidelines also exist for relating the α and β settings. The simplest is to set them equal, while one suggested by [8], which studies α, ϕ filters in detail, is:

$$\beta = \frac{\alpha^2}{2-\alpha} \quad (10-3)$$

The equality rule was used in this study, with both α and β set to 0.8.

When reports can arrive at the filter with varying inter-arrival times, as with the multiple report per scan applications, additional rules are required for setting α and β . As seen from (10-1), as T decreases, the velocity correction grows rapidly. Thus small measurement noise errors are greatly magnified when data points arrive closely spaced in time. For this reason, the values of α and β must satisfy:

$$\alpha, \beta = \begin{cases} 0 & T=0 \\ \alpha_0, \beta_0 & T=T_{\text{scan}} \end{cases} \quad (10-4)$$

where α_0 and β_0 are the settings selected by the above guidelines. The formula suggested by the above reference to meet this requirement is:

$$\alpha = 1 - \exp \left[-\frac{T}{T_{\text{scan}}} \log(1 - \alpha_0) \right] \quad (10-5)$$

This formula is employed in this study for both α and β .

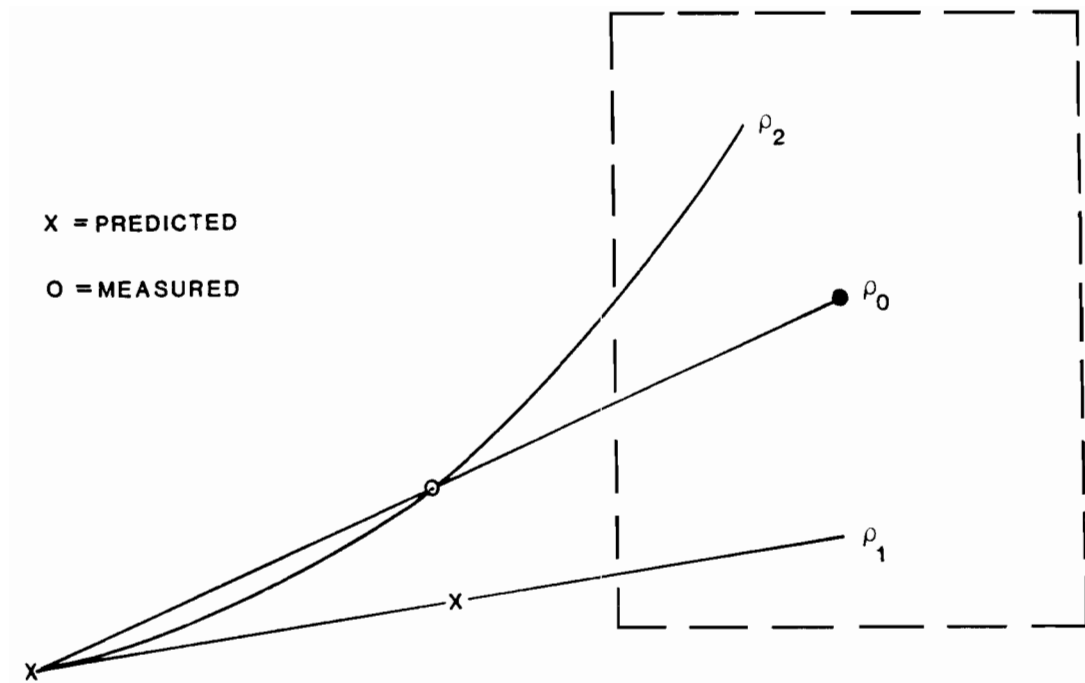
A special degenerate form of the α, β filter is the two-point interpolator. In this "smoother", the measured values are taken at face value, and the predicted track is simply a line drawn through the last two data points. This tracker was used in this study as a baseline performance indicator. Thus, the improvement in prediction accuracy of any tracker above that of the two-point interpolator is the payoff of the tracker algorithm.

Two points must be made about the two point-interpolator. First, if netting could supply error-free surveillance reports, this tracker would be as accurate as the Kalman filter. Second, the two-point interpolator is nearly optimum for correlation algorithms, in which the goal is to minimize the target search box required around the predicted aircraft position. As shown in Fig. 10-2, by using a linear projection through a suspect data point, the maximum error in prediction is minimized.

10.2 Curve Fitting Filters

When several surveillance reports are input per scan, the algorithms of the previous section include them one at a time into the predicted track. It is also possible to process such reports in a batch manner, similar to the methods that would be used by a human estimation of the data. As shown in Fig. 10-3, the most natural method of generating a track when viewing several points over time is to fit a least-squares curve through them.

The first curve fitting filter considered employs a straight line fit of the form shown in the figure. The earliest point in time, namely the one generated by last scan's curve fitting, serves as the anchor of the line. This point is used directly, rather than being subject to curve fitting, because of its expected small error variance, being produced by a previous curve fit. The new data points, on the other hand, all have full measurement errors connected with them.

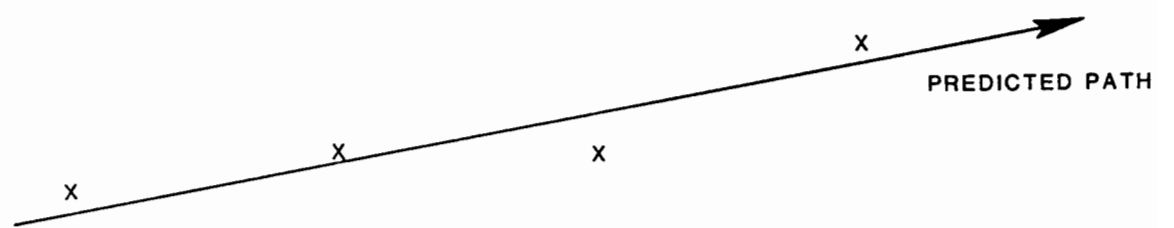


ρ_0 : PREDICTED NEXT SCAN POSITION, USING 2-POINT INTERPOLATOR

ρ_1 : TRUE POSITION IF REPORT WERE EXTRANEIOUS, AND REAL REPORT MISSING

ρ_2 : TRUE POSITION IF REPORT WERE CORRECT, AND TARGET IS TURNING

Fig. 10-2. Two-point interpolator search box.



X=MEASURED POINT

Fig. 10-3. Linear curve fitting.

The slopes of the least squares straight line are found as the solutions to the following problem:

Find m_ρ and m_θ

Such that

$$\sum_{i=1}^p (\hat{\rho}_i - m_\rho \hat{t}_i)^2 \text{ is minimized} \quad (10-6)$$

$$\sum_{i=1}^p (\hat{\theta}_i - m_\theta \hat{t}_i)^2 \text{ is minimized} \quad (10-7)$$

where

$$\hat{\rho}_i = \rho_i - \rho_0 \quad i=1, \text{ number of points } p$$

$$\hat{\theta}_i = \theta_i - \theta_0 \quad i=1, p$$

$$\hat{t}_i = t_i - t_0 \quad i=1, p$$

ρ_0, θ_0, t_0 are the range, azimuth, and time of the anchor point.

Differentiating (10-6) with respect to m_ρ yields the minimum value slope as:

$$2 \sum_{i=1}^p (\hat{\rho}_i - m_\rho \hat{t}_i) (-\hat{t}_i) = 0$$

$$m_\rho = \frac{\sum_{i=1}^p \hat{\rho}_i \hat{t}_i}{\sum_{i=1}^p \hat{t}_i^2} \quad (10-8)$$

Similarly,

$$m_\theta = \frac{\sum_{i=1}^p \hat{\theta}_i \hat{t}_i}{\sum_{i=1}^p \hat{t}_i^2} \quad (10-9)$$

The predicted position and velocity values for the track at the time of the current scan are then given by:

$$\begin{aligned}\dot{\rho} &= m_p \\ \dot{\theta} &= m_\theta \\ \rho &= \rho_0 + \dot{\rho} T_{\text{scan}} \\ \theta &= \theta_0 + \dot{\theta} T_{\text{scan}}\end{aligned}\tag{10-10}$$

These values of ρ and θ are then the anchor values for the next scan curve fit.

This algorithm is slightly modified when only one or two surveillance reports are received in the current scan's batch. Since so few points make the curve fit subject to error, an α , β smoothing factor is introduced as follows:

For one report input:

$$\dot{\rho}_{\text{new}} = \dot{\rho}_{\text{old}} + \beta_0(m_p - \dot{\rho}_{\text{old}})\tag{10-11}$$

$$\dot{\theta}_{\text{new}} = \dot{\theta}_{\text{old}} + \beta_0(m_\theta - \dot{\theta}_{\text{old}})$$

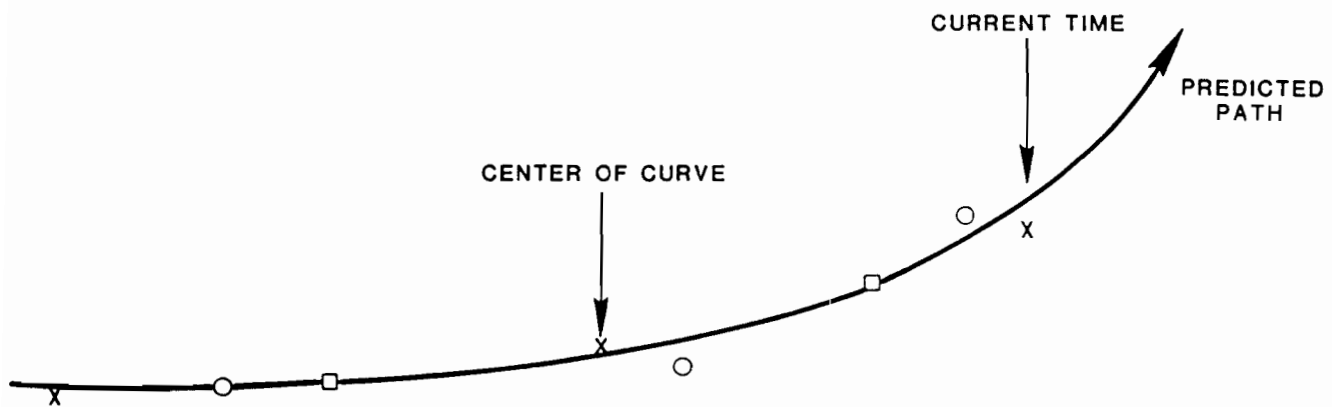
For two reports input:

$$\begin{aligned}\dot{\rho}_{\text{new}} &= \dot{\rho}_{\text{old}} + .5(1+\beta_0)(m_p - \dot{\rho}_{\text{old}}) \\ \dot{\theta}_{\text{new}} &= \dot{\theta}_{\text{old}} + .5(1+\beta_0)(m_\theta - \dot{\theta}_{\text{old}})\end{aligned}\tag{10-12}$$

where β_0 is determined as in the previous section. Note that if a one report per scan scheme uses this filter, it reduces to a simple α , β filter.

The second curve fitting approach that was considered in the study is more ambitious. In particular, it uses a quadratic fit, and attempts to predict both the current heading and the turn rate (if any) of the aircraft. To provide a reasonable chance of success, two full scans of surveillance reports, plus one additional primary sensor report, are used in the curve fitting formula. Thus, as shown in Fig. 10-4, the center of the curve is one scan behind real time.

The method for producing a quadratic curve fit is well documented. For details and a reference, the reader is referred to [9]. Let the solution variables be:



- X = PRIMARY SENSOR
- O = SECONDARY SENSOR
- = TERTIARY SENSOR

Fig. 10-4. Quadratic curve fitting.

ρ = range at time $T_{\text{scan}-1}$

θ = azimuth at time $T_{\text{scan}-1}$

$\dot{\rho}$ = ρ slope at time $T_{\text{scan}-1}$

$\ddot{\rho}$ = $\dot{\rho}$ slope at time $T_{\text{scan}-1}$

$\dot{\theta}$ = θ slope at time $T_{\text{scan}-1}$

$\ddot{\theta}$ = $\dot{\theta}$ slope at time $T_{\text{scan}-1}$

then the problem is to minimize the functions:

$$\sum_{i=1}^p (\rho_i - \rho - \dot{\rho} t_i - 1/2 \ddot{\rho} t_i^2)$$

$$\sum_{i=1}^p (\theta_i - \theta - \dot{\theta} t_i - 1/2 \ddot{\theta} t_i^2)$$

where $t_i = t_i - T_{\text{scan}-1}$. Once the solution is known, the predicted position at any time in the future (including the current scan) is given by:

$$\rho(t) = \rho + \dot{\rho}(t - T_{\text{scan}-1}) + 1/2 \ddot{\rho} (t - T_{\text{scan}-1})^2 \quad (10-13)$$

$$\theta(t) = \theta + \dot{\theta}(t - T_{\text{scan}-1}) + 1/2 \ddot{\theta} (t - T_{\text{scan}-1})^2$$

After each scan's curve fit procedure is completed, the reports older than $T_{\text{scan}-1}$ are dropped and the remaining reports saved for the next scan's fit.

Again, this approach is modified if too few reports are available to be curve fit. With only one report, no curve fit is attempted, and the tracker coasts. With two or three reports, only a linear curve fit is produced.

Thus $\ddot{\rho} = \ddot{\theta} = 0$.

Also, experimental testing has shown that improved accuracy is obtained if the accelerations are both set to 0 whenever a turn rate of less than $1^\circ/\text{sec}$ is predicted. The turn rate for this filter (in radians/sec) is given by:

$$TR = \dot{\theta} + \frac{\dot{\rho}^2 \ddot{\theta} + \rho \dot{\rho} \ddot{\theta} - \rho \dot{\rho} \ddot{\theta}}{\dot{\rho}^2 + \rho^2 \dot{\theta}^2}$$

10.3 Standard Kalman Filter

The Kalman filter is a recursive filter which minimizes the prediction error for any aircraft which obeys the assumed equations of motion. There is no single Kalman filter. Rather, there is a different one for every different set of motion equations. Should an aircraft violate the motion assumptions, such as by turning, the Kalman filter can have large prediction errors. The usual manner of handling such problems is to include a turn detector, and modify the filter equations in some heuristic manner that permits the turn to be followed, although with severely degraded tracking. This section describes the standard Kalman filter used for aircraft. The next two sections present new Kalman filters that attempt to maintain prediction accuracy during turns to the same degree as accuracy during straight flight.

To aid in understanding, it should be noted that a Kalman filter is in reality an α, β type filter. The main differences are that position and velocity are coupled, and that the gain values (generalized α and β) are computed and changed each and every update. The following description of the Kalman filter is taken from [10] and is included to provide a concise description of the approach. The turn detection algorithm and resulting actions are also provided.

"The state equation in xy coordinates, which in our case represents the equation of motion, is:

$$X(t + 1) = \phi(t)X(t) + \Gamma(t)A(t), \quad (10-14)$$

$$\text{where } X(t) = \begin{bmatrix} x(t) \\ \dot{x}(t) \\ y(t) \\ \dot{y}(t) \end{bmatrix}, \phi(t) = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix}, \Gamma(t) = \begin{bmatrix} 1/2 & T^2 & 0 \\ T & 0 \\ 0 & 1/2 & T^2 \\ 0 & T \end{bmatrix}, \text{ and } A(t) = \begin{bmatrix} a_x(t) \\ a_y(t) \end{bmatrix}$$

with $X(t)$ being the state vector at time t consisting of position and velocity

components $x(t), \dot{x}(t), y(t),$ and $\dot{y}(t)$, $t + 1$ being the next observation time, T being the time between observations, and $a_x(t)$ and $a_y(t)$ being random accelerations whose covariance matrix is $Q(t)$. The observation equation is

$$Y(t) = M(t)X(t) + V(t), \quad (10-15)$$

$$\text{where } Y(t) = \begin{bmatrix} x_m(t) \\ y_m(t) \end{bmatrix}, M(t) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \text{ and } V(t) = \begin{bmatrix} u_x(t) \\ u_y(t) \end{bmatrix},$$

with $Y(t)$ being the measurement at time t consisting of positions $x_m(t)$ and $y_m(t)$ and $V(t)$ being zero mean noise whose covariance matrix is $R(t)$.

The problem is solved recursively by first assuming the problem is solved at time $t-1$. Specifically it is assumed that the best estimate $\hat{X}(t-1|t-1)$ at time $t-1$ and its error covariance matrix $P(t-1|t-1)$ are known, where the circumflex ^ signifies an estimate and $\hat{X}(t|s)$ signifies that $X(t)$ is being estimated with observations up to $Y(s)$. The six steps involved in the recursive algorithm are as follows:

Step 1. Calculate one step prediction,

$$\hat{X}(t|t-1) = \phi(t-1)\hat{X}(t-1|t-1); \quad (10-16)$$

Step 2. Calculate the covariance matrix for one step prediction,

$$P(t|t-1) = \phi(t-1)P(t-1|t-1)\tilde{\phi}(t-1) + \Gamma(t-1)Q(t-1)\tilde{\Gamma}(t-1); \quad (10-17)$$

Step 3. Calculate the prediction observation,

$$\hat{Y}(t|t-1) = M(t)\hat{X}(t|t-1); \quad (10-18)$$

Step 4. Calculate the filter gain,

$$\Delta(t) = P(t|t-1)\tilde{M}(t)[M(t)P(t|t-1)\tilde{M}(t) + R(t)]^{-1}; \quad (10-19)$$

Step 5. Calculate a new smoothed estimate,

$$\hat{X}(t|t) = \hat{X}(t|t-1) + \Delta(t)[Y(t) - \hat{Y}(t|t-1)]; \quad (10-20)$$

Step. 6 Calculate a new covariance matrix,

$$P(t|t) = [I - \Delta(t)M(t)] P(t|t-1). \quad (10-21)$$

In summary, starting with an estimate $\hat{X}(t-1|t-1)$ and its covariance matrix $P(t-1|t-1)$, after receiving a new observation $Y(t)$ and calculating the six quantities in the recursive algorithm, a new estimate $\hat{X}(t|t)$ and its covariance matrix $P(t|t)$ are obtained.

For the Kalman filter in xy coordinates, the measurement covariance matrix $R(t)$ is a function of the radar-target geometry. Letting (at time t) ρ_t and θ_t be the range and azimuth of the target with respect to the radar (with the azimuth angle being measured counterclockwise from the x axis), the elements of the covariance matrix

$$R(t) = \begin{bmatrix} \sigma_x^2(t) & \sigma_{xy}(t) \\ a_{xy}(t) & \sigma_y^2(t) \end{bmatrix}, \quad (10-22)$$

are

$$\sigma_x^2(t) = \sigma_\rho^2 \cos^2 \theta_t + \rho_t^2 \sigma_\theta^2 \sin^2 \theta_t, \quad (10-23)$$

$$\sigma_y^2(t) = \sigma_\rho^2 \sin^2 \theta_t + \rho_t^2 \sigma_\theta^2 \cos^2 \theta_t, \quad (10-24)$$

and

$$\sigma_{xy}(t) = [\sigma_\rho^2 - \rho_t^2 \sigma_\theta^2] \sin \theta_t \cos \theta_t, \quad (10-25)$$

where σ_ρ^2 and σ_θ^2 are the variances of the range and azimuth measurement errors respectively.

The Kalman filter is the optimum filter as long as the target trajectory obeys the state equation (10-14), which describes a straight-line trajectory with random perturbations (the random perturbations bound the filter gains away from zero). However, when the target maneuvers, the maneuver must be detected and the error covariance matrix must be increased. In this study the error criterion is

$$E = [\hat{X}(t|t-1) - \hat{X}(t|t)]^T \tilde{M} [M P(t|t-1) \tilde{M}]^{-1} M [\hat{X}(t|t-1) - \hat{X}(t|t)] \\ + [Y(t) - M \hat{X}(t|t)]^T R(t)^{-1} [Y(t) - M \hat{X}(t|t)]. \quad (10-26)$$

This error is the squared Mahalanobis distance from the smooth position

$\hat{MX}(t|t)$ to the predicted position $\hat{MX}(t|t-1)$ plus the squared Mahalanobis

distance from the smooth position $\hat{MX}(t|t)$ to the measured position $Y(t)$. The Mahalanobis distance differs from the Euclidean distance by using a covariance-matrix kernel instead of an identity matrix.

When the error E is greater than a threshold (which in this study was set to $E = 16$, corresponding for example to covariance matrices that are diagonal and smooth coordinates that differ from the predicted and measured positions by twice the standard deviation), the error covariance matrix $P(t-1|t-1)$ is

increased and a new smooth position $\hat{MX}(t|t)$ is calculated. Increasing $P(t-1|t-1)$ causes the new position estimate $\hat{MX}(t|t)$ to be closer to the

measurement $Y(t)$ and further from the prediction $\hat{X}(t|t-1)$. Since

$P(t|t-1)$ increases when $P(t-1|t-1)$ increased, this increase in $P(t-1|t-1)$ will always cause E to decrease. This procedure is repeated until E is less than the threshold. Specifically terms P_{11} , P_{13} , P_{31} , and P_{33} are increased by $\sqrt{F}$; terms P_{12} , P_{14} , P_{21} , P_{23} , P_{32} , P_{34} , P_{40} , and P_{43} are increased by F ; and terms P_{22} , P_{24} , P_{42} , and P_{44} are increased by F^2 . (In this study $F = 1.5^n$, where n is the number of consecutive covariance matrix increases). The position covariance elements are not increased as much as the velocity elements because of coupling; that is, an uncertainty in predicted position is due not only to the uncertainty in the last position but also in the velocity. In a real system the track should also be bifurcated when a large error is encountered".

This presentation applies directly when raw sensor reports are input to the filter. When netted data is used as input, the measurement covariance matrix must be modified to account for the fact that two sensors are acting as data sources. The exact theoretical covariance matrix for multilateration is extremely complex. An approximately correct answer can be obtained by viewing the effect of multilateration to be that pictured earlier in Fig. 1-9. The joint error ellipse shown there is given simply by:

$$R_{\text{net}}(t) = [[R_{s1}(t)]^{-1} + [R_{s2}(t)]^{-1}]^{-1}$$

The rest of the Kalman filter is unchanged with multilateration inputs.

The filter initialization occurs after two inputs have been received, and is given by:

$$X(T) = \begin{bmatrix} x(T) \\ y(T) \\ (x(T)-x(0))/T \\ (y(T)-y(0))/T \end{bmatrix}$$

$$P(T) = \begin{bmatrix} \sigma_x^2(T) & \sigma_x^2(T)/T & \sigma_{xy}^2(T) & \sigma_{xy}^2(T)/T \\ \sigma_x^2(T)/T & 2\sigma_x^2(T)/T^2 & \sigma_{xy}^2(T)/T & 2\sigma_{xy}^2(T)/T^2 \\ \sigma_{xy}^2(T) & \sigma_{xy}^2(T)/T & \sigma_y^2(T) & \sigma_y^2(T)/T \\ \sigma_{xy}^2(T)/T & 2\sigma_{xy}^2(T)/T^2 & \sigma_y^2(T)/T & 2\sigma_y^2(T)/T^2 \end{bmatrix}$$

where 0 and T are the times of the two inputs. The derivations of these values are all straightforward. For example:

$$P_{12} = E [xx] = E \left[x(T) \frac{x(T)-x(0)}{T} \right]$$

$$= \sigma_x^2(T)/T$$

since measurements are independent, identically distributed random variables.

Finally, the one parameter in this filter, namely the magnitude of the random acceleration, must be set. This setting determines the responsiveness of the filter to turns. A value of 2 knots per second has been chosen for this study.

10.4 Heading Kalman Filter

The standard Kalman filter just described assumes a constant velocity aircraft motion. Thus it has trouble following turns and predicting heading during turns. A more general model of aircraft motion is one that assumes a constant turn rate. When this rate is zero, this model reduce to the previous one; when non-zero, the model will track aircraft performing smooth turns. As above, the formulation requires a turn detection mechanism to determine when turns begin and end. However, whereas the standard Kalman filter merely attempts to survive the turn, this model resets its turn rate estimate and should continue accurate tracking throughout the maneuver.

The equations of motion under a constant turn rate assumption are:

$$x(t+T) = x(t) + s(t) * \sin h(t) * T \quad (10-27)$$

$$y(t+T) = y(t) + s(t) * \cos h(t) * T \quad (10-28)$$

$$h(t+T) = h(t) + \dot{h}(t) T \quad (10-29)$$

$$\dot{h}(t+T) = \dot{h}(t) \quad (10-30)$$

where $h(t)$ and $s(t)$ are the heading and speed of the aircraft at time t . Unfortunately, these equations are non-linear. Thus no regular Kalman filter can be employed for their recursive solution. The next section describes an extended (non-linear) Kalman filter developed for this application, while this section develops a two-step, approximate, linear formulation.

The two equations (10-29) and (10-30) satisfy a linear Kalman filter model, with h instead of x, y being the coordinate. Thus the formulation of the previous section applies directly with

$$X(t) = \begin{bmatrix} h(t) \\ \dot{h}(t) \end{bmatrix} \quad \phi(t) = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \quad \Gamma(t) = \begin{bmatrix} 1/2T^2 \\ T \end{bmatrix}$$

$$M(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \text{ etc.}$$

Having only 2 states, the Kalman intermediate equations become considerably simplified. For example, the gain matrix $\Delta(t)$ reduces to:

$$\Delta(t) = \frac{1}{P_{11} + R(t)} \begin{bmatrix} P_{11} \\ P_{12} \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

where α and β have the same meaning as in an α, β filter.

This filter has but a single measurement variable, the heading h :

$$h = \tan^{-1} \left[\frac{x_2 - x_1}{y_2 - y_1} \right] \quad (10-31)$$

where the subscripts 2 and 1 refer respectively to the current and previous sensor reports. Since two real measurements are needed to define a single heading measurement, some complexity is introduced into the measurement side of the filter. First, the time at which the value of h applies is the midpoint of the two measurement times:

$$t_h = 1/2 (t_1 + t_2)$$

so that the filter always lags real time. Second, the heading variance σ_h^2 is somewhat more involved. Using the rule:

$$\sigma_h^2 (f(x_1, \dots, x_n)) = \sum_{i=1}^n \sum_{j=1}^n \frac{\partial f}{\partial x_i} \frac{\partial f}{\partial x_j} \sigma_{x_i x_j}^2 \quad (10-32)$$

combined with the definition (10-31) and the independence of measurements 1 and 2, the variance is given by:

$$\sigma_h^2 = \frac{(y_2 - y_1)^2 (\sigma_{x1}^2 + \sigma_{x2}^2) + (x_2 - x_1)^2 (\sigma_{y1}^2 + \sigma_{y2}^2) - 2(y_2 - y_1)(x_2 - x_1)(\sigma_{x1y1}^2 + \sigma_{x2y2}^2)}{[(y_2 - y_1)^2 + (x_2 - x_1)^2]^2}$$

where the σ_x and σ_y values are determined by (10-23) through (10-25).

The turn detector chosen for this tracker is the existence of two successive heading measurements that are both outside the expected range in the same direction. This is, a turn is declared whenever:

$$\begin{aligned} |\gamma_n| &> 2 \\ |\gamma_{n-1}| &> 2 \\ \gamma_n * \gamma_{n-1} &> 0 \end{aligned} \quad (10-33)$$

$$\text{where } \gamma = \frac{h_{\text{measured}} - h_{\text{predicted}}}{\sqrt{P_{11}}}$$

is the tracker heading standard deviation. This same criterion is used to detect the end of the turn, or the transition to a turn of a different turn rate. Whenever any of these events occurs, the tracker is restarted with:

$$h = h_{\text{measured}}$$

$$\dot{h} = \frac{h_{\text{measured},n} - h_{\text{measured},n-1}}{T}$$

$$P = \begin{vmatrix} 2 & 2 \\ \sigma_h & \sigma_h \\ 2 & 2 \\ \sigma_h & 2\sigma_h \end{vmatrix}$$

Whenever sensor or transponder biases exist in the system, a heading measurement using reports from two different sources could be significantly in error. Thus, heading estimates are only permitted from successive reports from the same source, as depicted in Fig. 10-5. As shown, these estimates can overlap each other in time. This presents no problem as long as the estimates arrive at the filter in time order. Occasionally the new heading estimate will be older than the previous one; this estimate is discarded. Appendix C, as discussed earlier, shows that heading accuracy is preserved under transponder biases when the same source is used for both points.

The next aircraft attribute required for this overall tracking system is its speed. Assuming that speed changes very slowly for an aircraft in flight, the following very simple moving average filter was chosen:

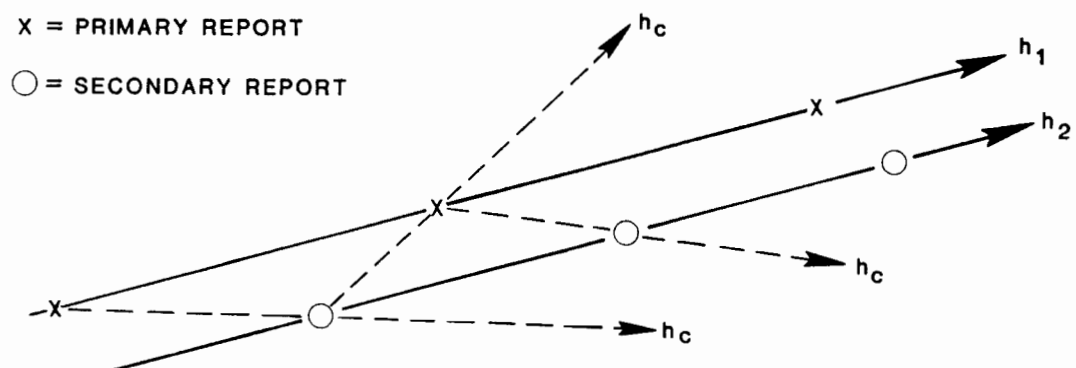
$$S_{\text{new}} = \frac{N-1}{N} S_{\text{old}} + \frac{1}{N} S_{\text{measured}} \quad N = \begin{cases} n & n \leq 5 \\ 5 & n > 5 \end{cases}$$

where n is the number of speed measurement samples that have been taken. Each such measurement is given by:

$$S_{\text{measured}} = \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{T} \frac{\dot{h}T/2}{\sin(\dot{h}T/2)} \quad (10-34)$$

where T is the interval between the first and second reports. Figure 10-6 illustrates how the second factor supplies a needed correction to the usual speed formula when an aircraft is in a turn. As with the heading, only successive reports from the same source are used in this calculation.

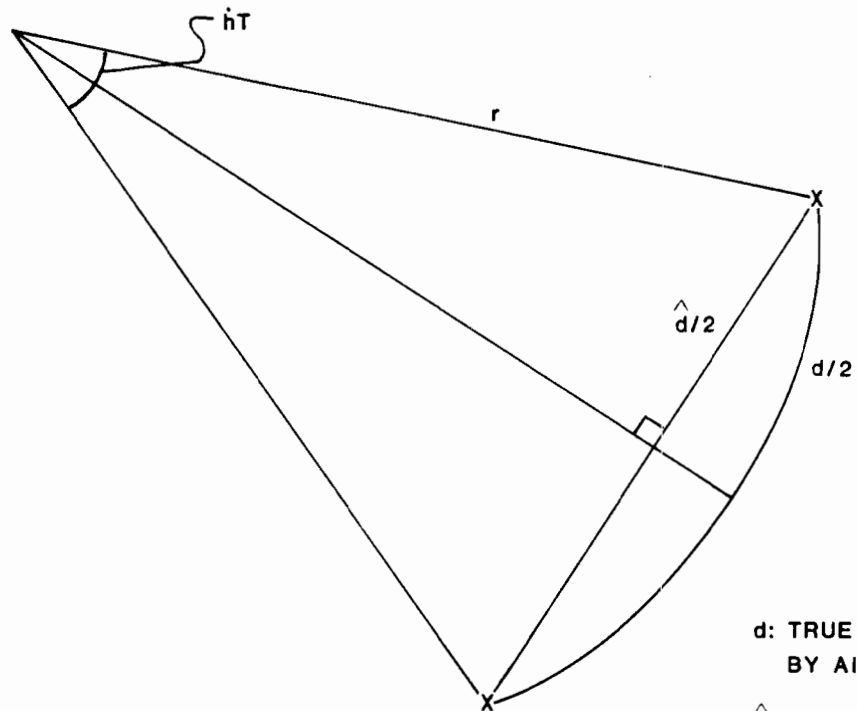
Once the aircraft heading and speed are known, equations (10-27) and (10-28) can be solved by a Kalman filter approach. The state vector consists of two components, x and y . The state update step becomes:



h_1 : PRIMARY HEADING ESTIMATE: ACCURATE
 h_2 : SECONDARY HEADING ESTIMATE: ACCURATE
 h_c : CROSS-TERM HEADING ESTIMATE: ERRONEOUS

Fig. 10-5. Heading estimate procedure.

$\dot{h}$ = HEADING RATE OF CHANGE
 T = TIME BETWEEN SCANS
 r = RADIUS OF TURN



d : TRUE DISTANCE TRAVELED
 BY AIRCRAFT

$\hat{d}$: APPARENT DISTANCE TRAVELED
 BY AIRCRAFT

$$d/2 = r \dot{h} T/2$$

$$\hat{d}/2 = r \sin \dot{h} T/2$$

$$\therefore d = \hat{d} \frac{\dot{h} T/2}{\sin \dot{h} T/2}$$

Fig. 10-6. Turn correction factor.

$$\begin{aligned}x(t+T) &= x(t) + \hat{s} T \sin[h + (t+T/2-\tau)\dot{h}] \\y(t+T) &= y(t) + \hat{s} T \cos[h + (t+T/2-\tau)\dot{h}]\end{aligned}\tag{10-35}$$

where h and $\dot{h}$ are the most recent values from the heading Kalman filter, valid at time τ , and $\hat{s}$ is the speed term modified to account for turns:

$$\hat{s} = s \frac{\sin(\dot{h}T/2)}{\dot{h}T/2}$$

The covariance matrix update uses the rule expressed in (10-32), yielding:

$$\Delta P_{11} = \left(\frac{\partial \dot{x}}{\partial h} \right)^2 \sigma_h^2 = \hat{s}^2 T^2 \cos^2(h) \sigma_h^2$$

$$\Delta P_{12} = \Delta P_{21} = \left(\frac{\partial \dot{x}}{\partial h} \right) \left(\frac{\partial \dot{y}}{\partial x} \right) \sigma_h^2 = -\hat{s}^2 T^2 \cos(h) \sin(h) \sigma_h^2$$

$$\Delta P_{22} = \left(\frac{\partial \dot{y}}{\partial h} \right)^2 \sigma_h^2 = -\hat{s}^2 T^2 \sin^2(h) \sigma_h^2$$

where

$$\hat{h} = h + (T/2-\tau)\dot{h}$$

is the average heading during the update. The remaining steps of the approach for x and y are identical to those of the standard Kalman filter (steps 3 through 6 of the previous section).

The approach presented in this section can be summarized as follows:

1. estimate the heading and turn rate of the aircraft using a 2-state Kalman filter
2. estimate the aircraft speed via a moving average
3. smooth the aircraft x , y position using a Kalman filter approach
4. predict future positions using the present position, heading, turn rate, and speed.

Although this method appears to be complex, it actually requires significantly less processing time and storage than the standard Kalman filter, mainly because of the fact that all matrices are 2x2, not 4x4.

While testing this method, one modification was found to improve performance: whenever the estimated turn rate is less than 1° per second, assume the aircraft is really flying straight. This assumption is used in both the position smoothing and future prediction steps. The results presented later use this modification, and assume a heading filter acceleration noise of 1/16° per second per second.

The study, design, and optimization of this tracker has not yet been completed. In particular, as shown later in the results, its x,y smoothing section tends to cause drift from the true aircraft position. Parameter adjustments, or even algorithm modifications, may be needed to remove this effect.

10.5 Extended Kalman Filter

The major problem with the previous approach is that the heading and position estimates are uncoupled. That is, first heading is estimated, then position is smoothed. This was forced by the desire for a linear solution method. This section considers the extended Kalman filter [11], an approach that permits non-linear state equations of motion.

Table 10-1, taken from the reference, presents the equations that define the extended Kalman filter. Let a 5-state system be employed: x, y, $\dot{h}$, h, and s. The equations of motion defining the system model thus become (refer to (10-27) through (10-30)):

$$\dot{\underline{X}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{h} \\ \ddot{h} \\ \dot{h} \\ \dot{s} \end{bmatrix} = \begin{bmatrix} s \sin h \\ s \cos h \\ \dot{h} \\ 0 \\ 0 \end{bmatrix} = \underline{f}$$

and the measurement model is:

$$\underline{h} = \begin{bmatrix} x \\ y \end{bmatrix}$$

From this, the two major matrices needed by the formulation are given by:

TABLE 10-1

SUMMARY OF CONTINUOUS DISCRETE EXTENDED KALMAN FILTER

System Model	$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t), t) + \underline{w}(t); \quad \underline{w}(t) \sim N(\underline{0}, Q(t))$
Measurement Model	$\underline{z}_k = \underline{h}_k(\underline{x}(t_k)) + \underline{v}_k; \quad k = 1, 2, \dots, \quad \underline{v}_k \sim N(\underline{0}, R_k)$
Initial Conditions	$\underline{x}(0) \sim N(\hat{\underline{x}}_0, P_0)$
Other Assumptions	$E[\underline{w}(t) \underline{v}_k^T] = 0$ for all k and all t
State Estimate Propagation	$\dot{\underline{x}}(t) = \underline{f}(\hat{\underline{x}}(t), t)$
Error Covariance Propagation	$\dot{P}(t) = F(\hat{\underline{x}}(t), t) P(t) + P(t) F^T(\hat{\underline{x}}(t), t) + Q(t)$
State Estimate Update	$\hat{\underline{x}}_k(+) = \hat{\underline{x}}_k(-) + K_k [\underline{z}_k - \underline{h}_k(\hat{\underline{x}}_k(-))]$
Error Covariance Update	$P_k(+) = [I - K_k H_k(\hat{\underline{x}}_k(-))] P_k(-)$
Gain Matrix	$K_k = P_k(-) H_k^T(\hat{\underline{x}}_k(-)) [H_k(\hat{\underline{x}}_k(-)) P_k(-) H_k^T(\hat{\underline{x}}_k(-)) + R_k]^{-1}$

$$F(\hat{\underline{x}}(t), t) = \left. \frac{\partial \underline{f}(\underline{x}(t), t)}{\partial \underline{x}(t)} \right|_{\underline{x}(t) = \hat{\underline{x}}(t)}$$

Definitions

$$H_k(\hat{\underline{x}}_k(-)) = \left. \frac{\partial \underline{h}_k(\underline{x}(t_k))}{\partial \underline{x}(t_k)} \right|_{\underline{x}(t_k) = \hat{\underline{x}}_k(-)}$$

$$F = \begin{vmatrix} 0 & 0 & s \cos h & 0 & \sin h \\ 0 & 0 & -s \sin h & 0 & \cos h \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

$$H = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{vmatrix}$$

With these definitions, the solution proceeds as shown in the table.

The initial covariance matrix is constructed by using the rule (10-32).

As an example of its application, $\sigma_{13}^2 (= \sigma_{xh}^2)$ is derived. The heading is:

$$h = \tan^{-1} \left[\frac{x_3 - x_2}{y_3 - y_2} \right]$$

where 3 is the most recent report of the three needed for this method. Then:

$$\sigma_{13}^2 = \frac{\partial}{\partial x_3} (x) \frac{\partial}{\partial x_3} (h) \sigma_x^2 + \frac{\partial}{\partial x_3} (x) \frac{\partial}{\partial y_3} (h) \sigma_{xy}^2$$

all other terms being zero, as reports are assumed to be independent. Thus finally:

$$\sigma_{13}^2 = \frac{(y_3 - y_2) \sigma_x^2 - (x_3 - x_2) \sigma_{xy}^2}{(y_3 - y_2)^2 + (x_3 - x_2)^2}$$

Other terms are derived in a similar manner.

This method was not fully developed when the study concluded, so no results can be shown. However, it is far more complex than any other method, both because a 5x5 matrix is involved and because incremental updates are required due to the inability to integrate the non-linear functions.

10.6 Tracker Performance Comparisons

The smoothing and prediction filters presented in this chapter were tested against various types of simulated data in an attempt to learn how each performs against the types of aircraft report environments that would be encountered in practice. The results are summarized in Tables 10-2 through 10-4. As seen, only a small subset of the combination of antenna scan rates, system biases, aircraft trajectories, netting timing, and netting algorithms are presented. Also, no fine tuning of tracking parameters was attempted. Thus the results discussed here concerning tracker comparisons should be taken as representative. More study would be required before a total netting system recommendation could be made.

The three tables differ with respect to the types of aircraft motion assumed:

Table 10-2: 24 real aircraft trajectories
Table 10-3: straight flight
Table 10-4: constant 2°/sec turning flight

For each table, a 4-second antenna was assumed, and type 1 or 2 netting timing was employed as appropriate for the number of output reports per scan desired. The other data components were each represented by two opposite options:

input data: raw or netted
reports/scan: 1 or 3
biases: none or transponder and system

This accounts for the 8 rows (2x2x2) in each table. The results presented in each box are the average position error in miles and heading error in degrees produced by the top-named tracker operating on the left-named input options.

The conclusions that can be supported by the results in these tables are as follows:

Two-point interpolator:

This "tracker" is not all that bad for single report per scan inputs. It handles straight and turning trajectories about equally well, thereby showing the minimax property that makes it suitable for Mode S correlation use. Note in particular that it does far better than the Kalman filter on turning tracks. It is totally unsuitable, however, on closely spaced reports as occur with three report per scan inputs.

α/β filter

This tracker is uniformly superior to the two-point interpolator, while requiring little additional storage or processing. It is also capable of processing several reports per scan.

TABLE 10-2

TRACKER PERFORMANCE, 24 AIRCRAFT TRAJECTORIES

		Tracker					
Input Data	Any Biases?	2-pt	α, β	Linear Fit	Quad Fit	Normal Kalman	Heading Kalman
Primary Sensor	N	.04 7.6	.04 5.6	.04 5.6	.04 4.2	.03 3.3	.08 4.5
	Y	.07 7.6	.07 5.6	.07 5.6	.07 4.2	.07 3.3	.10 4.5
Three Sensor	N	.06 32.7	.04 10.6	.04 7.5	.03 3.9	.03 7.1	.06 6.6
	Y	.11 47.0	.05 13.6	.05 8.2	.05 4.5	.06 11.6	.07 6.6
Extended Flat Earth (1/scan)	N	.03 5.1	.03 4.0	.03 4.0	.03 3.2	.03 2.9	.06 3.5
	Y	.07 5.1	.07 4.0	.07 4.0	.07 3.3	.07 2.9	.10 3.5
Multilat (3/scan)	N	.03 14.3	.03 4.9	.02 3.6	.02 2.6	.02 3.7	.18 3.7
	Y	.05 15.6	.04 5.6	.04 4.1	.04 2.9	.04 4.1	.18 4.3

Key:

.02
2.7

--- average position error, nm

--- average heading error, degrees

TABLE 10-3
TRACKER PERFORMANCE, STRAIGHT TRAJECTORY

		Tracker					
Input Data	Any Biases?	2-pt	α, β	Linear Fit	Quad Fit	Normal Kalman	Heading Kalman
Primary Sensor	N	.03 9.2	.03 7.5	.03 7.5	.03 4.7	.02 2.7	.18 4.7
	Y	.06 9.2	.06 7.5	.06 7.5	.06 4.7	.06 2.7	.19 4.7
Three Sensor	N	.06 29.3	.02 10.3	.02 7.5	.02 3.5	.02 1.5	.18 4.0
	Y	.11 46.3	.03 12.7	.03 8.8	.03 3.6	.03 6.1	.18 3.8
Extended Flat Earth (1/scan)	N	.03 7.1	.03 6.0	.03 6.0	.03 4.6	.03 3.1	.21 5.0
	Y	.07 7.1	.07 5.9	.07 5.9	.06 4.5	.07 3.1	.22 6.3
Multilat (3/scan)	N	.03 10.1	.02 5.6	.01 3.8	.01 2.3	.01 1.1	.15 2.6
	Y	.02 11.9	.02 6.3	.01 4.4	.02 2.5	.01 1.2	.12 4.0

Key:

.02	---	average position error, nm
2.7	---	average heading error, degrees

TABLE 10-4

TRACKER PERFORMANCE, TURNING TRAJECTORY

Input Data	Any Biases?	Tracker					
		2-pt	α, β	Linear Fit	Quad Fit	Normal Kalman	Heading Kalman
Primary Sensor	N	.02 6.6	.02 6.8	.02 6.8	.03 8.6	.04 12.1	.19 2.8
	Y	.05 6.6	.05 6.7	.05 6.7	.05 8.5	.06 12.1	.21 2.8
Three Sensor	N	.06 34.2	.02 8.3	.02 6.9	.02 7.9	.03 12.7	.08 5.4
	Y	.11 54.7	.03 15.9	.03 8.8	.03 7.8	.03 15.7	.08 5.4
Extended Flat Earth (1/scan)	N	.03 9.1	.03 7.2	.03 7.2	.03 10.3	.04 12.4	.17 5.7
	Y	.06 9.0	.06 7.1	.06 7.1	.06 10.3	.07 12.4	.19 9.4
Multilat (3/scan)	N	.03 22.2	.02 6.1	.02 6.6	.03 8.3	.04 11.8	.12 4.8
	Y	.03 25.7	.02 6.8	.02 6.8	.03 8.4	.04 11.7	.11 4.7

Key:

.02
2.7

--- average position error, nm
 --- average heading error, degrees

Linear fit tracker

This tracker is the extension of the α, β filter that applies specifically to multiple report per scan inputs. It does, as expected, provide superior performance in such cases.

Quadratic fit tracker

This tracker, for typical aircraft trajectories, is superior to the previous linear curve fit tracker. It is particularly improved for straight flight and when inter-sensor registration errors are present. Surprisingly, although this tracker computes an acceleration term whereas the linear one does not, it is inferior for turning flight. The reason for this seeming anomaly is that it is centered one scan behind real time. Thus a full scan projection is required for a current estimate, magnifying any heading error. Examination of other data shows that the heading error of this tracker, relative to that of the linear one, grows less rapidly with longer range projections; thus the acceleration term is a useful feature.

Kalman filter

As expected, this tracker is optimum for straight-flying aircraft in unbiased systems. However, it is the worst at following turns, and is affected by biases. In particular, it appears incapable of handling raw data from three sensors when registration errors exist. Various modifications would be required to upgrade its performance.

Heading Kalman filter

This tracker appears to be the best for processing reports from turning aircraft, thereby validating its design goal. It also has the property of providing reasonable to good performance for all types of trajectories, system biases, and data sources, and as such is a minimax type of tracker. Its one apparent problem is the tendency for positional drift. Since conflict alert algorithms are compromised most by large heading errors, this tracker has the potential of being the best.

10.7 Tracker Performance in Diffraction

To determine how well each type of tracker performed in diffraction zones, the 24 aircraft trajectory tests were repeated with simulated diffracted primary sensor azimuths. The method used, as described in section 2.4, assumed a 20-scan-wide diffraction zone reached after a 5-scan normal period that allowed each tracker to reach steady state. The σ_θ for the diffraction zone was set at 5 milliradians. The results, shown in Table 10-5, should be compared with the previous results reported in Table 10-2.

The first two rows, in which the primary diffracted reports were input to each tracker, clearly indicate that no tracker is capable of overcoming diffraction. The large azimuth variance is simply too great to be smoothed.

TABLE 10-5

TRACKER PERFORMANCE, 24 AIRCRAFT TRAJECTORIES, DIFFRACTION ZONES

		Tracker					
Input Data	Any Biases?	2-pt	α, β	Linear Fit	Quad Fit	Normal Kalman	Heading Kalman
Primary Sensor	N	.19 30.9	.17 24.9	.17 24.9	.17 17.4	.18 23.2	.23 31.4
	Y	.20 30.9	.18 24.8	.18 24.8	.18 17.4	.19 23.2	.23 31.3
Secondary Sensors	N	.21 28.9	.05 8.4	.05 6.2	.05 6.2	.06 8.4	.07 8.4
	Y	.29 36.4	.07 10.6	.07 6.9	.07 6.9	.09 11.2	.08 8.4
Extended Flat Earth (1/scan)	N	.03 3.6	.03 3.0	.03 3.0	.04 3.0	.03 2.8	.07 3.0
	Y	.06 3.6	.06 3.1	.06 3.1	.08 3.1	.06 2.8	.09 3.0
Multilat (3/scan)	N	.03 14.4	.03 4.8	.02 3.5	.02 2.6	.03 3.8	.13 3.8
	Y	.07 16.2	.05 5.4	.04 3.9	.04 2.8	.05 4.2	.13 4.1

Key:

.02
2.7

--- average position error, nm

--- average heading error, degrees

Thus, some form of netting approach is required to successfully track aircraft through a diffraction zone.

The second two rows assume that the two secondary sensor raw reports are input to each tracker; the primary, diffracted, reports are discarded. The results, for the most part, are slightly inferior to those of Table 10-2, although a few slight improvements in heading can be noted. These heading improvements are probably due to fewer data points producing fewer closely spaced data intervals. As explained in previous chapters, positional errors can be magnified into large velocity corrections at such intervals. More than likely, for enroute sensors, a heading degradation would occur in diffraction zones.

The tracker with the greatest performance degradation is the quadratic fit tracker. Since curve fitting counts on many input points for its accuracy, the loss of one-third of the reports not surprisingly has taken its toll. However, this tracker is still the best performer of those being considered for these input rows.

The extended flat earth input rows are essentially unaffected by the diffraction, for any of the trackers. This follows from the results of the last chapter, in which this netting algorithm was shown to perform well in diffraction.

Finally, the rows for multilateration, which doesn't use the primary sensor azimuth, are not surprisingly virtually unchanged for diffraction. The only difference caused by diffraction is that the tracker must now coast when no secondary report exists, whereas before it would use the raw primary report for its input.

10.8 Tracker Class Comparisons

At this time, it is possible to compare the three general classes of netting tracking techniques described in Chapter 6: high data rate, netted reports, and curve fitting.

For terminal 4-second sensors, the high data rate approach, of inputting many reports into standard sensor trackers, has yielded poor results. These reports tend to occur so close together in time that any small position errors are magnified into large velocity and heading errors. Thus this technique appears to have potential merit only for 10 and 12-second enroute sensors. An enroute study has not yet been undertaken.

The second technique, of jointly using measurements from two or three sensors to form netted tracker input reports, has yielded substantial performance improvements. By supplying more accurate position data to the trackers, their smoothing function is enhanced. The reports generated by the extended flat earth approach, when fed into a Kalman filter type tracker, have yielded excellent results.

Finally, the curve fitting approach has been found to be very competitive with the netted data input approach. The output of the curve fitting process, even without further smoothing, has yielded accurate position and heading results. Curve fitting versus netted input involves several storage and processing tradeoffs that are as yet not clearly defined.

11.0 NON-POSITIONAL DATA IMPROVEMENT ALGORITHMS

The last several chapters have concentrated on the improvement of positional accuracy for reports known to correspond to real aircraft. There are many uses for netting, however, in addition to this primary one. This chapter discusses the types of algorithms that might be employed in these cases. None have yet been implemented or tested on live data, so no performance details are available.

No sensor can maintain a perfect blip/scan ratio on all its aircraft. Netting can help maintain tracking continuity during such temporary loss of coverage. Moreover, even when a target report is present, some code or altitude information may be absent due to garble. Thus, having a second sensor to call on for help is often very useful for data continuity. In the extreme case, netting can provide a failsoft mechanism for some modes of sensor failure.

When a target report is missing, it is easy for a sensor to determine it needs help. It is far more difficult, though, for a sensor to know when an existing report is extraneous and should be filtered or discarded. If data from a second sensor were available, the absence of a correlating report could be used as a powerful added test for such a confirmation. In addition, some categories of extraneous reports that could not be identified at all from a single sensor might be found when netting is used. Finally, second sensor aid may well prevent track swaps or track captures due to ambiguous or suspect target-to-track correlations.

11.1 Data Substitution and Code Improvement

The major causes of missing target reports are blockages (buildings or mountains) and fades (antenna pattern lobing or aircraft banking). Both effects often last for several scans, and hence loss of track could occur unless data were supplied from another sensor. A particularly common data loss situation occurs when an aircraft flies through the sensor cone of silence; here the loss is generally long, and reacquisition especially difficult.

At other times, an ATRBS target report may exist but contain a low confidence identity code or altitude. Either or both could become garbled, and even unusable, when aircraft flight paths cross or fruit is heavy. Garble during aircraft crossing situations may well lead to improper correlations, and track swaps may occur should the code information needed to choose between the reports become obscured. Secondary sensor data could prevent such errors if the alternate direction of view provides garble-free codes for the reports.

Assuming sensor-to-sensor correlation can be done properly (a big assumption for crossing aircraft), the algorithms for supplying missing data from a second sensor are straightforward. However, their ease of application in real-time, and the benefits resulting from their use, remain to be

determined. It is clear that simulation cannot be used to predict these benefits. The characteristics of real data would be needed for the simulation model.

11.2 False Data Suppression

Mode S contains algorithms for identifying various types of false alarm reports, such as reflection false targets, correlating fruit reports, ringaround and multiple reports, and split reports. However, all such identifications are at best educated guesses. If data from a second sensor were available, the presence or absence of correlating reports could be used as a contradictory or confirming criterion respectively for the decision. In the latter case, the report would be kept and a possible error avoided. When multiple reports with the same Mode S or discrete ATCRBS code exist, such as in ringaround, the one seen by the other sensor would be retained.

The algorithms that can be devised for identifying extraneous reports are totally dependent upon the quality of sensor-to-sensor correlation. These algorithms must also guarantee a very high degree of confidence when a report is to be eliminated, as the worst possible error is the failure to report the existence of a real aircraft. Thus, it is possible that two or more scans will be required, for example, to declare a track to be due to a reflection. The absence of a live testbed has precluded development of algorithms in these cases, and no recommendations as to the value of netting for data editing can yet be attempted.

11.3 Correlation Improvement

Whenever two similarly coded ATCRBS aircraft are crossing, it is possible for target-to-track correlation to make incorrect pairings. This is true either when the codes are identical (such as both 1200) or differ only in bits undergoing garble. A second sensor may be able to aid in the correlation decision by providing a non-garbled view of the two reports, as shown in Fig. 11-1.

The algorithm for performing this task would appear to require complex inter-sensor correlation, involving for example a 2-dimensional deviation score. At present, no attempt to develop the procedure has been made. Simulations could aid in this process, so live data is not a necessity, merely desirable.

Help from a second sensor may also be important in preventing suspect target-to-track correlations from causing track captures. Whenever the proper report for a track is absent, the track is subject to correlation with extraneous data. Such events can cause tracking errors that may prevent later reacquisition by the correct data. With only a single sensor, all one-on-one associations must be accepted, even when suspect. Secondary sensor data could prevent these errors either by supplying the missing real reports or by indicating the extraneous ones to be false.

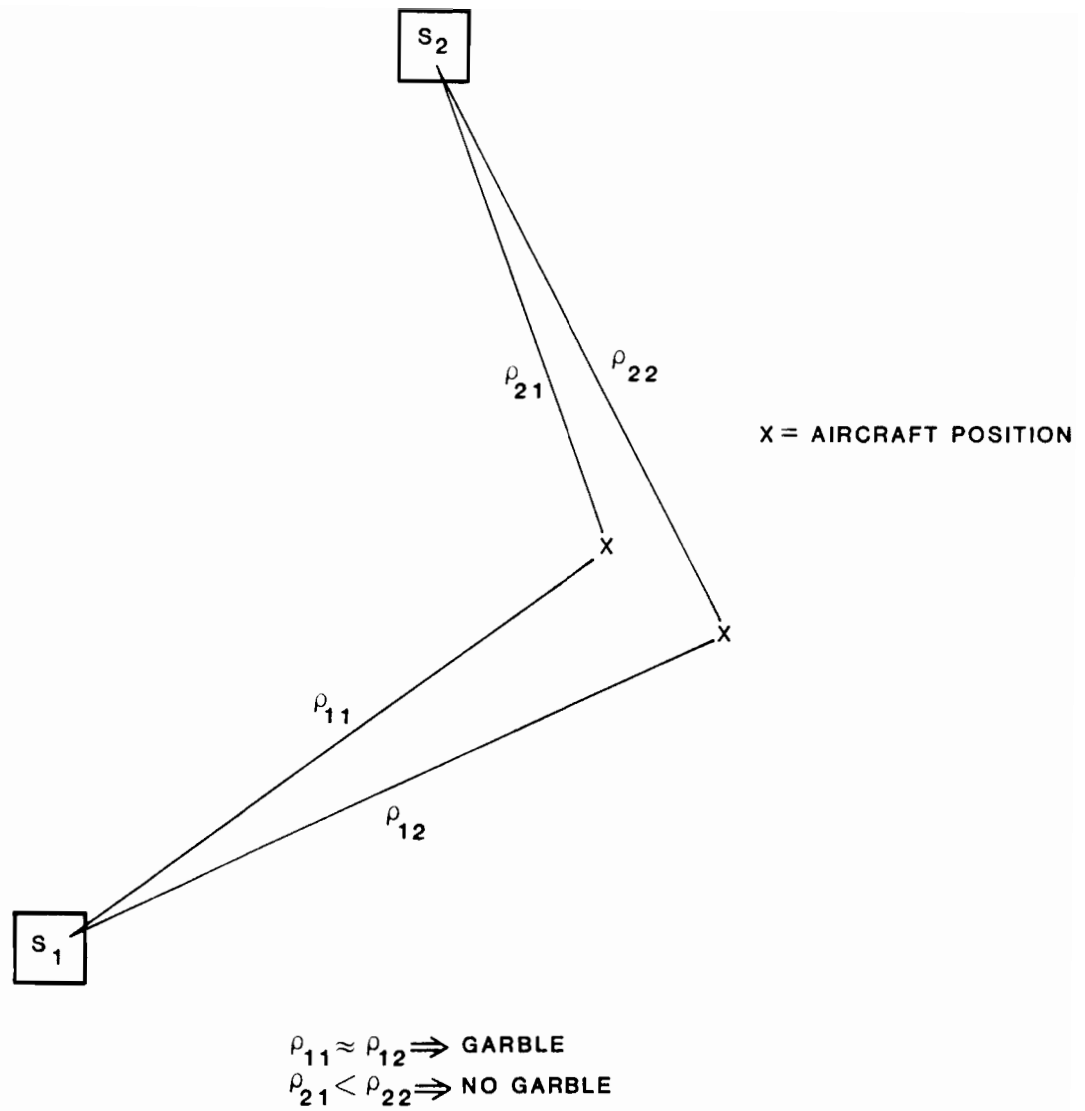


Fig. 11-1. Alternate view of garble.

Such secondary sensor help will also permit the ATCRBS tracker to follow aircraft maneuvers more rapidly. The present sensor design includes a turn detection algorithm whose purpose is to prevent a track from straying due to correlation with bad data. Unfortunately, true sharp turns are flagged by this algorithms as well. Secondary sensor confirmation of a turn would eliminate tracker delay.

12.0 NETTING DEMONSTRATION FACILITY

The principal tool of the surveillance netting project was to have been a live two-sensor demonstration facility. This facility would have permitted realistic testing of the algorithms developed in the study as well as providing a viewing area for real time netting. At the time the project concluded, this facility was nearing completion.

This chapter provides an overall description of the hardware, software, and communications that was being developed to implement this facility. The reason for the presentation is to document the effort to date to permit resumption at a later time if desired.

The basis of the facility is a Data General Eclipse S-250 computer. This computer provides a central point at which data from two or more sensors could be combined and visually examined in a variety of modes. The viewing function is provided via a Megatek display computer tied to the Eclipse. The two sensors currently used for live data gathering are the fixed MODSEF facility and the mobile AMPS sensor.

12.1 Computer Configuration

Figure 12-1 illustrates the various computers that constitute the demonstration facility, as well as the interconnections linking them. The SEL-86 computer has served for a decade as the processing center of the MODSEF sensor. Its only output link has been via a Nova 800 computer that drives a controller display. The difficulty of changing the SEL hardware configuration made it necessary to keep the Nova in the system.

MODSEF target reports will be passed first from the SEL to the Nova, and then from the Nova to the Eclipse. The latter link employs a synchronous interface as shown in the figure. Since both computers are made by Data General, this hardware link is straightforward.

The AMPS sensor uses a Digital Equipment PDP 11/55 computer for its processing component. Since AMPS is to be a remote mobile sensor, its connection to the Eclipse must be via a phone line and a pair of modems. During real-time centralized netting configuration tests, this link will be used to output AMPS reports to the Eclipse. For real-time localized netting configuration experiments, the link will carry data in both directions: requests for SEL reports and reports for display to the Eclipse, reports in response to requests to AMPS. Finally, in the hybrid configuration, the link will also be duplex: reports for display to the Eclipse, improved track files to AMPS.

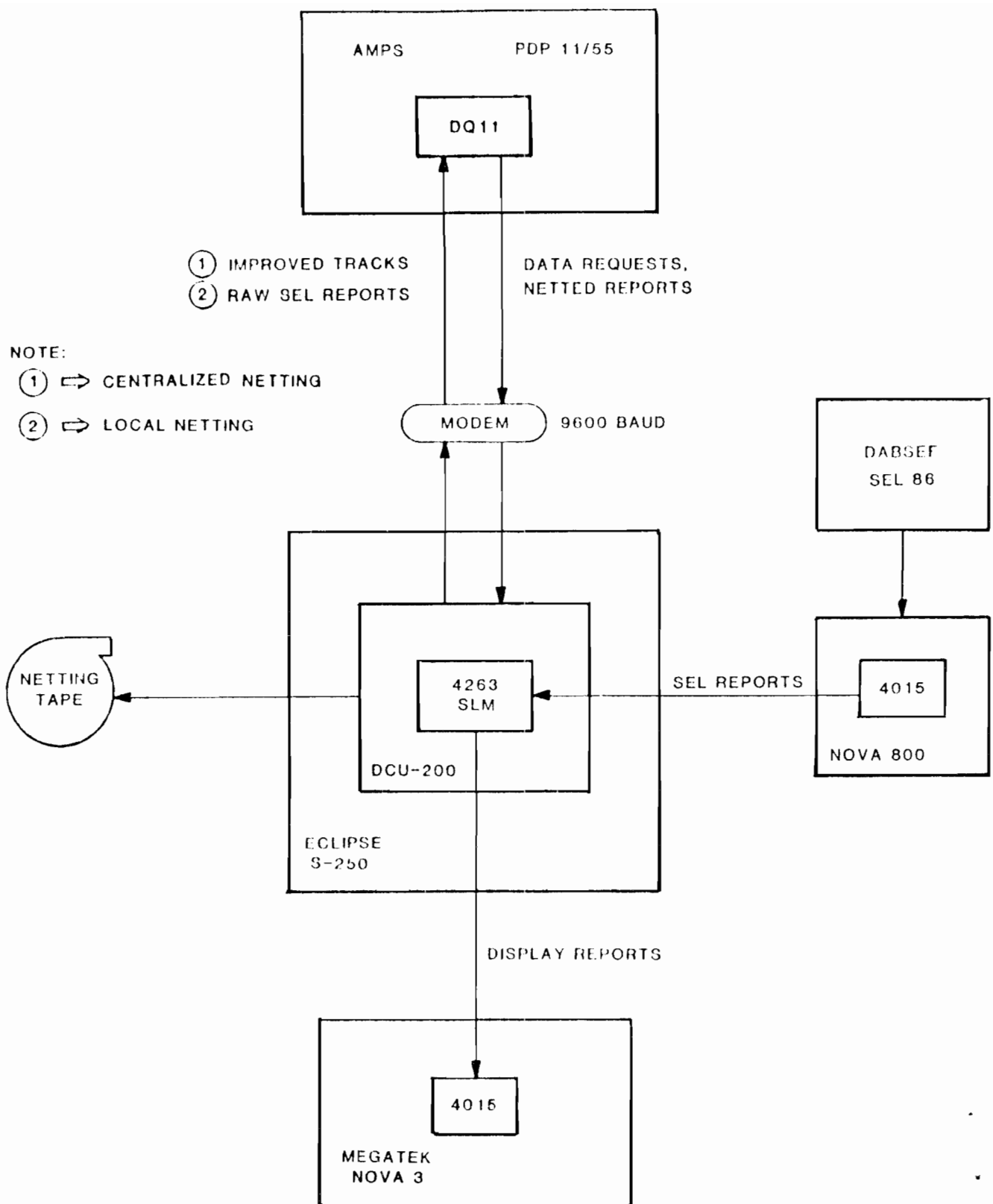


Fig. 12-1. Demonstration Facility architecture.

The Megatek computer serves as the system display terminal. No matter which configuration is being tested, all reports to be displayed will be shipped to it by the Eclipse, even if the Eclipse merely serves as a data transfer agent (such as for reports originating at AMPS). This decision was made for two reasons. First, since the Eclipse and Megatek (whose processor is a Nova 3 computer) are both Data General units, the hardware link is simplified. Second, the Eclipse can then contain special filtering programs to allow it to select the reports desired by the user-chosen display option (AMPS only, SEL only, netted) without the need to modify any other system computer.

12.2 Communications Channels

All data links connecting components to the central Eclipse are implemented by synchronous RS-232 lines operating in full duplex mode at 9600 baud. These lines are controlled by a DCU-200 (data communications unit), which is a separate high-speed I/O processor, tied to the Eclipse, and communicating with it via direct memory access (DMA). The DCU eliminates the processing overhead involved with handling the synchronous lines, and simplifies future I/O configuration changes.

The link between the Eclipse and the AMPS sensor, shown in Fig. 12-2, utilizes a four-wire, full duplex, dedicated phone circuit. The specific modems, elements, and connections that constitute this circuit are defined in the figure. The DQ11 at the AMPS end is a device that controls the I/O interface through hardware and software routines in a manner similar to the Eclipse DCU. Both of them monitor and handle the various buffers and interrupt vectors required by the full duplex operation.

The communications protocol for this link provides for message typing and synchronization. Each transmission, whether from AMPS or from the Eclipse, requires both a header block and a data block. The formats defined for these entities are outlined in Figs. 12-3 and 12-4 respectively.

The header block is a fixed size, so its handling is known before decoding. It then specifies the size of the following data block, so the I/O processing routines can be set up to properly receive it. Both blocks, as shown, contain synchronization bytes (Synch) and error detection bytes (CRC and LRC) to prevent misinterpretations or data errors from occurring.

12.3 Timing Considerations

Prior to the netting project, neither MODSEF nor AMPS target reports contained their time of measurement. Such times are a key ingredient of any netting algorithm, since without them data alignment from multiple sensors is impossible. Thus, the times had to be added in some manner. This requirement was easily met for AMPS reports, as its 10-word format includes 28 consecutive bits of mode 2 information. This mode will not be used for netting, and so the report time will be stored in this field.

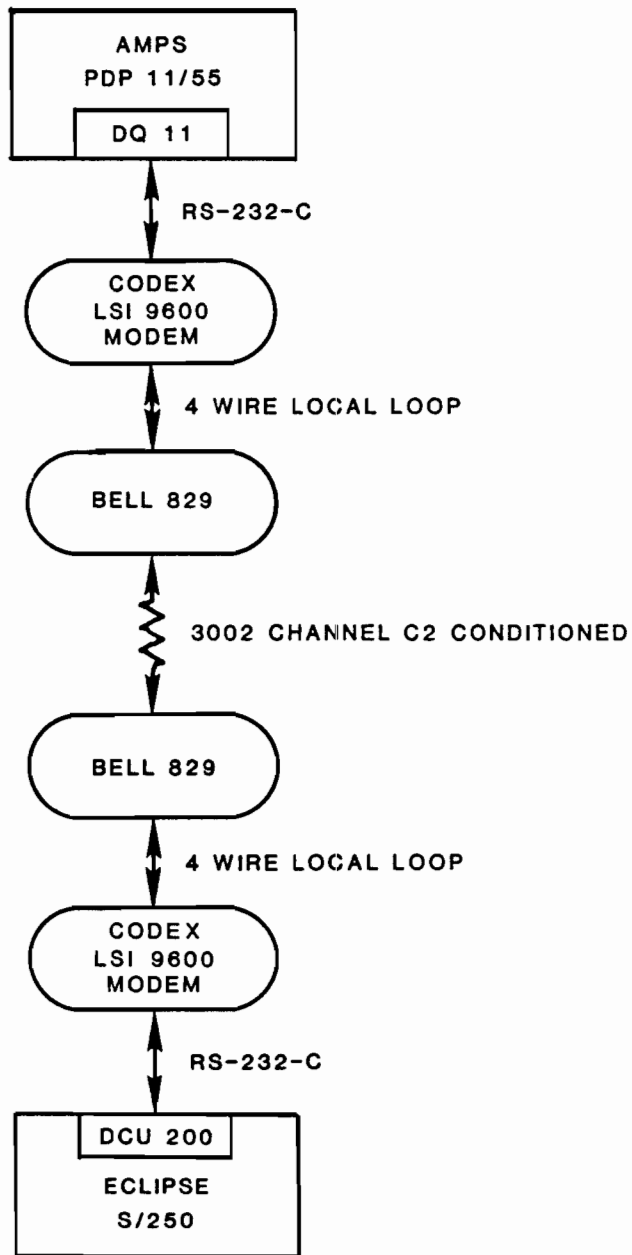


Fig. 12-3. AMPS/Eclipse link.

BYTE	0	START OF HEADER
	1	MESSAGE TYPE (TO FOLLOW)
	2	BYTE COUNT OF DATA BLOCK
	3	(TO FOLLOW)
	4	TRANSMISSION SEQUENCE NUMBER
	5	LAST SEQUENCE NUMBER RECEIVED
	6	FLAG
	7	SPARE
	8	CYCLIC REDUNDANCY CHECK
	9	(CRC)
	10	PARITY (LRC)
		SYNCH
		⋮
		⋮

Fig. 12-3. Header block format.

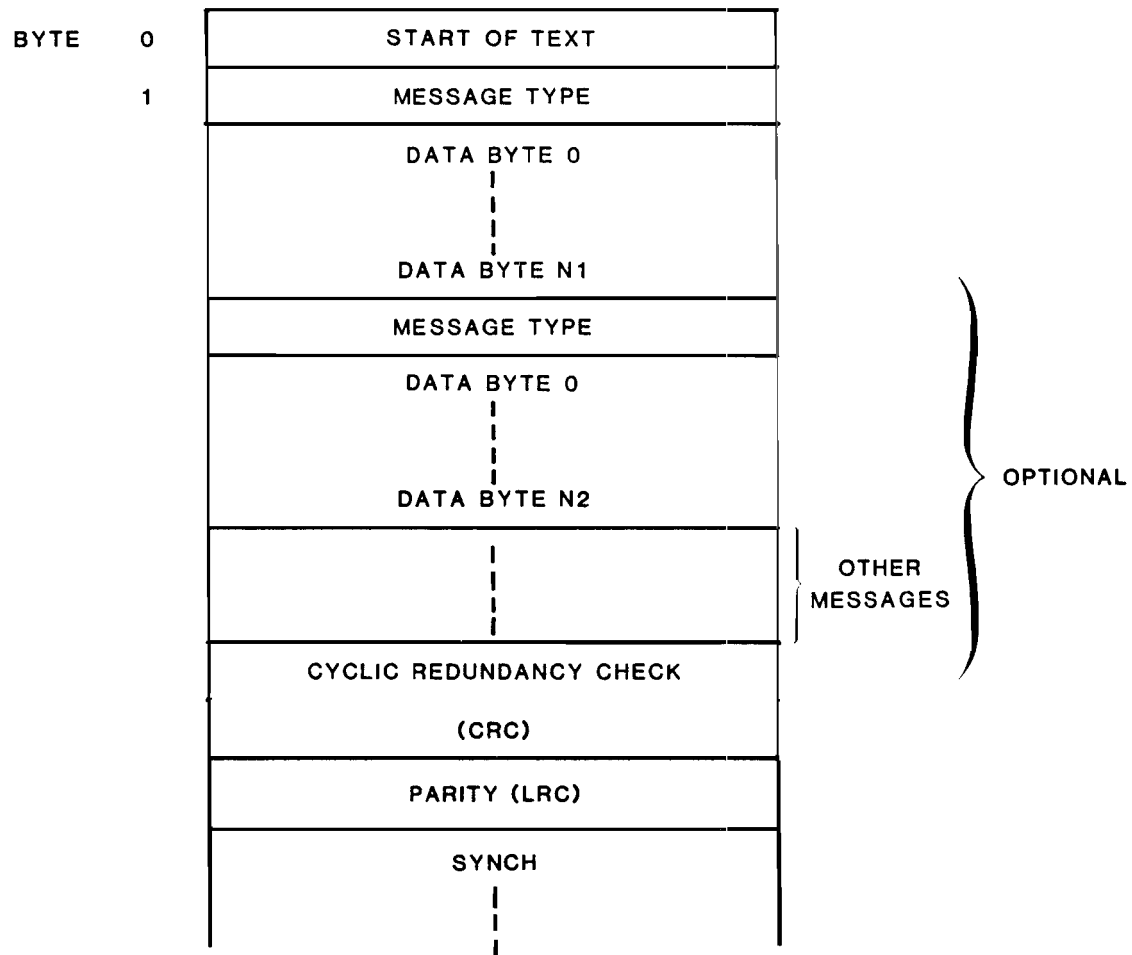


Fig. 12-4. Data block format.

MODSEF reports, unfortunately, are only 8 words long and contain no extra fields. Adding 2 words to these reports would have involved recoding numerous complex SEL functions and thus was not a viable option. Other simpler modifications, such as adding new types of output entities, would have required changes to the SEL/NOVA link as well as preventing netting runs from being used by other MODSEF users. Thus, the only option found acceptable was to add an extra target report to each sector output buffer. This report, whose format is shown in Figure 12-5, provides a time/azimuth synchronization for the sensor by reporting the time corresponding to the end of the sector. Then the Eclipse can compute the time for any real report in the buffer by interpolating between the times in the current and previous time/azimuth reports to obtain the time corresponding to the azimuth of the report. The range placed in these extra reports is greater than 200 miles, so they will automatically be filtered out by other data users.

Time alignment between the AMPS and MODSEF clocks is essential if these report times are to have any meaning. Thus, a True Time Instruments satellite clock was purchased for each sensor. These clocks provide a stable, synchronized time source. The MODSEF clock was tied to the existing computer real-time clock so as to provide its time source, while the AMPS clock can be read directly from a memory location by the software via a DSS11 special interface.

12.4 Netting Software

The major software routines written to date for the demonstration system reside in the Eclipse, where they run under the AOS advanced operating system. These routines process the communications on the various data links and handle the interface between application software and DCU routines. A set of routines were implemented which simplify the interfaces to the synchronous I/O while providing complete control over the protocol, format, and configuration of each link independently.

A number of routines were also written for the Megatek display computer. Surveillance data from several sensors may be displayed individually, simultaneously, or jointly in netted form on the scope. This data can also be recorded for later playback or analysis.

At the time the project ended, programs for performing the actual data netting functions were under development. These would have permitted the system to perform in centralized, localized, or hybrid modes. For the most part, these programs implement the system described in detail in Chapter 4. Inter-sensor correlation, request generation and processing, and multilateration algorithms were all to be incorporated in the Eclipse and PDP 11/55 computers.

RANGE > 200 MILES	
AZIMUTH SECTOR END	
CODE SPECIAL VALUE	
CONFIDENCE HIGH	
TIME CORRESPONDING TO	

SECTOR END AZIMUTH	
-0-	
-0-	

TIME/AZ
SYNCHRONIZATION

Fig. 12-5. Time synchronization report.

Software was also under development to permit data analysis to be generated during live system runs. Such analysis would have included:

1. position improvement measurements
2. correlation improvement measurements
3. inter-sensor correlation accuracy
4. false alarm identification accuracy
5. communications overhead
6. percent of time each type of netting was required
7. tradeoffs of centralized versus localized systems

12.5 Netting Experiments

Once the netting demonstration system was completed, a number of experiments would have been run to test, analyze, and quantify the netting algorithm performance. Centralized netting would have been tried first, as it is conceptually simpler and it forms a superset of the results obtainable by localized netting.

These experiments would have resulted in recommendations as to which netting algorithms were suitable for implementation in FAA systems.

13.0 STUDY ACHIEVEMENTS

Azimuth accuracy is generally improved via multilateration. Unfortunately, many different multilateration formulas are required to cover all aircraft under surveillance. Presence or absence of altitude reporting, and precise calibration or not of the transponder, generate four different cases. Also, sensor biases can affect any multilateration formula, possibly even degrading its performance below that of single sensor surveillance. All of these issues and cases were covered in the study.

In addition, a completely new method for improving azimuth consistency via netting was developed in this project. This approach is a modified form of incremental bilateration, and uses a flat earth model. Surprisingly, this method was found to be at least as accurate as the spherical earth multilateration approaches, and it applies to all cases described above. It even handles sensor biases without noticeable degradation, and can survive with only minor loss of accuracy in diffraction zones. This method is felt to be the major result to date of the netting project.

The problem of surveillance of non-altitude-reporting aircraft was addressed in this study. In particular, two major results were demonstrated. First, netting will permit altitude estimation at least accurately enough to differentiate between two crossing aircraft that are threats to each other, and two aircraft that are well separated vertically. Second, accurate azimuth determination can be made with netting even when aircraft altitude is unknown. Thus, much of the work reported here is applicable to primary, as well as secondary, radar systems.

A number of trackers, both well known and novel, were considered and developed in this study. Two reasons exist for this interest. First, existing trackers do not adequately handle turning aircraft, and second, netting data places new requirements on any smoothing algorithm. The tracker found to provide the best performance is a new, two phase, heading Kalman filter. The study of filters was not concluded at study termination, but a fairly complete presentation is provided in this report.

The other major aspect of netting that received detailed consideration early in the study was inter-sensor communications and algorithms for supplying and using netted data. These issues address the framework of the netted system in which the improvement algorithms can reside. Particular care was taken to construct a framework that meshed well with the existing stand-alone sensor, even to the extent of integrating the new required techniques into existing algorithms.

The result of this integrated netted system design was an algorithm architecture quite similar to the existing one. Tracks are still updated once per scan, although now with netted reports. Target-to-track correlation is performed as before, only now requests for help from other sensors are added to raw reports generated locally. The correlation algorithm used to match tracks from different sensors is described here in both a general and specific

manner, the former to provide guidelines, the latter as an example. The major system addition was an inter-sensor communications link. Its message protocol and formats are described in this report, as well as the data structures needed to store and process the messages.

The area of data editing, or using secondary sensor reports to identify false alarms and improve the code and altitude of real reports, was studied only theoretically.

The main area of netting not covered was implementation in a real system. Since the test and demonstration system was not completed, no idea of the actual improvement that is obtainable exists. Also, such issues as computer power and memory size needed remain unresolved.

REFERENCES

1. V.A. Orlando, P.R. Drouilhet, "Mode S System Description," Project Report ATC-42B, Lincoln Laboratory, M.I.T., (27 October 1982), FAA-RD-82/52.
2. "ARTS III Multisensor Utilization Study Report," Sperry Univac Defense Systems Division, Report PX-10288 (June 1973).
3. W.L. Fischer, C.E. Muehe, A.G. Cameron, "Registration Errors in a Netted Air Surveillance System," Technical Note 1980-40, Lincoln Laboratory, M.I.T., (2 September 1980), DTIC-AD A093691.
4. J.L. Gertz, "The ATCRBS Mode of DABS," Project Report ATC-65, Lincoln Laboratory, M.I.T., (31 January 1977), FAA-RD-76-39.
5. "Handbook of Geophysics for Air Force Designers", Geophysics Research Directorate, Air Force Cambridge Research Center, Air Research and Development Command, United States Air Force (1975).
6. R.G. Mulholland, D.W. Stout, "Numerical Studies of Conversion and Transformation in a Surveillance System Employing a Multitude of Radars, Part I," FAA/NAFEC, (April 1979), FAA-NA-79-17,
7. B.H. Cantrell et al, "Formulation of a Platform-to-Platform Radar Integration System," NRL Memorandum Report 3404 (December 1976).
8. B.H. Cantrell, "Adaptive Tracking Algorithm," NRL Memorandum Report 3037 (April 1875).
9. The Analytic Sciences Corporation, A. Gelb Editor, Applied Optimal Estimation, (M.I.T. Press 1874), Pgs. 182-203.
10. G.V. Trunk and J. D. Wilson, "Tracking Filters for Multiple-Platform Radar Integration," NRL Report 8087 (December 1976).
11. S. Kotz and N.L. Johnson Editors in Chief, Encyclopedia of Statistical Sciences, (John Wiley and Sons, 1982), Vol. 2, Pgs. 242-253.

APPENDIX A

Inter-Sensor Report Conversion

This appendix presents the mathematics associated with converting a target report from the position coordinates and aircraft viewing time of one sensor (S_1) to the position coordinates and aircraft viewing time of another sensor (S_2). Let

$$\begin{aligned} T_1 &= \text{viewing time at sensor } S_1 \\ T_2 &= \text{viewing time at sensor } S_2 \end{aligned}$$

and define the position variables to be used as follows (Fig. A-1):

	S_1, T_1	S_2, T_1	S_2, T_2
radar coordinates	ρ_1, θ_1	$\hat{\rho}_2, \hat{\theta}_2$	ρ_2, θ_2
cartesian coordinates	x_1, y_1	$\hat{x}_2, \hat{y}_2$	x_2, y_2
height above sea level	h	h	h
altitude coordinate	z_1	$\hat{z}_2$	-
radar rates	$\dot{\rho}_1, \dot{\theta}_1$	$\dot{\rho}_2, \dot{\theta}_2$	-
cartesian rates	$\dot{x}_1, \dot{y}_1$	$\dot{x}_2, \dot{y}_2$	-

The middle column represents an intermediate step in the computation. Dashes signify values not required for the conversion. Finally, the coordinates of the two sensors are represented by:

$$\lambda_i = \text{latitude of sensor } i$$

$$\gamma_i = \text{longitude of sensor } i$$

$$h_i = \text{altitude of sensor } i$$

and E is the radius of the earth at the average of the sensor latitudes.

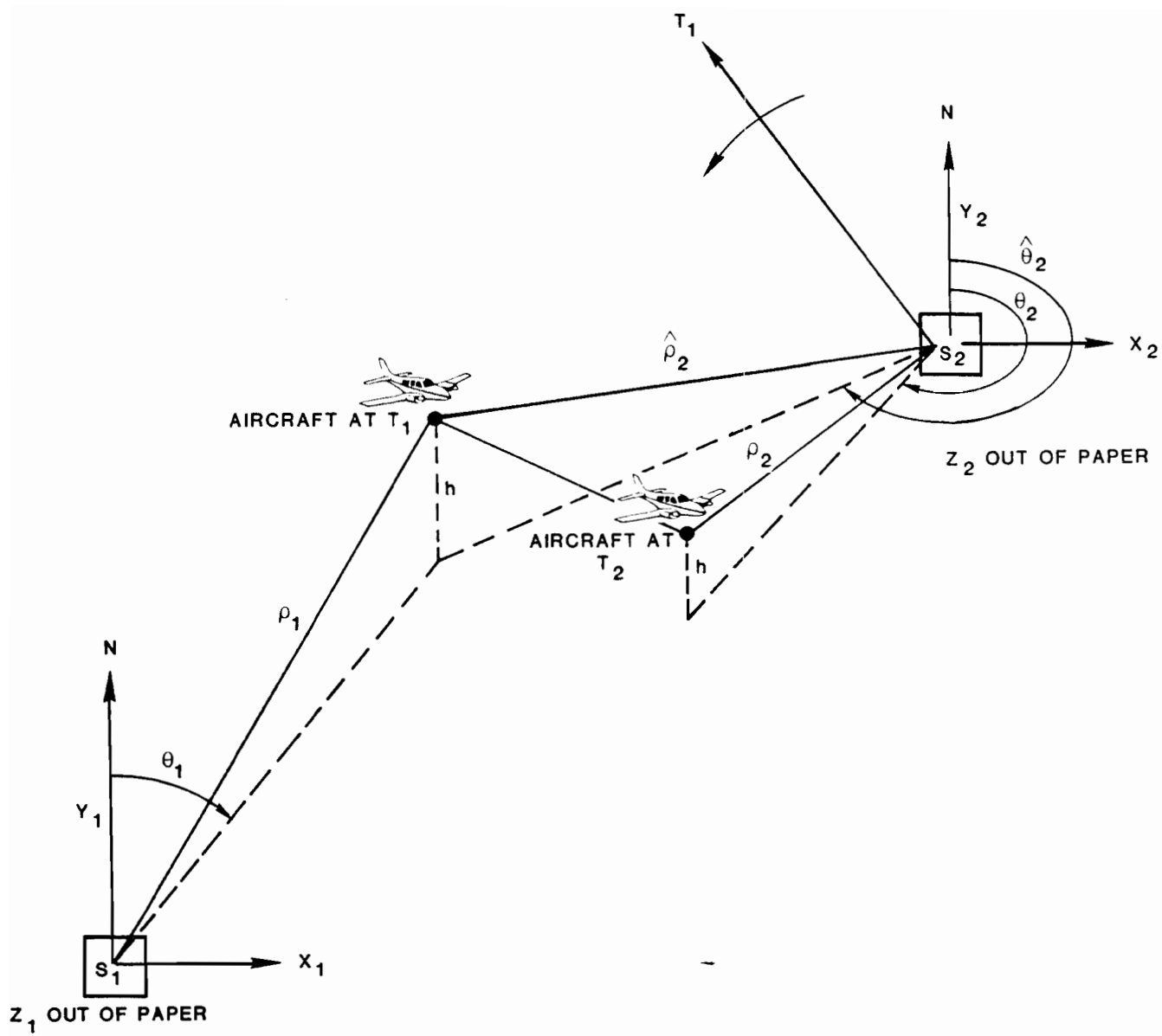


Fig. A-1. Coordinate conversion geometry.

The first step in the transformation is the conversion of the input report from radar to cartesian coordinates:

$$z_1 = \frac{(h-h_1)^2 + 2(h-h_1)(E+h_1) - \rho_1^2}{2(E+h_1)}$$

$$\rho_{1g} = \sqrt{\rho_1^2 - z_1^2}$$

$$x_1 = \rho_{1g} \sin \theta_1$$

$$y_1 = \rho_{1g} \cos \theta_1$$

$$\dot{x}_1 = \frac{\rho_1}{\rho_{1g}} x_1 \dot{\rho}_1 + y_1 \dot{\theta}_1$$

$$\dot{y}_1 = \frac{\rho_1}{\rho_{1g}} y_1 \dot{\rho}_1 - x_1 \dot{\theta}_1$$

$$\dot{z}_1 = - \frac{\rho_1}{(E+h_1)} \dot{\rho}_1$$

By assumption in this step, the aircraft is not climbing or descending. To first order in its effect this is always true because of the small vertical rates involved. In addition, no vertical rate is maintained in Mode S track files.

The second step is to transform from sensor 1 to sensor 2 coordinates:

$$\begin{bmatrix} \hat{x}_2 \\ \hat{y}_2 \\ \hat{z}_2 \end{bmatrix} = T \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} + U$$

where the matrix elements are given by:

$$T_{11} = \cos(\gamma_1 - \gamma_2)$$

$$T_{12} = -\sin\lambda_1 \sin(\gamma_1 - \gamma_2)$$

$$T_{13} = \cos\lambda_1 \sin(\gamma_1 - \gamma_2)$$

$$T_{21} = \sin\lambda_2 \sin(\gamma_1 - \gamma_2)$$

$$T_{22} = \sin\lambda_1 \sin\lambda_2 \cos(\gamma_1 - \gamma_2) + \cos\lambda_1 \cos\lambda_2$$

$$T_{23} = -\cos\lambda_1 \sin\lambda_2 \cos(\gamma_1 - \gamma_2) + \sin\lambda_1 \cos\lambda_2$$

$$T_{31} = -\cos\lambda_2 \sin(\gamma_1 - \gamma_2)$$

$$T_{32} = \cos\lambda_1 \sin\lambda_2 - \sin\lambda_1 \cos\lambda_2 \cos(\gamma_1 - \gamma_2)$$

$$T_{33} = \cos\lambda_1 \cos\lambda_2 \cos(\gamma_1 - \gamma_2) + \sin\lambda_1 \sin\lambda_2$$

$$U_1 = (E + h_1) \cos\lambda_1 \sin(\gamma_1 - \gamma_2)$$

$$U_2 = -(E + h_1) [\cos\lambda_1 \sin\lambda_2 \cos(\gamma_1 - \gamma_2) - \sin\lambda_1 \cos\lambda_2]$$

$$U_3 = (E + h_1) [\cos\lambda_1 \cos\lambda_2 \cos(\gamma_1 - \gamma_2) + \sin\lambda_1 \sin\lambda_2] - (E + h_2)$$

In addition:

$$\dot{x}_2 = T_{11}\dot{x}_1 + T_{12}\dot{y}_1 + T_{13}\dot{z}_1$$

$$\dot{y}_2 = T_{21}\dot{x}_1 + T_{22}\dot{y}_1 + T_{23}\dot{z}_1$$

$$\dot{z}_2 = T_{31}\dot{x}_1 + T_{32}\dot{y}_1 + T_{33}\dot{z}_1$$

Then, since

$$\hat{\theta}_2 = \tan^{-1} \left(\frac{\hat{x}_2}{\hat{y}_2} \right)$$

we find:

$$\dot{\theta}_2 = \frac{\hat{y}_2 \dot{\hat{x}}_2 - \hat{x}_2 \dot{\hat{y}}_2}{\hat{x}_2^2 + \hat{y}_2^2}$$

The next step is to determine the time T_2 at which sensor 2 will view the target. If it is assumed that the sensor is at azimuth ϕ_2 at time T_1 , and the antenna rate of revolution is $\dot{\phi}_2$ (essentially constant), the sensor viewing time will satisfy:

$$\phi_2 + \dot{\phi}_2(T_2 - T_1) = \hat{\theta}_2 + \dot{\theta}_2(T_2 - T_1)$$

where it is assumed that $\dot{\theta}_2$ has very small acceleration (true for targets not near the sensor). Thus

$$T_2 = T_1 + \frac{\hat{\theta}_2 - \phi_2}{\dot{\phi}_2 - \dot{\theta}_2}$$

where the subtraction in the numerator is modulo 2π .

Finally, the output target report position can be determined:

$$x_2 = \hat{x}_2 + \dot{x}_2(T_2 - T_1)$$

$$y_2 = \hat{y}_2 + \dot{y}_2(T_2 - T_1)$$

$$z_2 = \hat{z}_2 + \dot{z}_2(T_2 - T_1)$$

$$\rho_2 = \sqrt{\hat{x}_2^2 + \hat{y}_2^2 + \hat{z}_2^2}$$

$$\theta_2 = \tan^{-1}\left(\frac{x_2}{y_2}\right)$$

APPENDIX B

Heading Error Due to Transponder Bias

The presence of a transponder turnaround delay error (Δ) in an aircraft causes the sensor to misread the true aircraft range. This appendix derives the calculated heading error that results from this data bias. The geometry of the situation under study, and the definitions of the notation being employed, are provided by Fig. B-1. In this appendix all ρ_i 's are ground range ρ_{ig} 's; the g has been dropped for ease of reading.

The true aircraft heading is given by:

$$\begin{aligned} h &= \tan^{-1} [\tan h] = \tan^{-1} \left[\frac{x_2 - x_1}{y_2 - y_1} \right] \\ &= \tan^{-1} \left[\frac{\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1}{\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1} \right] \end{aligned} \quad (B-1)$$

The heading calculated at the sensor, in the presence of Δ , is:

$$h' = \tan^{-1} \left[\frac{(\rho_2 + \Delta) \sin \theta_2 - (\rho_1 + \Delta) \sin \theta_1}{(\rho_2 + \Delta) \cos \theta_2 - (\rho_1 + \Delta) \cos \theta_1} \right] \quad (B-2)$$

Manipulating h' yields:

$$\begin{aligned} h' &= \tan^{-1} \left[\frac{(\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1) + \Delta(\sin \theta_2 - \sin \theta_1)}{(\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1) + \Delta(\cos \theta_2 - \cos \theta_1)} \right] \\ &= \tan^{-1} \left[\frac{\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1 \left\{ 1 + \Delta \frac{\sin \theta_2 - \sin \theta_1}{\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1} \right\}}{\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1 \left\{ 1 + \Delta \frac{\cos \theta_2 - \cos \theta_1}{\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1} \right\}} \right] \end{aligned} \quad (B-3)$$

For any reasonable transponder bias, $\Delta \ll 1$. Thus:

$$\begin{aligned} h' &\approx \tan^{-1} \left[\frac{\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1}{\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1} \left\{ 1 + \Delta \left(\frac{\sin \theta_2 - \sin \theta_1}{\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1} - \frac{\cos \theta_2 - \cos \theta_1}{\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1} \right) \right\} \right] \end{aligned}$$

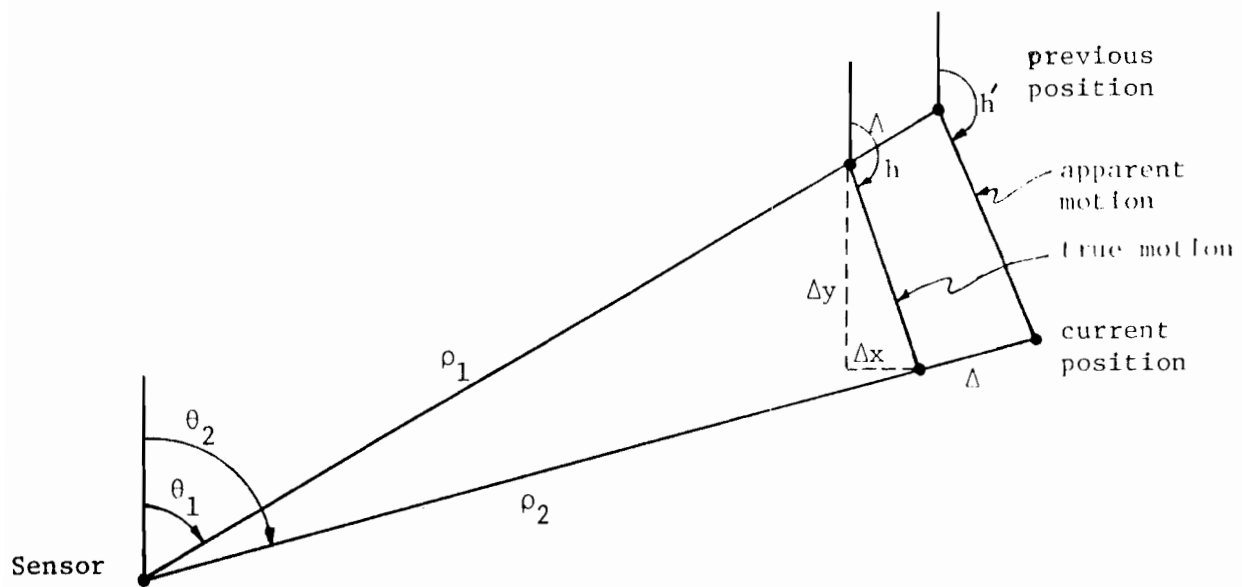


Fig. B-1. Heading geometry.

$$= \tan^{-1} \left[\frac{\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1}{\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1} + \frac{\Delta}{(\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1)^2} * \right. \\ \left. \left\{ (\sin \theta_2 - \sin \theta_1) (\rho_2 \cos \theta_2 - \rho_1 \cos \theta_1) \right. \right. \\ \left. \left. - (\cos \theta_2 - \cos \theta_1) (\rho_2 \sin \theta_2 - \rho_1 \sin \theta_1) \right\} \right] \quad (B-4)$$

Now employ the mean value theorem:

$$\sin \theta_2 - \sin \theta_1 = (\theta_2 - \theta_1) \cos \theta_a$$

$$\cos \theta_2 - \cos \theta_1 = -(\theta_2 - \theta_1) \sin \theta_b$$

where θ_a and θ_b lie in the interval between θ_1 and θ_2 .

Defining θ_d as $\theta_2 - \theta_1$ and using (B-1) yields:

$$h' = \tan^{-1} \left[\tan h + \frac{\Delta}{(y_2 - y_1)^2} \left\{ \theta_d \cos \theta_a (y_2 - y_1) + \theta_d \sin \theta_b (x_2 - x_1) \right\} \right] \quad (B-5)$$

For small Δ , we can apply the derivative rule

$$f(x+c) \approx f(x) + cf'(x)$$

to obtain:

$$h' \approx h + \frac{1}{1 + \left(\frac{x_2 - x_1}{y_2 - y_1} \right)^2} \frac{\Delta}{(y_2 - y_1)^2} \left[\theta_d \cos \theta_a (y_2 - y_1) + \theta_d \sin \theta_b (x_2 - x_1) \right] \\ = h + \frac{\Delta \theta_d}{(y_2 - y_1)^2 + (x_2 - x_1)^2} \left[\cos \theta_a (y_2 - y_1) + \sin \theta_b (x_2 - x_1) \right] \quad (B-6)$$

Thus the heading error is given by:

$$h_e = \frac{\Delta \theta_d}{(y_2 - y_1)^2 + (x_2 - x_1)^2} \left[\cos \theta_a (y_2 - y_1) + \sin \theta_b (x_2 - x_1) \right] \quad (B-7)$$

This error can be bounded by noting that:

$$|\cos \theta_a| \leq 1$$

$$|\sin \theta_b| \leq 1$$

$$\left| \frac{y_2 - y_1}{(y_2 - y_1)^2 + (x_2 - x_1)^2} \right| \leq \frac{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}}{(y_2 - y_1)^2 + (x_2 - x_1)^2} = \frac{1}{d}$$

$$\left| \frac{x_2 - x_1}{(y_2 - y_1)^2 + (x_2 - x_1)^2} \right| \leq \frac{1}{d}$$

where d is the distance between the aircraft positions on the two scans.
Thus:

$$|h_e| \leq \frac{2\Delta\theta_d}{d} \quad (\text{B-8})$$

Finally, for subsonic aircraft $\rho_1 \approx \rho_2 = \rho$, making it true that:

$$\rho \theta_d = d$$

and hence the result becomes:

$$|h_e| \leq \frac{2\Delta}{\rho} \quad (\text{B-9})$$

This error is less than 1° , for any transponder within specifications, for any aircraft beyond 5 miles from the sensor.

APPENDIX C

Sensitivity of Three-Sensor Altitude Estimates

This appendix investigates the sensitivity of the three-sensor altitude estimate to noise in the sensor range measurements. To simplify the analysis, the "altitude" of the aircraft relative to a plane passing through the three sensors is considered instead of the true altitude from the earth's surface. Figure C-1 presents the geometry to be employed. Clearly the error in this pseudo-altitude estimate will behave virtually identically to that of the real one.

First the (x, y, z) position of the aircraft relative to this planar coordinate system must be found. The three equations that this position must satisfy, one for each sensor, are:

$$x^2 + y^2 + z^2 = \rho_1^2 \quad S_1 \quad (C-1)$$

$$(d_{12} - x)^2 + y^2 + z^2 = \rho_2^2 \quad S_2 \quad (C-2)$$

$$(a - x)^2 + (b - y)^2 + z^2 = \rho_3^2 \quad S_3 \quad (C-3)$$

Subtracting equation (C-2) from equation (C-1) yields x :

$$\begin{aligned} -d_{12}^2 + 2d_{12}x &= \rho_1^2 - \rho_2^2 \\ x &= \frac{\rho_1^2 - \rho_2^2 + d_{12}^2}{2d_{12}} \end{aligned} \quad (C-4)$$

Then, y is found by subtracting equation (C-3) from equation (C-1):

$$\begin{aligned} -a^2 + 2ax - b^2 + 2by &= \rho_1^2 - \rho_3^2 \\ y &= \frac{\rho_1^2 - \rho_3^2 + a^2 + b^2 - 2ax}{2b} \end{aligned} \quad (C-5)$$

where x is given by (C-4). Finally, z is found from equation (C-1):

$$z = \sqrt{\rho_1^2 - x^2 - y^2} \quad (C-6)$$

where (C-4) and (C-5) specify x and y .

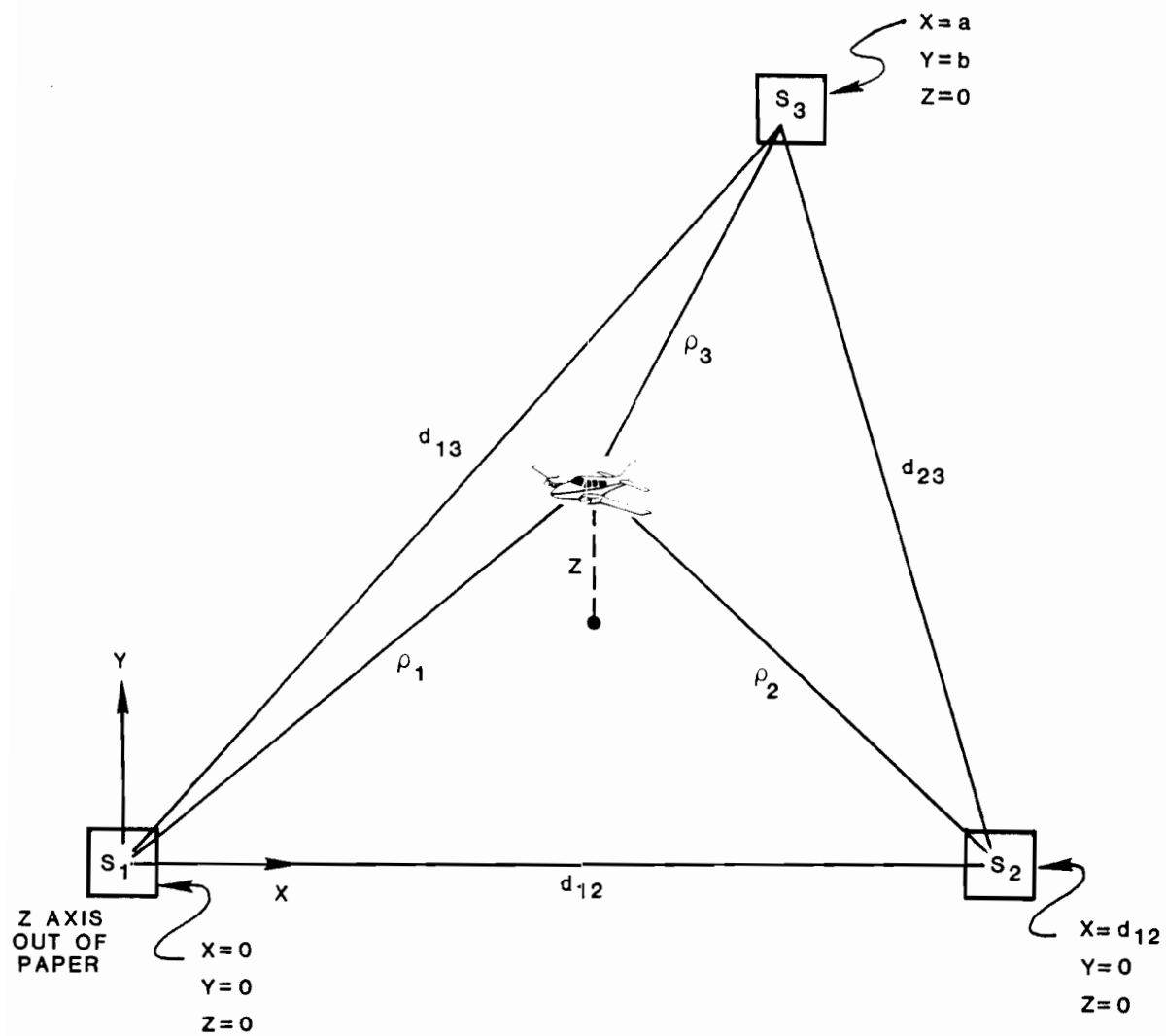


Fig. C-1. Three sensor altitude geometry.

Since a and b are independent of the ranges, the range derivatives of z are simple to compute. The results are found as follows:

ρ_1

$$\frac{\partial x}{\partial \rho_1} = \frac{2\rho_1}{2d_{12}} = \frac{\rho_1}{d_{12}}$$

$$\frac{\partial y}{\partial \rho_1} = \frac{2\rho_1}{2b} - \frac{2a}{2b} \frac{\partial x}{\partial \rho_1} = \frac{\rho_1}{b} \left(1 - \frac{a}{d_{12}}\right)$$

$$\begin{aligned} \frac{\partial z}{\partial \rho_1} &= \frac{1}{2z} \left[2\rho_1 - 2x \frac{\partial x}{\partial \rho_1} - 2y \frac{\partial y}{\partial \rho_1} \right] \\ &= \frac{\rho_1}{z} \left[1 - \frac{x}{d_{12}} - \frac{y}{b} \left(1 - \frac{a}{d_{12}}\right) \right] \end{aligned}$$

(C-7)

ρ_2

$$\frac{\partial x}{\partial \rho_2} = -\frac{2\rho_2}{2d_{12}} = -\frac{\rho_2}{d_{12}}$$

$$\frac{\partial y}{\partial \rho_2} = -\frac{2a}{2b} \frac{\partial x}{\partial \rho_2} = \frac{a}{b} \frac{\rho_2}{d_{12}}$$

$$\begin{aligned} \frac{\partial z}{\partial \rho_2} &= \frac{1}{2z} \left[-2x \frac{\partial x}{\partial \rho_2} - 2y \frac{\partial y}{\partial \rho_2} \right] \\ &= \frac{\rho_2}{z} \left[\frac{x}{d_{12}} - \frac{a}{b} \frac{y}{d_{12}} \right] \end{aligned}$$

(C-8)

ρ_3

$$\frac{\partial x}{\partial \rho_3} = 0$$

$$\frac{\partial y}{\partial \rho_3} = -\frac{2\rho_3}{2b} = -\frac{\rho_3}{b}$$

$$\frac{\partial z}{\partial \rho_3} = \frac{1}{2z} \left[-2x \frac{\partial x}{\partial \rho_3} - 2y \frac{\partial y}{\partial \rho_3} \right]$$

$$= \frac{\rho_3}{z} \begin{bmatrix} y \\ - \\ b \end{bmatrix} \quad (C-9)$$

Thus, the derivatives (C-7), (C-8), and (C-9) are all of the form

$$\frac{\partial z}{\partial \rho_i} = \frac{k}{\tan \psi} \quad (C-10)$$

and the elevation angle is the critical parameter for the error sensitivity.

APPENDIX D

Flat Earth Inter-Sensor Distance

The spherical-equivalent flat earth model requires that the two sensors be treated as if they were separated by the distance D given by:

$$D = d(1 + \frac{z_0}{E}) \quad (D-1)$$

where d , as shown in Fig. D-1, is the distance between the earth surface locations of the two sensors. The usually computed straight line distance d_2 between the sensors, however, differs from this value whenever either sensor has a non-zero altitude. This appendix relates d to d_2 so that the calculation (D-1) can be performed. Figure D-1 should be referenced for definitions of the notation being employed.

First, using the properties of similar triangles:

$$\frac{h_{s2}+E}{m} = \frac{E}{d} \quad (D-2)$$

so that:

$$m = \frac{d(h_{s2}+E)}{E} = d + \frac{dh_{s2}}{E} \quad (D-3)$$

Next, by the law of cosines:

$$d_2^2 = m^2 + (h_{s1}-h_{s2})^2 - 2m(h_{s1}-h_{s2}) \cos \left(\frac{\pi}{2} + \frac{\Phi}{2} \right) \quad (D-4)$$

But by trigonometric identities and the figure, the cosine can be expressed as:

$$\cos \left(\frac{\pi}{2} + \frac{\Phi}{2} \right) = -\sin \left(\frac{\Phi}{2} \right) = -\frac{d/2}{E} \quad (D-5)$$

Substituting (D-5) into (D-4):

$$d_2^2 = m^2 + (h_{s1}-h_{s2})^2 + m(h_{s1}-h_{s2}) \frac{d}{E} \quad (D-6)$$

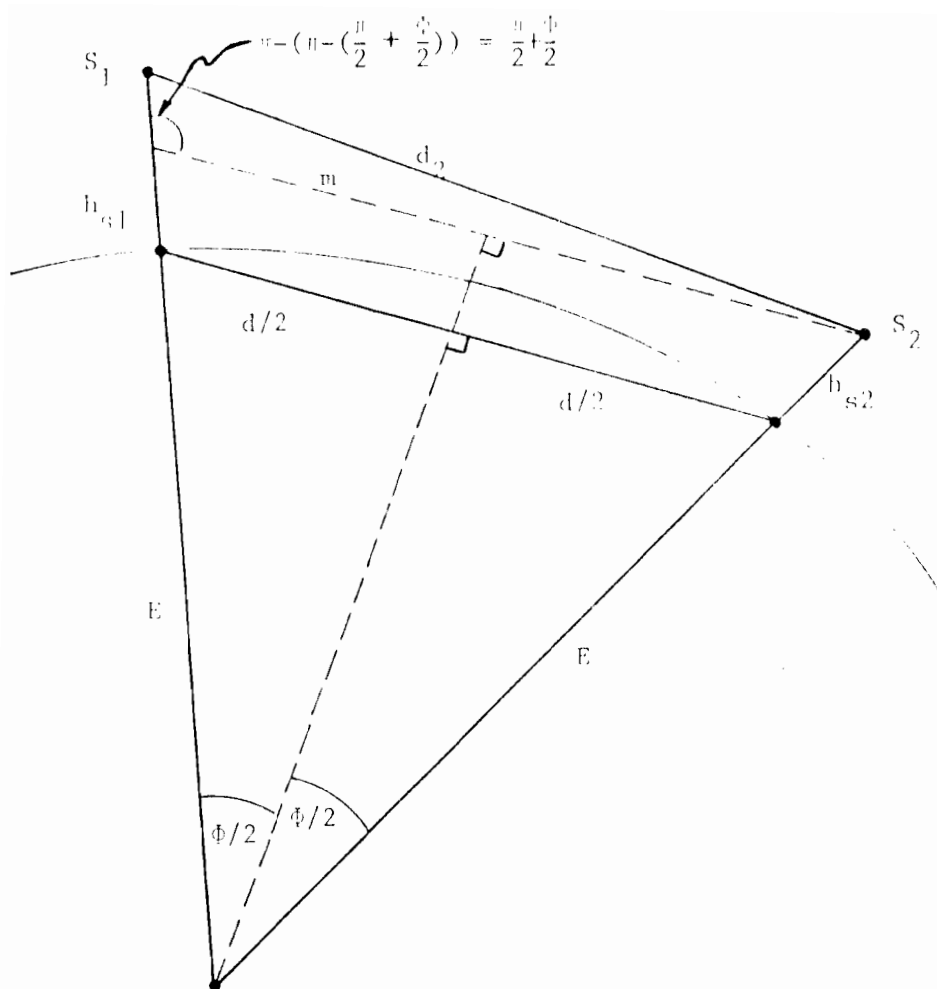


Fig. D-1. Inter-sensor distance geometry.

Then using (D-3) yields:

$$\begin{aligned}
 d_2^2 &= d^2 + \frac{2d^2 h_{s2}}{E} + \frac{d^2 h_{s2}^2}{E^2} + (h_{s1} - h_{s2})^2 + \frac{d^2}{E} (h_{s1} - h_{s2}) + \frac{d^2 h_{s2}}{E^2} (h_{s1} - h_{s2}) \\
 &= d^2 + (h_{s1} - h_{s2})^2 + \frac{d^2}{E} (h_{s1} + h_{s2}) + \frac{d^2 h_{s1} h_{s2}}{E^2} \quad (D-7)
 \end{aligned}$$

Solving for d and using the fact that $h_{si} \ll E$:

$$\begin{aligned}
 d^2 \left(1 + \frac{h_{s1} + h_{s2}}{E} + \frac{h_{s1} h_{s2}}{E^2} \right) &= d_2^2 - (h_{s1} - h_{s2})^2 \\
 d^2 &\approx \frac{d_2^2 - (h_{s1} - h_{s2})^2}{1 - \frac{h_{s1} + h_{s2}}{E} - \frac{h_{s1} h_{s2}}{E^2}} \\
 &\approx \frac{d_2^2 - (h_{s1} - h_{s2})^2}{1} - \frac{d_2^2 (h_{s1} + h_{s2})}{E} + \frac{(h_{s1} - h_{s2})^2 (h_{s1} + h_{s2})}{E} \quad (D-8)
 \end{aligned}$$

so that

$$d = d_2 \sqrt{1 - \frac{(h_{s1} - h_{s2})^2}{d_2^2} - \frac{(h_{s1} + h_{s2})}{E} + \frac{(h_{s1} - h_{s2})^2 (h_{s1} + h_{s2})}{d_2^2 E}} \quad (D-9)$$

Finally, it is almost always true for reasonable bilateration that $d_2 \gg h_{si}$, and thus:

$$d \approx d_2 - \frac{(h_{s1} - h_{s2})^2}{2d_2} - \frac{(h_{s1} + h_{s2})d_2}{2E} + \frac{(h_{s1} - h_{s2})^2 (h_{s1} + h_{s2})}{2d_2 E} \quad (D-10)$$

which is the desired result.