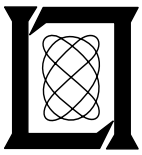
Project Report
ATC-131

TCAS II ATCRBS Surveillance Algorithms

M.L. Wood

28 January 1985

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

TECHNICAL REPORT STANDARD TITLE PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/PM-84/19		

4. Title and Subtitle	5. Report Date
TCAS II ATCRBS Surveillance Algorithms	28 January 1985
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
M. Loren Wood	ATC-131

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Massachusetts Institue of Technology Lincoln Laboratory P.O. Box 73 Lexington, MA 02173-0073	11. Contract or Grant No. DOT-FA77-WAI-817
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address Program Engineering and Maintenance Service Department of Transportation Federal Aviation Administration Washington, D.C. 20591	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 F19628-85-C-0002.

16. Abstract

The Traffic Alert and Collision Avoidance System (TCAS) has been developed to reduce mid air collisions between transponder equipped aircraft. The TCAS concept encompasses a range of capabilities. TCAS I is a low-cost version which provides traffic advisories only. TCAS II adds vertical resolution advisories and is intended to provide a comprehensive level of separation assurance in all current and predicted airspace environments through the end of this century. Enhanced TCAS II uses more accurate intruder bearing data to allow it to generate horizontal resolution advisories. All three forms of TCAS equipment track aircraft equipped with both the existing Air Traffic Control Radar Beacon System (ATCRBS) transponders and with the new Mode S transponders. A TCAS equipped aircraft makes ATCRBS or Mode S range measurements on nearby aircraft. The ATCRBS or Mode S replies contain the altitude of the aircraft if it has an encoding altimeter. TCAS II uses range rate and altitude rate to decide if a collision is imminent. Therefore the replies from a given aircraft must be tracked and correlated in range and altitude. This report documents surveillance techniques developed by Lincoln Laboratory for use by TCAS II equipment in tracking aircraft equipped with ATCRBS transponders. Specifically, it describes the two tracking algorithms used for ATCRBS replies. One algorithm is for aircraft that report alitude, and the other is for those that do not.

17. Key Words		18. Distribution Statement
Surveillance	Correlation	Document is available to the public through the National Technical Information Service, Springfield, Virgina 22161.
Radar	ATCRBS	
Secondary radar	Mode S	
Collision		

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	42	

Form DOT F 1700.7 (8-69)

CONTENTS

CONTENTS (CONT'D)

1.0 INTRODUCTION AND SUMMARY

1.1 Background

The traffic Alert and collision Avoidance System (TCAS) is a beacon-based airborne collision avoidance system that operates by providing air-to-air surveillance of all transponder-equipped aircraft.

The TCAS concept encompasses a range of capabilities. TCAS I is a low-cost version which provides traffic advisories only. TCAS II adds vertical resolution advisories and is intended to provide a comprehensive level of separation assurance in all current and predicted airspace environments through the end of this century. Enhanced TCAS II uses more accurate intruder bearing data to allow it to generate horizontal resolution advisories. All three forms of TCAS equipment track aircraft equipped with both the existing Air Traffic Control Radar Beacon System (ATCRBS) transponders and with the new Mode S transponders.

TCAS equipment operates by interrogating once each second and measuring reply delay to determine the range of nearby aircraft. The replies to these interrogations contain the altitude of the aircraft if it includes an encoding altimeter. TCAS II uses the range and range rate of the aircraft to determine if it is a collision threat. The relative altitude and altitude rate of the aircraft is used to determine proper maneuver direction for collision avoidance. Thus the TCAS equipment must perform both range and altitude tracking on all aircraft that respond to its interrogations.

This report documents tracking techniques developed by Lincoln Laboratory for use by TCAS II equipment in tracking aircraft equipped with ATCRBS transponders. It describes two correlation and tracking algorithms used for ATCRBS replies. On algorithm is for aircraft that report altitude and the other is for those that do not report altitude. These algorithms are intended to be used by TCAS equipment that meets requirements for Minimum TCAS II as defined in Reference 1. Minimum TCAS II equipment employs a four-beam directional antenna mounted on top of the aircraft and an omnidirectional antenna mounted on the bottom of the aircraft. As described in Reference 1, minimum TCAS II transmits a "whisper-shout" sequence of interrogations of varying power in each beam position as a means of overcoming sychronous garble.

A preliminary surveillance technique for tracking altitude-reporting (AR) transponders was developed by the MITRE Corporation in the mid 1970's (Ref.2.) This design was expanded and improved upon by Lincoln Laboratory, and tested in the Los Angeles (LA) Basin in 1982, where it performed very well. The performance in LA and a description of the units (except for the surveillance algorithm) appears in Ref.3.

Lincoln Laboratory then developed the additional capability to perform surveillance on non-altitude-reporting (NAR) transponders. NAR surveillance is complicated by the absence of the altitude codes which are normally used to help distinguish between replies transmitted from different aircraft. Without altitude, false tracks can start by chance combinations of replies, including fruit, from different NAR aircraft. The performance of the NAR surveillance was validated with the same LA data, with altitude codes suppressed.

1.2 Design Approaches

Both the AR and NAR efforts were undertaken with some restrictions. The AR improvements were limited to modifications to the original computer program provided to Lincoln Laboratory. The first NAR algorithm was based on the AR algorithm, and tested on low density data. The later NAR improvements were made only after the high-density LA data became available.

1.2.1 AR Design Approach

The original AR algorithm included an interference resistant sequence of four interrogations per second, and a computer program that tracked all apparent aircraft. Each track had an associated altitude that was assumed to stabilize to a correct value after a period of time. This approach had two major problems. One was a very high incidence of tracks at the wrong altitude, despite waiting many seconds for the altitude to stabilize. The other was a serious phenomenon called "track bloom" in which large numbers of immature tracks used up all the computer's execution time. Various fixes to these problems had been proposed, but a re-examination of the underlying premises of the algorithm and a study of reply data led to the following conclusions:

a. The data was good enough in most cases to quickly form tracks at the correct altitude; there was no need to wait for 30 seconds.

b. The formation of new tracks should be based on solid evidence from nearly uncorrupted data, not on the inferred presence of an aircraft.

Therefore, the algorithm was changed so it forms tracks only on good reply data, and immediately provides such tracks to the collision avoidance function.

1.2.2 NAR Design Approach

The first attempt at an NAR algorithm was based on the AR algorithm with two modifications. First, an α,β,γ range tracker (which tracks the squares of the range measurements) was used for smoothing NAR tracks in place of the AR algorithm's α,β tracker. Secondly, "zero" was treated as a permissible altitude code. This change permitted the algorithm to form and update tracks from empty bracket replies which had been discarded as "illegal" in the AR

algorithm. Range-squared tracking is appropriate for CAS surveillance, because the square of the distance between any two non-accelerating aircraft is a parabolic function of time. The resulting NAR algorithm performed well in low and medium density airspace (typically less than 0.03 aircraft/sq nmi). It was first implemented in the Lincoln Laboratory TCAS Experimental Units (TEU's) in mid-1982 and has been used experimentally since then.

When data from the high density (up to 0.26 aircraft/sq nmi) airspace in the LA Basin became available in early 1983, it became apparent that the rate of false NAR tracks was too high, and that the NAR algorithm required modification.

In developing these modifications, it was decided to focus on improvements to reply correlation and range tracking rather than relying on bearing correlation and correlation with the interrogation/suppression level (called a "whisper-shout" bin), because of the greater accuracy and resolution of the available range data. The resulting algorithm makes use of bearing data and whisper-shout information mainly to discard redundant replies in an early stage of processing.

One simple method of reducing the rate of false tracks is to require TCAS to wait longer before establishing a new track. This was the method used by the original tracker. However, compared to the other techniques considered, an increase in time-to-establish is a relatively unattractive way of decreasing false tracks because it always reduces the probability of detecting real tracks. The resulting loss is particularly significant since it typically occurs at the beginning of an inbound track, which is the most important period. By comparison, improved range correlation can both reduce false tracks and improve the probability of detecting real tracks. Therefore, the time to establish a new track was kept at the standard value of four seconds.

1.3 Results

The final versions of both the AR and NAR algorithms were found to perform very well for the high density data collected in the LA Basin in 1982.

1.3.1 AR Results

The final AR algorithm was intensively evaluated for two hours of data from the LA Basin, in the region 2 to 5 nautical miles in range, and +/-10 degrees in elevation.

The performance is described in Reference 3, and summarized here as:

a. AR Probability of track P(T) for the region 2-5 nmi. and +/- 10 degrees elevation. 89.5 %

b. AR P(T) for 19 aircraft that came within 2 nmi. and 1200 feet, during the 50 seconds before closest approach. 97 %

c. The ratio of false AR tracks to the real AR tracks in a.	1.1 %

1.3.2 NAR Performance

The final NAR algorithm was evaluated by comparing its performance both with the performance of the first NAR algorithm and the algorithm used for tracking AR aircraft using flight test reply data on AR aircraft collected during the LA basin measurements. The Mode C codes of these replies were set to zero, so as to appear to be NAR replies, before they were input to the NAR algorithm. The use of AR replies with codes set to zero allowed the NAR performance to be compared to the performance of the AR algorithm. The results are shown below.

NAR ALGORITHM PERFORMANCES
IN TRACKING THE SAME REPLIES
TRACKED BY THE AR ALGORITHM

	FIRST	IMPROVED
a. NAR Probability of track P(T) for the region 2-5 nmi. and +/- 10 degrees elevation.	70%	85.8 %
b. NAR P(T) for 19 aircraft that came within 2 nmi. and 1200 feet, during the 50 seconds before closest approach.	78%	93 %
c. The ratio of false NAR tracks to the real NAR tracks in a.	>25%	1.9 %

The algorithm for tracking NAR aircraft has been improved significantly. Its performance in high traffic density now approaches that of the algorithm for tracking AR aircraft, as reported in 1.3.1.

1.4 Organization of this Report

Section 2 describes the inputs and outputs of the AR and NAR algorithms. Sections 3 and 4 describe the AR and NAR algorithms respectively, followed by their performance in Section 5. Sections 6 through 8 contain detailed "pseudo-code" descriptions of the algorithms.

2.0 ALGORITHM INTERFACES

2.1 Overview

Both the AR and NAR algorithms receive replies from hardware that detects the presence of ATCRBS replies in the listening interval following the transmission of ATCRBS interrogations. Both algorithms output to the collision avoidance algorithms. The output is those replies that correlated to tracks that are at least 4 seconds old, and do not appear to be caused by multipath. The AR replies have unsmoothed range, bearing, and altitude code. The NAR replies may have a range and bearing obtained by averaging the values of several replies which appeared to have come from the same aircraft.

2.2 Input

The input to surveillance processing consists of replies received from the transponders of nearby aircraft. These replies are elicited by own aircraft's whisper-shout interrogations as described below. Each reply contains the following information:

a. range	15 bits	LSB = 62.5 feet
b. altitude (gray code)	12 bits	LSB = 100 feet
c. garble word	12 bits	
d. arrival angle (bearing)	8 bits	LSB = 1.4 deg
e. bearing quality	4 bits	

f. interrogation on which reply was received:
   1. interrogation power
   2. suppression power
   3. antenna (top or bottom)
   4. beam (front, back, right, left, or omni)

The AR algorithm only uses those replies that have a legal gray code, that is the $C_1C_2C_4$ bits must be 001, 010, 011, 100, or 110, and the D_4 bit must be 0. The NAR algorithm only uses replies with gray codes of zero.

The measurement sigmas are about 50 feet in range, and 10 degrees in bearing. The TEU provides a "garble word" containing 12 bits. Each bit corresponds to one of the altitude code pulse positions, and indicates whether or not that position was overlapped by a code pulse position of some other reply. The TEU also provides a "bearing quality" indicator, which contains the number of monopulse bearing measurements (potentially, one per code pulse) that are averaged to obtain the reply's bearing.

2.3 Transmitter/Receiver Control

To control ATCRBS synchronous interference and facilitate TCAS II operation in airspace with higher traffic densities, a sequence of interrogations at different power levels is transmitted during each scan. Each of the interrogations, other than the one at the lowest power, is preceded by a suppression pulse (designated S_1) 2 microseconds before the P_1 pulse. The combination of lower power S_1 and higher power P_1 serves as a suppression transmission when both are detected by the transponder. Thus, a given transponder will ideally respond to each of the interrogations in the sequence in one of the following ways:

a. It detects none of the pulses, and it does not reply.
b. It detects both P_1 and P_2, and it replies.
d. It detects S_1 and P_1, and it is suppressed.

These ideal responses illustrate the whisper-shout principle. There are, of course, other outcomes, such as detecting only one of the P_1, P_2 pulses. The ideal reply mechanism is that the transponder only replies to a few of the interrogations in the sequence. Those at lower power are not noticed, and those at higher power cause suppression. The interference-reduction action arises because it is likely that aircraft with sufficiently similar ranges to cause overlapping replies if both respond to the same interrogation, will actually reply to different interrogations. This is because, even though the propagation losses to each are nearly the same, they can have quite different antenna gain products in relation to the TCAS II unit, different cable losses, minimum trigger levels, etc.

The complete sequence of 24 interrogations is as follows:

If i is the interrogation index, i = 1 to 24, then:

P_1, P_2 pulse transmit power = (57-i) dBm , i = 1 to 24

S_1 pulse transmit power = (57-i) - 2 - δ , i = 1 to 23

S_1 pulse transmit power = 0 , i = 24

where δ is 1 when i is even, and 0 when i is odd

Since aircraft that respond to low power interrogations have low path and other losses, and the reply is made at normal power, the TCAS II can increase its minimum trigger level (MTL) (ie. make itself less sensitive) in the listening interval after the interrogation. This helps reduce the amount of fruit received. The MTL is set to:

$$\text{TCAS MTL} = -74 + Q \quad \text{dBm}$$

where $Q = 0$, i = 1 to 7

$Q = i-7$, i = 8 to 24

The surveillance process operates on a one-second update rate, called a scan.

2.4 Output

The output is fed to the TCAS II collision avoidance algorithm, and consists of replies that correlated with established tracks. The replies contain the following items, and are smoothed by the collision avoidance function:

a. Range Measurement 15 bits LSB = 62.5 feet
b Altitude 12 bits LSB = 100 feet
c. Bearing Measurement 8 bits LSB = 1.4 deg
d. The identification number of the track to which it correlated.

3.0 AR ALGORITHM DESCRIPTION

3.1 Summary of Modifications

The original AR algorithm used all replies to form new tracks, and formed them from any set of four replies (from consecutive scans) that lay in a straight line in range, regardless of whether the reply altitude codes agreed. The track altitude was formed using a "majority vote" rule. During track update, a track was split into as many tracks as there were replies in the range correlation window. The codes of the split-off tracks were formed by logical AND and OR operations that favored the conversion of ONEs to ZEROs, on the theory that the ONEs were actually created by the garbling of ZEROs. The large number of tracks so formed were condensed into a smaller set by deleting, or "merging away", those having a low "firmness score".

The main modifications were to form tracks using only three scans, using only replies that did not correlate to existing tracks, and to insist on substantial agreement among the three codes. Using three scans instead of four increases probability of track, while requiring code agreement reduces the false track rate, as does using only uncorrelated replies. Agreement between the track's and candidate correlating reply's altitudes was also required during track update, thus reducing the false track rate, and eliminating track splits.

3.2 AR Reply Preprocessing

3.2.1 Update Rate

The sequence of whisper-shout interrogations is repeated once per second, as described in Section 2.2. The details of the sequence depend on whether the top-mounted antenna is omni-directional or directional (with front, back, left, and right beams). In either case, there is an omni-directional bottom mounted antenna.

The sequence for a top omni antenna contains the 24 interrogations described in Section 2.2. The sequence for a top directional antenna contains the same 24 interrogations on the forward beam, 20 each on the left and right beams, (eliminating the 4 highest power ones) and 15 on the back beam (eliminating the 9 highest power ones). The side and back beams do not need the higher power interrogations because intruder aircraft in those directions cannot have high range rates. In either case, the bottom omni sequence contains 4 low-power interrogations. The bottom antenna is quite susceptible to multipath, so is only used to "fill in" the portion of the top antenna pattern that is shielded by the fuselage.

The TEU waits 2 ms between interrogations, so the sequence of 83 interrogations takes about 166 ms. The 29-interrogation top omni sequence takes about 58 ms. The sequences are repeated once per second.

3.2.2 Receive Sidelobe Suppression (RSLS)

If the top antenna is directional, all replies having a bearing estimate beyond 65 degrees from the center of the beam are discarded. This eliminates false tracks arising from the "late Mode C reply" mechanism (Ref. 3, Sec. 3.2.).

3.2.3 Elimination of Redundant Replies

The TCAS II usually receives more than one reply per second from each aircraft because of redundancies in the interrogation sequence, beam overlap (when using a directional antenna), and use of a top and bottom antenna. Only one reply per aircraft per scan is needed for track updating and new track formation, and more than one would increase the false track rate. Therefore, redundant replies must be eliminated. The method is to examine the replies in increasing range order, and to eliminate any reply having the same altitude code as its nearest neighbor at shorter range if the range difference between the two is sufficiently small (see Section 7).

3.3 AR Track Updating

As a result of the processing described in Sections 3.2.1 to 3.2.3 the replies are ready for correlation to existing tracks, the replies have legal altitude codes and, normally, a given aircraft is represented by no more than one reply.

3.3.1 Range Correlation Window

A range correlation window is set for each track centered on the range predicted to the current scan time. Range is tracked with an α,β filter. Because such a filter does not adequately model straight line flight when the target aircraft passes by the TCAS II at close range, the range correlation window is a function of range, increasing in size with smaller range. The size of the window also depends on the target altitude and the track's update history, as described in Section 7. The range tracker and correlation windows are those of the original design.

3.3.2 Reply Selection

After all replies that lie in the track's correlation window have been found, their altitudes are examined to select the best reply to use to update the track. The best is the reply with the shortest range that also has the predicted track altitude, or has an altitude within +/-100 feet of the prediction. If none is found, the search is repeated with altitude values +/-200 feet from the prediction. The selection of the shortest range reply rather than the one closest to the range prediction, helps discriminate against multipath replies. Altitude is also tracked with an α,β filter.

3.3.3 Updating the Track

Each track file contains the following items:

range estimate
range rate estimate
altitude estimate
altitude rate estimate

x component of position estimate
x component of velocity estimate
y component of position estimate
y component of velocity estimate
bearing (arctangent of y/x) estimate

age in seconds
current number of consecutive coasts
track identification number
establishment indicator

Range is tracked with the α,β filter of the original design but with new values for α and β. The bearing is tracked in x,y coordinates. The bearing gains are pre-computed and stored in a table that is indexed by track age. The table contains least-squares gains, for which the minimum permitted values were determined by experimentation. Neither the range nor bearing filter adapts to turns or coasts.

A track is deleted when it has coasted for 6 seconds without an update. When a track is updated its coast counter is set to zero. The reply used to update the track is discarded, and cannot be used to update other tracks, or to form new tracks.

3.4 AR Formation of New tracks

New tracks are formed from three replies received on consecutive scans. The replies must lie in an approximately straight line in range and have nearly identical altitude gray codes. First, replies from two scans are used to identify possible tracks with range rates less than 1200 kts. The two reply ranges are then projected ahead one second, and a third reply is looked for in a window just big enough to account for the inaccuracies in the range measurements. Some code differences are allowed, to account for non-level flight and small amounts of corruption from interference. The three codes must agree in all eight of their D, A and B code pulses, or in seven of their D, A and B pulses and at least one of their C pulses.

The test for code agreement among the three replies is made individually for each of the reply pulse positions. This test is based on the presence of code pulses alone: agreement occurs for a given reply pulse position if all three replies are detected with a ONE in that position or all three replies are detected with a ZERO in that position. The confidence associated with those pulse detections does not affect agreement.

When agreement among the three replies does not occur for a given reply pulse position, the initial track pulse code estimate for that position is based on the values of the individual pulse codes and the confidence flags associated with those pulse codes in the three replies.

The confidence flag for a reply pulse position is set LOW whenever there exists another received reply (either real or phantom) that could have had a pulse within +/-0.121 microsecond of the same position. Otherwise, the confidence flag is set HIGH.

When agreement fails for a given pulse position, the rules for estimating the initial track code for that position are based on the principle that LOW-confidence ONES are suspect. The rules are as follows:

a) If in the most recent (third) reply the detected code for a given pulse position is HIGH confidence or a ZERO, the initial track pulse code estimate for that position is the same as the code detected in that position in the most recent reply.

b) If in the most recent reply the detected code for a given pulse position is a LOW-confidence ONE, the initial track pulse code estimate for that position is the same as the code detected in that position in the second reply, provided that was not also a LOW-confidence ONE. If the second was also a LOW-confidence ONE, the initial track pulse code estimate is the same as the code detected in that position in the first reply.

The three replies may or may not have valid bearing measurements. The initial x and y states are estimated as straight lines passing through the valid bearing measurements. If there is only one valid bearing measurement, the x,y rates are set to zero.

3.5 AR Track Merging

This function attempts to eliminate redundant tracks. It considers one of any pair of tracks within 0.082 nmi of each other to be redundant. The rules for deciding which, if any, of the two to delete are based on the age and history of coasts. They are described in detail in Section 7. Established tracks (Section 3.7) cannot be deleted by the merge function.

The merge function was originally much more complex, and quite necessary due to the large numbers of tracks formed by the original algorithm. The number of tracks was large because new tracks were formed without requiring any code agreement among the four replies, all replies (even those that were used to update existing tracks) participated in new track formation, and existing tracks were split into as many tracks as there were replies found in the range correlation windows. Now, the merge function is simpler and rarely deletes a track.

3.6 AR Image Rejection

Replies are sometimes received by specular multipath phenomema, and can be persistent enough to be tracked. Such replies are received from real aircraft in two common ways. One is when both the interrogation and reply were bounced off the ground, or alternatively, when one of the signals was reflected off the ground, and the other travelled by the direct path. The multipath "image" track will usually have the same altitude as the real track, and a range rate which can be calculated from the real aircraft's track parameters. The fact that the range rate of the image of a real track can be calculated (for the one-way and two-way bounces) provides the means for a hypothesis test that can be applied to each track. Note that an image track can thus only be identified as such if the real aircraft which is responsible for the image is itself being tracked.

3.7 AR Track Establishment

A track is considered mature or "established" upon receipt of the first correlating reply. Since three replies were used to initiate the track, establishment occurs with the fourth reply, for tracks that are not marked as images. Replies that correlate to established tracks are sent to the collision avoidance algorithms.

The establishment function was intended in the original algorithm to play a critical role in preventing false tracks from being sent to the collision avoidance algorithms. Establishment times were as long as 30 seconds. The long times degraded the probability of track but were largely ineffective in reducing the false track rate. The new algorithm is so effective that the fourth reply rule is only rarely required for false track suppression.

4.0 NAR ALGORITHM DESCRIPTION

4.1 Summary of Development

The NAR algorithm was developed in two steps. First, the AR algorithm was modified in two simple ways: it tracked zero codes, that is, empty brackets; and secondly, it used an α,β,γ range tracker. This algorithm worked well in low density, but had too high a false track rate when applied to the LA data. Therefore, it was extensively redesigned in the reply correlation section and the range tracking filter. Particular attention was given to selecting the range correlation windows and the behavior of the tracker gains during accelerating flight and after coasts.

4.2 NAR Reply Preprocessing

4.2.1 Update Rate

The sequence of whisper/shout interrogations is the same set used for AR tracking, as described in Section 3.2.1.

4.2.2 Receive Sidelobe Suppression (RSLS)

The NAR RSLS function is identical to that used for AR tracking as described in Section 3.2.2.

4.2.3 Zero Codes and Illegal Codes

The replies received in a given second are divided into three groups, according to their altitude code:

a. legal altitude codes (which are tracked by the AR algorithm)
b. zero codes (empty brackets)
c. illegal codes ($C_1C_2C_4$ bits are 000, 101, or 111; or D_4 is 1)

The first NAR algorithm tracked the combined sets of zero and illegal codes. The illegal codes were simply set to zero, and thus treated as empty brackets. This approach was intended to enable the TCAS II system to detect the presence of aircraft having a faulty altitude reporting system; for example a "stuck" C bit. We now believe that these two groups should be treated separately, for the following reasons:

(a) The occurrence of aircraft with faulty encoders is rare. None were observed in range-versus-time plots of the illegal code replies for the morning flight in Los Angeles on December 5, 1982. However, it was observed that many of the apparently illegal code replies were actually Mode A fruit replies, which clearly should not be allowed to corrupt the tracking of empty brackets.

(b) Observation of plots of the empty brackets showed that the NAR aircraft could be tracked using them alone. Comparing these plots to plots of the empty brackets plus illegal codes showed that the illegal codes added no information about the NAR aircraft. The conversion of empty brackets into illegal codes by garbling was very rare.

Since the illegal codes do not contribute significantly to probability of track, and can increase the false track rate, it was decided not to use them.

4.2.4 NAR Reply Correlation

Ground-based ATCRBS surveillance systems usually receive many replies from a target as the slowly rotating beam sweeps by. These replies are typically combined into a "target report". Early TCAS II designs seldom received more than one reply per aircraft per second, not enough to consider generating target reports from replies. The current TCAS II design receives an average of about two in the omni version, to about three in the directional version. Combining these several replies into a single target report has the advantages of:

a. increasing accuracy (obtained by averaging the reply ranges, and the bearing measurements)

b. eliminating redundant replies, which will reduce the number of track initiations, and

c. providing reports that are more indicative of the presence of an aircraft than a single "unreinforced" reply, which may be merely a fruit reply.

The rules for combining replies into a report are:

a. they must lie within $3*\sigma$(range), i.e., 150 ft, of each other

b. their bearings must lie within $2*\sigma$(bearing), i.e., 20°, of the bearing of the shortest-range reply in the set.

c. their whisper/shout bins must overlap.

The replies that were combined into the report are discarded.

4.3 NAR Track Updating

As a result of the processing described in Sections 4.2.1 to 4.2.4 reports have the following attributes:

a. zero codes

b. bearings within 65 degrees of beam center (directional TCAS II)

c. range and bearing averaged over 2 or more replies.

Replies with the following attributes are discarded:

a. those outside 65 degrees from beam center (directional TCAS II)

b. those that were absorbed into reports.

c. those with illegal codes

Candidate NAR reports and uncombined replies are then correlated to existing tracks, as described below.

4.3.1 Range Correlation Window

A range correlation window is set for each track before any correlations are attempted. A preliminary window is first set for each track. It is centered on the predicted range. Its width (in units of 1/128 nmi) is computed as:

$$W = (7\text{-age}) + 5 + \frac{\text{"up/down" coast count}}{2} + \frac{\text{bias}}{2} \quad ; \text{ but less than } 15$$

A newly initiated track has an age of 3. When the term (7-age) is negative, it is dropped from the formula. The "bias" is a measure of how well the track is following recent range measurements, and is described in Section 8. The windows of adjacent tracks are not allowed to overlap. Regions of overlap are divided in half, and each half is allocated to the appropriate track.

4.3.2 Selection of the "Best" Correlating Reply

The replies/reports that lie in a track's range window are examined to select the "best" one to use to update the track. If no reply/report is within $3*\sigma$(bearing), that is, 30° of the predicted bearing then none is selected, and the track is coasted.

Two options for selecting the best one were initially considered for the case when more than one reply/report lies within 30° of the track. One option selected the shortest-range reply/report. The other selected the one closest to the track range. The latter was found to give the best performance in the absence of multipath.

However, this option did not work well for tracks subject to multipath. Multipath replies have a range greater than the aircraft's. Sometimes a track initiates or updates on a multipath reply/report when one or more real replies are absent. The track range is thus corrupted in the out-range direction. Later, when both real and multipath replies/reports are available, the "closest to the track range" rule erroneously selects the multipath reply.

Therefore the "closest to the track range" rule was modified. Whenever the two candidate replies/reports have nearly the same bearing, the one having the smaller range is selected. This rule helps corrupted tracks get back on the real trajectory.

The whisper-shout interrogation to which a target replies corresponds to the link margin relative to the TCAS II aircraft. The link margin was found to be too variable from second to second to serve as a correlation attribute. The variability is due to the many nulls and lobes in the antenna patterns.

4.3.3 Updating the NAR Track

Each track file contains the following items:

range estimate
range-squared estimate
range-squared rate estimate
range-squared acceleration estimate
track-to-reply bias (lowpass filtered range residuals)
three range-squared tracker gains (alpha, beta, gamma)

x component of position estimate
x component of velocity estimate
y component of position estimate
y component of velocity estimate
bearing (arctangent of y/x) estimate

age in seconds
number of correlating replies (updates) since track initiation
current number of consecutive coasts
up/down coast counter
track identification number
establishment indicator

Range is tracked in the range-squared domain. This is done because the square of the target's range varies parabolically with time in non-accelerating flight. The tracker gains are approximations to a least-squares Kalman formulation. They are recursively computed as the response of a simple lowpass filter to an impulse plus a step. This causes

them to gradually decay to an asymptotic value corresponding to the modest deviations from straight-line motion that are usually observed. Whenever the track is coasted, the gains are immediately raised by another lowpass filter in a way that approximates an optimal formulation.

The asymptotic values of gain are too low to track turning targets. A turn detector is used to lowpass filter the track residuals over the last few seconds. When this "bias" exceeds a threshold, the gains are forced upwards to a value corresponding to normal turn rates.

The bearing is tracked in x,y coordinates. The gains are precomputed and stored in a table that is indexed by track age. The table contains least-squares gains, and the minimum permitted values were determined by experimentation. No attempt is made to detect turns and raise the gain during them. Neither are any gain changes made during coasts.

4.4 NAR Formation of New Tracks

New tracks are formed from three replies/reports received on consecutive scans. The replies/reports must lie in an approximately straight line in range, and have bearings within 20° of each other. The initial range-squared state is estimated as a parabola passing through the three ranges, as described in detail in Section 8.

The three replies/reports may or may not have valid bearing measurements. The initial x and y states are estimated as straight lines passing through the valid bearing measurements.

4.5 NAR Track Merge

This function is identical to that used for AR tracks, as described in Section 3.5.

4.6 NAR Image Rejection

This function is identical to that used for AR tracks, as described in Section 3.6.

4.7 NAR Track Establishment

This function is identical to that used for AR tracks, as described in Section 3.7.

5.0 ALGORITHM PERFORMANCE

5.1 AR Algorithm Performance

The algorithm was evaluated using about 2 hours of data collected by both units in Los Angeles on December 5, 1982. The average density during this period was approximately 0.1 aircraft/sq nmi, with peaking as high as 0.26 aircraft/sq nmi. Three measures of performance were made: "case studies" of 19 close encounters, a statistical study of all proximate targets, and the rate of occurrence of false tracks.

The criteria for a close encounter were that an aircraft came within 2 nmi in range while being within 1200 feet in altitude. A set of 19 such encounters was analyzed in detail. In most of them the target was in track continually throughout the 50 seconds before closest approach. There were a few instances of gaps or late track startups. The percentage of time these aircraft were in track was 97%. Most of the track gaps were due to fades.

The statistical analysis divided the data into one-minute segments, and for each the maximum traffic density was determined. For this purpose, density was computed as the number of aircraft between 2 and 5 nmi divided by the area. The count included all transponder-equipped aircraft, whether or not they were altitude reporting. The counting involved a detailed manual procedure based on computer plots of replies and tracks. Probability of track, P(T), was estimated as the percentage of aircraft-seconds during which the aircraft was in track, limiting attention to "targets of interest". Targets of interest were defined as aircraft within +/-10 degrees in elevation angle and for which both own aircraft and the target were at least 600 feet above ground level. Both directional and omni units were evaluated, and performed about the same. For this analysis the directional unit used 24 interrogations on all four directional beams. The performance was essentially independent of density, being more affected by fading.

All false tracks were identified for both units in the 3 to 5 nmi, +/- 10 degree elevation region. The average performance of the two units was:

a. Probability of track P(T) for the region 2-5 nmi. and +/- 10 degrees elevation.	89.5 %
b. P(T) for 19 aircraft that came within 2 nmi. and 1200 feet, during the 50 seconds before closest approach.	97 %
c. The ratio of false tracks to the real tracks in a.	1.1 %

This performance is well within the needs of the collision avoidance function.

5.2 NAR Algorithm Performance

The NAR algorithm was evaluated with the same data used to evaluate the surveillance performance for AR aircraft. To evaluate the NAR algorithm performance, all Mode C replies in the data were converted to empty brackets, and tracked. The results for three performance measures are given below for the old NAR algorithm, the new NAR algorithm, and the AR algorithm.

	AR ALGORITHM WITH MODE C REPLIES	NAR ALGORITHMS WITH SAME REPLIES	
		OLD	NEW
a. Probability of track P(T) for the region 2-5 nmi. and +/- 10 degrees elevation.	89.5 %	70%	85.8 %
b. P(T) for 19 aircraft that came within 2 nmi. and 1200 feet, during the 50 seconds before closest approach	97 %	78%	93 %
c. The ratio of false tracks to the real tracks in a.	1.1 %	>25%	1.9 %

The new algorithm for tracking NAR aircraft performs significantly better than the old NAR algorithm. It also performs nearly as well as the TCAS II AR algorithm when tracking AR aircraft despite the fact that it has significantly less target information available to it for correlation purposes.

6.0 PSEUDO-CODE DESCRIPTION

Pseudo-code is a method for describing computer algorithms. It is a convenient way to describe an algorithm that is to be programmed in a structured language.

NOTE: THE AR AND NAR ALGORITHMS WERE PROGRAMMED IN ASSEMBLY LANGUAGE. THE PSEUDO-CODE GIVEN IN SECTIONS 7 AND 8 HAS NEVER BEEN COMPILED OR EXECUTED.

The pseudo-code used here contains the following items:

PROGRAM — The PROGRAM is the computer solution of the problem

TASK — TASKs are subdivisions of the PROGRAM. Task names are formed by connecting words with underscored spaces (e.g., "TRACK_UPDATE")

EXECUTIVE — The EXECUTIVE contains CALLs to all the TASKs necessary to perform the program's purpose

FUNCTION — FUNCTIONs are subdivisions of TASKs

CALL — The CALL directs the computer's activity to the called TASK or FUNCTION. The CALLed TASK or FUNCTION, when complete, RETURNs to the activity that follows the CALL

STATEMENT — STATEMENTS are English-language (including common mathematical symbols such as +, * /, = etc., logical variables such as TRUE, FALSE, and trigonometric functions such as cosine and sine) descriptions of what is to be done

IN/OUT — IN and OUT statements are lists of inputs and outputs of tasks

REPEAT WHILE "STATEMENT of the condition under which the STATEMENTs that follow are to be done"
[The REPEAT WHILE is used to indicate the familiar "loop"]

IF
THENIF
ELSEIF
[The IFs are used to direct program control. They always have a subordinate associated THEN (or THENIF), and may have an ELSE (or ELSEIF)]

THEN STATEMENT(s) to be done when the IF, THENIF, or ELSEIF is satisfied

ELSE STATEMENT(s) to be done when the IF, THENIF, or ELSEIF is not satisfied

7.0 AR ALGORITHM PSEUDO CODE DESCRIPTION

7.1 AR Algorithm Executive

The AR algorithm is executed once per second, and consists of calls to the following tasks:

PROGRAM: SURVEILLANCE-OF-ALTITUDE-REPORTING-AIRCRAFT

CALL: FORMATION_OF_ALTITUDE_REPLY_BUFFER
This task identifies replies from different whisper-shout interrogations that were probably received from a single aircraft. The most in-range of those is retained and the others are discarded.

CALL: TRACK_UPDATE
This task predicts the range, altitude, and bearing of each track to the current scan time. A window in range is set up about each range prediction. The most in-range of the ALTITUDE-REPLY-BUFFER replies in the window that also matches the track's altitude is used to update the track. Replies used to update tracks are discarded. Tracks not updated for several scans are deleted.

CALL: FORMATION_OF_NEW_TRACKS
This task uses the ALTITUDE-REPLY-BUFFERS from the last three scans. Triplets of replies that lie in a straight line and whose altitude gray codes are in close agreement are used to form an initial track in range, altitude, and bearing.

CALL: MERGE_TRACKS
This task deletes tracks that appear to be redundant.

CALL: IMAGE_REJECTION
This task deletes tracks that appear to be caused by multipath.

CALL: ESTABLISH_TRACKS
The term "established" means that the track is mature enough to pass to the collision avoidance function. This task determines which tracks are established.

CALL: ROTATE_ALTITUDE_REPLY_BUFFERS
Three ALTITUDE-REPLY-BUFFERS are used; the current buffer, one from one scan ago, and one from two scans ago. The latter is not needed anymore after the ESTABLISH_TRACKS task is complete, so it is emptied and "rotated" back to "current" status for use by the FORMATION_OF_ALTITUDE_REPLY-BUFFER task on the next scan.

7.2 Formation of the "Altitude-Reply-Buffer"

```
TASK  FORMATION_OF_ALTITUDE_REPLY_BUFFER
      IN [REPLY-BUFFER, contains all replies received during the scan]
      OUT [ALTITUDE-REPLY-BUFFER, contains replies for TRACK_UPDATE and
                                  FORMATION_OF_NEW_TRACKS]
arrange the replies in the REPLY-BUFFER in increasing range order
empty the current ALTITUDE_REPLY_BUFFER
KEYRANGE = minus infinity
KEYCODE = zero
KEYQUALITY = zero
REPEAT WHILE (replies remain in the REPLY-BUFFER)
  IF (the altitude code of the next reply is legal)
  | THENIF (the reply's range-KEYRANGE is less than (0.5*σ(range))
  |     | THENIF ( the reply's altitude code is not equal to KEYCODE)
  |     |     | THEN place the reply in the ALTITUDE-REPLY BUFFER
  |     |     |      KEYRANGE = the range of the reply
  |     |     |      KEYCODE = the altitude code of the reply
  |     |     |      KEYQUALITY = the bearing quality of the reply
  |     |     | ELSEIF ( the reply's bearing quality is greater than KEYQUALITY)
  |     |     |     | THEN replace the bearing and bearing-quality values in the
  |     |     |     |                ALTITUDE-REPLY-BUFFER with those of the reply
  |     |     |     |      KEYRANGE = the range of the reply
  |     |     |     |      KEYQUALITY = bearing quality of the reply
  |     |     |     | ELSE KEYRANGE = the range of the reply
  |     |  ELSE place the reply in the ALTITUDE-REPLY BUFFER
  |     |       KEYRANGE = the range of the reply
  |     |       KEYCODE = the altitude code of the reply
  |     |       KEYQUALITY = the bearing quality of the reply
  | ELSE KEYRANGE = the range of the reply
ENDTASK FORMATION_OF_ALTITUDE_REPLY_BUFFER
```

7.3 Track Update

```
TASK  TRACK_UPDATE
      IN/OUT [ALTITUDE-REPLY-BUFFER, track file]
REPEAT WHILE  (tracks remain in track file)
| DT           = time elapsed since last scan
| AGE          = AGE + 1
| RANGE        = RANGE       + RANGERATE * DT
| ALTITUDE     = ALTITUDE    + ALTITUDERATE * DT
| ALTITUDE100 = ALTITUDE quantized to the nearest 100 feet
REPEAT WHILE  (tracks remain in track file)
| HALFWINDOW = 570 feet
| IF ( COASTS is greater than zero)
| | THEN  HALFWINDOW = 760 feet
| IF ( RANGE is greater than or equal to 0.00 nmi. and less than 0.17 nmi.)
| | THEN ADDWINDOW =  2000 feet
| IF ( RANGE is greater than or equal to 0.17 nmi. and less than 0.33 nmi.)
| | THEN ADDWINDOW =  1000 feet
| IF ( RANGE is greater than or equal to 0.33 nmi. and less than 1.00 nmi.)
| | THEN ADDWINDOW =   600 feet
| IF ( RANGE is greater than or equal to 1.00 nmi. and less than 1.50 nmi.)
| | THEN ADDWINDOW =   240 feet
| IF ( RANGE is greater than or equal to 1.50 nmi.)
| | THEN ADDWINDOW =     0 feet
| IF (ALTITUDE is greater than 10000 feet)
| | THEN ADDWINDOW = ADDWINDOW * 4
| HALFWINDOW = HALFWINDOW + ADDWINDOW
| INWINDOW  = RANGE - HALFWINDOW
| OUTWINDOW = RANGE + HALFWINDOW
| IF (this is the first track)
| | THEN INWINDOW = 0
| IF (this is the last track)
| | THEN OUTWINDOW = infinity
| REPEAT WHILE (replies remain in the ALTITUDE-REPLY-BUFFER)
| | IF (reply's range is between INWINDOW and OUTWINDOW, and reply's
| | | altitude is at, or 100 feet above or below ALTITUDE100, and
| | | no reply has been selected yet)
| | | THEN  select this reply to update this track
| IF (no reply has been selected)
| | THEN REPEAT WHILE (replies remain in the ALTITUDE-REPLY-BUFFER)
| |      IF (reply's range is between INWINDOW and OUTWINDOW, and reply's
| |      | altitude is 200 feet above or below ALTITUDE100, and
| |      | no reply has been selected yet)
| |      | THEN select this reply to update the track
| IF (a reply was selected)
| | THEN CALL: TRACK SMOOTHING
| |      delete the selected reply from the ALTITUDE-REPLY-BUFFER
| |      UPDATES = UPDATES + 1
| |      COASTS = 0
| | ELSE COASTS = COASTS + 1
| |      IF (COASTS.GE.6)
| |      | THEN delete the track
arrange the track file in increasing range order
END   TRACK_UPDATE
```

7.3 Track Update (continued)

```
FUNCTION TRACK_SMOOTHING
         IN/OUT [selected reply, trackfile]
X           = X                    XDOT * DT
Y           = Y             +      YDOT * DT
BEARING = ARCTANGENT (Y/X)
RESIDUAL = (reply's range) - (track's RANGE)
RANGE     = RANGE    + RESIDUAL * 0.67
RANGERATE = RANGERATE + RESIDUAL * 0.25
BEARTEST = 22.5 * 2 EXPONENT(BEARCOAST + 1)
IF (absolute value of reply's bearing - BEARING is less than BEARTEST)
| THEN XRESIDUAL = (reply's range * cosine of reply's bearing) - (X)
       YRESIDUAL = (reply's range *   sine of reply's bearing) - (Y)
       IF (track age.LE.8)
       | THEN INDEX = track AGE
       | ELSE INDEX = 8
       XYALPHA = XYALPHA(INDEX)
       XYBETA  = XYBETA(INDEX)
       X    = X    + XYALPHA * XRESIDUAL
       XDOT = XDOT + XYBETA  * XRESIDUAL
       Y    = Y    + XYALPHA * YRESIDUAL
       YDOT = YDOT + XYBETA  * YRESIDUAL
       BEARING = ACRTANGENT (Y/X)
ALTITUDERESIDUAL = (reply's altitude) - (track's ALTITUDE)
ALTITUDE = ALTITUDE + 0.28 * ALTITUDERESIDUAL
ALTITUDERATE = ALTITUDERATE + 0.06 * ALTITUDERESIDUAL
RETURN
END TRACK_SMOOTHING
```

INDEX	XYALPHA	XYBETA
3	0.700	0.300
4	0.600	0.200
5	0.524	0.143
6	0.464	0.107
7	0.417	0.083
8	0.400	0.067

7.4 Formation of New Tracks

```
TASK   FORMATION_OF_NEW_TRACKS
       IN/OUT [ZERO-BUFFER, trackfile]
REPEAT WHILE (replies remain in the 1 second old ALTITUDE-REPLY-BUFFER
 | RANGE1    = range of the next reply
 | HIGHWINDOW = RANGE1 + 2100 feet
 | LOWWINDOW  = RANGE1 - 2100 feet
 | REPEAT WHILE (replies remain in the 2 sec. old ALTITUDE-REPLY-BUFFER
 |   RANGE0 = range of the next reply
 |   IF ((RANGE0.GT.LOWWINDOW).AND.(RANGE0.LT.HIGHWINDOW))
 |   | THEN CENTERWINDOW = RANGE1 + (RANGE1-RANGE2)/2
 |   |      OUTWINDOW    = CENTERWINDOW + 312.5 feet)
 |   |      INWINDOW     = CENTERWINDOW - 312.5 feet)
 |   |      REPEAT WHILE (replies remain in the current ALTITUDE-REPLY-BUFFER
 |   |        RANGE2 = range of the next reply
 |   |        ALTITUDE2 = altitude of next reply
 |   |        IF ((RANGE2.GT.INWINDOW).AND.(RANGE2.LT.OUTWINDOW)
 |   |        | THENIF (all three altitude codes are the same, OR differ in one
 |   |        |          of the C bit positions, or differ in one of the DAB bit
 |   |        |          positions, OR differ in one of the C bits AND one of
 |   |        |          the DAB bits, OR differ in two of the C bits)
 |   |        |          (AGREEMENT IS DETERMINED FROM THE TABLE IN 7.5.1)
 |   |        |              THEN RANGE        = (RANGE0)
 |   |        |                   RANGERATE    = (RANGE2 - RANGE0)/2
 |   |        |                   ALTITUDE     =(AS DETERMINED FROM TABLE 7.5.1)
 |   |        |                   ALTITUDERATE = 0
 |   |        |                   AGE          = 3
 |   |        |                   UPDATES      = 3
 |   |        |                   CALL: INITIALIZE TRACK BEARING
merge the new tracks into the track file, in increasing range order
END    FORMATION_OF_NEW_TRACKS
```

7.4 Formation of New Tracks (continued)

```
FUNCTION  INITIALIZE_TRACK_BEARING
          IN/OUT [ZERO-BUFFERS, track fiile]
BEARING0 = bearing of reply from two second old ALTITUDE-REPLY-BUFFER
BEARING1 = bearing of reply from one second old ALTITUDE-REPLY-BUFFER
BEARING2 = bearing of reply from current ALTITUDE-REPLY-BUFFER
IF (BEARING0 is valid)
 THEN X0 = RANGE0 * COSINE(BEARING0)
      Y0 = RANGE0 *   SINE(BEARING0)
      IF (BEARING1 is valid)
       THEN  X1 = RANGE1 * COSINE(BEARING1)
             Y1 = RANGE1 *   SINE(BEARING1)
             IF (BEARING2 is valid)
              THEN X2 = RANGE2 * COSINE(BEARING2)
                   Y2 = RANGE2 *   SINE(BEARING2)
                    X = 5/6 * X2 + 2/6 * X1 - 1/6 * X0  XDOT = X2/2 - X0/2
                    Y = 5/6 * Y2 + 2/6 * Y1 - 1/6 * Y0  YDOT = Y2/2 - Y0/2
              ELSE  X  = 2 * X1 - X0                    XDOT = X1 - X0
                    Y  = 2 * Y1 - Y0                    YDOT = Y1 - Y0
       ELSEIF (BEARING2 is valid)
            THEN X2 = RANGE2 * COSINE(BEARING2)
                 Y2 = RANGE2 *   SINE(BEARING2)
                  X = 5/6 * X2 + 1/6 *X0          XDOT = X2/2 - X0/2
                  Y = 5/6 * Y2 + 1/6 *Y0          YDOT = Y2/2 - Y0/2
            ELSE  X = X0                          XDOT = 0
                  Y = Y0                          YDOT = 0
 ELSEIF (BEARING 1 is valid)
      THEN X1 = RANGE1 * COSINE(BEARING1)
           Y1 = RANGE1 *   SINE(BEARING1)
           IF (BEARING2 is valid)
            THEN X2 = RANGE2 * COSINE(BEARING2)
                 Y2 = RANGE2 *   SINE(BEARING2)
                  X = X2                          XDOT = X2 - X1
                  Y = Y2                          YDOT = Y2 - Y1
             ELSE X = X1                          XDOT = 0
                  Y = Y1                          YDOT = 0
       ELSEIF (BEARING2 is valid)
            THEN X2 = RANGE2 * COSINE(BEARING2)
                 Y2 = RANGE2 *   SINE(BEARING2)
                 X = X2                           XDOT = 0
                 Y = Y2                           YDOT = 0
            ELSE X = 0                            XDOT = 0
                 Y = 0                            YDOT = 0
RETURN
END   INITIALIZE_TRACK_BEARING
```

7.4.1 Table for New Track Altitude

THIS TABLE DESCRIBES HOW THE ALTITUDE CODE OF A NEW AR TRACK IS FORMED FROM THE ALTITUDE CODES OF THE THREE REPLIES. THE TABLE BELOW IS USED ONCE FOR EACH OF THE CODE BIT POSITIONS. THE COLUMN LABELLED "agree" INDICATES WHETHER OR NOT THE THREE REPLIES ARE CORRELATED IN THE PARTICULAR CODE BIT POSITION, AND THE COLUMN LABELLED "est" GIVES THE VALUE THE TRACK CODE IS TO HAVE IN THAT POSITION.

	reply1,reply2,reply3	agree	est		reply1,reply2,reply3	agree	est
1	0-high,0-high,0-high	yes	0	33	0-high,0-high,0-low	yes	0
2	1-high,0-high,0-high	no	0	34	1-high,0-high,0-low	no	0
3	0-low ,0-high,0-high	yes	0	35	0-low ,0-high,0-low	yes	0
4	1-low ,0-high,0-high	no	0	36	1-low ,0-high,0-low	no	0
5	0-high,1-high,0-high	no	0	37	0-high,1-high,0-low	no	0
6	1-high,1-high,0-high	no	0	38	1-high,1-high,0-low	no	0
7	0-low ,1-high,0-high	no	0	39	0-low ,1-high,0-low	no	0
8	1-low ,1-high,0-high	no	0	40	1-low ,1-high,0-low	no	0
9	0-high,0-low ,0-high	yes	0	41	0-high,0-low ,0-low	yes	0
10	1-high,0 low ,0-high	no	0	42	1-high,0-low 0-low	no	0
11	0-low ,0-low ,0-high	yes	0	43	0-low ,0-low ,0-low	yes	0
12	1-low ,0-low ,0-high	no	0	44	1-low ,0-low ,0-low	no	0
13	0-high,1-low ,0-high	no	0	45	0-high,1-low ,0-low	no	0
14	1-high,1-low ,0-high	no	0	46	1-high,1-low ,0-low	no	0
15	0-low ,1-low ,0-high	no	0	47	0-low ,1-low ,0-low	no	0
16	1-low ,1-low ,0-high	no	0	48	1-low ,1-low ,0-low	no	0
17	0-high,0-high,1-high	no	1	49	0-high,0-high,1-low	no	0
18	1-high,0-high,1-high	no	1	50	1-high,0-high,1-low	no	0
19	0-low ,0-high,1-high	no	1	51	0-low ,0-high,1-low	no	0
20	1-low ,0-high,1-high	no	1	52	1-low ,0-high,1-low	no	0
21	0-high,1-high,1-high	no	1	53	0-high,1-high,1-low	no	1
22	1-high,1-high,1-high	yes	1	54	1-high,1-high,1-low	yes	1
23	0-low ,1-high,1-high	no	1	55	0-low ,1-high,1-low	no	1
24	1-low ,1-high,1-high	yes	1	56	1-low ,1-high,1-low	yes	1
25	0-high,0-low ,1-high	no	1	57	0-high,0-low ,1-low	no	0
26	1-high,0-low ,1-high	no	1	58	1-high,0-low ,1-low	no	0
27	0-low ,0-low ,1-high	no	1	59	0-low ,0-low ,1-low	no	0
28	1-low ,0-low ,1-high	no	1	60	1-low ,0-low ,1-low	no	0
29	0-high,1-low ,1-high	no	1	61	0-high,1-low ,1-low	no	0
30	1-high,1-low ,1-high	yes	1	62	1-high,1-low ,1-low	yes	1
31	0-low ,1-low ,1-high	no	1	63	0-low ,1-low ,1-low	no	0
32	1-low ,1-low 1-high	yes	1	64	1-low ,1-low ,1-low	yes	1

7.5 Merge Tracks

```
TASK    MERGE_TRACKS
        IN/OUT [track file]
REPEAT WHILE (tracks remain in the track file)
| refer to this track as the "inrange" track
| REPEAT WHILE (tracks remain in the track file)
| | refer to this track as the "outrange" track
| | IF (the two ranges differ by at most 0.082 nmi).AND.(the range rates
| | |  differ by at most 8.9 kts)
| | | THENIF (the altitudes differ by at most 100 feet).OR.(the altitude
| | |      |  rates differ by at most 10 ft/s).AND.(both tracks were
| | |      |  formed during the same scan)
| | |      | THENIF (one track is established, and the other is not)
| | |      |      | THEN delete the non-established track
RETURN
END     MERGE_TRACKS
```

7.6 Image Rejection

```
TASK    IMAGE_REJECTION
        IN/OUT [track file]
REPEAT WHILE (tracks remain in the track file)
| IF (track altitude is unknown)
| | THEN RALT=own altitude
| | ELSE RALT=track altitude
| A=(track range)*(track rangerate)-
|   ((own alt.)-(RALT))*((own alt.rate)-(track alt.rate))
| B=(own alt.rate)-(track alt.rate)
| C=(track range)*(track range)-
|   ((own alt.)-(RALT))*((own alt.)-(RALT))
| RANGE=track range
| RANGERATE=track range rate
| ALTITUDE=track altitude
| REPEAT WHILE (tracks remain in the track file)
| | IF ((track range is within 2 nmi. of RANGE) AND ((track altit. is unknown)
| |     OR (within 200 feet of ALTITUDE))
| | | THEN RREFDOT2=(1/track range)*
| |                 square root of[(track range)*(track range)-C)]*B+A
| |      D=2*(track range)-RANGE
| |      RREFDOT1=0.5*[RANGERATE+(1/D)*
| |                 square root of[D*D-C]*B+A
| |      IF (track range rate is within 40 knots of RREFDOT1 or RREFDOT2)
| |      | THEN mark the track as a reflection
RETURN
END IMAGE_REJECTION
```

7.7 Establish Tracks

```
TASK ESTABLISH_TRACKS
     IN/OUT [track file]
REPEAT WHILE (tracks remain in the track file)
| IF ((track's AGE).GE.4).AND.(track's COASTS).EQ.0).AND.(the track is not
| |    marked as a reflection)
| | THEN establish the track
RETURN
END ESTABLISH_TRACKS
```

8.0 NAR ALGORITHM PSEUDO CODE DESCRIPTION

8.1 NAR Algorithm Executive

The NAR algorithm is executed once per second, and consists of calls to the following tasks:

PROGRAM: SURVEILLANCE OF NON-ALTITUDE-REPORTING-AIRCRAFT

CALL: FORMATION_OF_ZERO_CODE_REPLY_BUFFER

This task identifies replies (from different whisper-shout interrogations) that were probably received from a single aircraft. A COMPOSITE REPLY is formed by combining the attributes of such replies. The replies are discarded. The COMPOSITE REPLIES are then combined with the remaining replies. The resulting ZERO-BUFFER is used by the TRACK_UPDATE, and NEW_TRACK_FORMATION tasks.

CALL: TRACK_UPDATE

This task predicts ahead the range and bearing of eack track by one scan. A window in range is set up about each range prediction. The window size depends on various attributes of the track. The windows are not allowed to overlap. The best reply from the ZERO-BUFFER that lies within the window is then selected. The criterion for "best" takes into account reply-to-reply and reply-to-track differences in range and bearing. The track is then updated with the selected reply. Replies used to update tracks are deleted. Tracks not updated for several consecutive seconds are deleted.

CALL: FORMATION_OF_NEW_TRACKS

This task uses the ZERO-BUFFERS from the last three seconds. Triplets of replies that lie in a straight line in range are used to form an initial track file in range and bearing.

CALL: MERGE_TRACKS

This function deletes tracks that appear to be redundant.

CALL: IMAGE_REJECTION

This task deletes tracks that appear to be caused by multipath.

CALL: ESTABLISH_TRACKS

The term "established" means that the track is mature enough to pass to the collision avoidance function. The criteria is a function of the track's age and its history of updates.

CALL: ROTATE_ZERO_BUFFERS

Three ZERO-BUFFERS are used; the current buffer, one from one scan ago, and one from two scans ago. The latter is not needed anymore after the ESTABLISH_TRACK task is complete, so it is "rotated" back to "current" status for use by the FORMATION_OF_THE_ZERO-BUFFER task on the next scan.

8.2 Formation of the "Zero-Code-Reply-Buffer"

```
TASK  FORMATION_OF_ZERO_CODE_REPLY_BUFFER
      IN [REPLY-BUFFER, contains all replies received during the scan]
      OUT [ZERO-BUFFER, contains replies for TRACK_UPDATE and
                        FORMATION_OF_NEW_TRACKS]
place all zero-code replies found in the REPLY-BUFFER into the ZERO-BUFFER, in
|                                                      increasing range order
REPEAT WHILE (replies remain in ZERO-BUFFER)
| empty temporary buffer
| put next reply into the temporary buffer
| designate all beams as "have not been noted"
| KEYRANGE = range of the reply
| REPEAT WHILE (replies remain in the ZERO-BUFFER)
| | IF ((next reply's range-KEYRANGE).LT.(0.5*σ(range))
| | | THEN put reply in temporary buffer
| REPEAT WHILE (replies remain in temporary buffer)
| | IF (this is the first reply)
| | | THEN note the reply's beam, and associate with it:
| | |      a. RIP =  reply's interrogation power
| | |      b. RSP =  reply's suppression power
| | |      c. BEAM'S POWER SUM = (RIP + RSP)/2
| | |      d  BEAM'S NUMBER OF REPLIES = 1
| | |      KEYBEARING = bearing of the reply
| | |      NUMBERREPLIES = 1
| | | ELSEIF (|reply bearing-KEYBEARING|.LE.2σ(θ))
| | |        | THENIF (reply's beam has been noted before)
| | |        | | THEN CALL TESTPOWERS
| | |        | |      IF (AGREE.EQ.TRUE)
| | |        | |      | THEN add POWERSUMUPDATE to BEAM'S POWER SUM
| | |        | |      |      increment BEAM'S NUMBER OF REPLIES
| | |        | |      |      increment NUMBERREPLIES
| | |        | |      | ELSE delete reply from temporary buffer
| | |        | | ELSE note the reply's beam, and associate with it:
| | |        | |      a. RIP =  reply's interrogation power
| | |        | |      b. RSP =  reply's suppression power
| | |        | |      c. BEAM'S POWER SUM = (RIP + RSP)/2
| | |        | |      d  BEAM'S NUMBER OF REPLIES = 1
| | |        | |      increment NUMBERREPLIES
| | |        | ELSE delete reply from temporary buffer
| IF (NUMBERREPLIES.GT.1)
| | THEN compute the average range and bearing of replies in temporary buffer
|        for each beam;
|      divide BEAM'S POWER SUM by BEAM'S NUMBER OF REPLIES form a
|        COMPOSITE REPLY with the above attributes;
|      delete the counterpart of each reply that remains in the temporary
|        buffer from the ZERO-BUFFER;
merge, in increasing range order, all COMPOSITE REPLIES with the replies
                                                    remaining in the ZERO-BUFFER
ENDTASK FORMATION_OF_ZERO_CODE_REPLY_BUFFER
```

```
FUNCTION  TESTPOWERS
  IN [RIP of beam on which current reply was received]
  IN [RSP of beam on which current reply was received]
  OUT [POWERSUMUPDATE]
  OUT [AGREE]
AGREE=TRUE
PF = RIP  (the interogation Power of the First reply from this beam)
SF = RSP  (the Suppression power of the First reply from this beam)
PC = interrogation Power of Candidate reply from the beam of interest
SC = Suppression power of Candidate reply from the beam of interest
IF (SC.GT.SF)
  THENIF (PC.GT.SF)
        THENIF (PC-SF.LT.10dB)
              THEN compute POWERSUMUPDATE = (PC+SF)/2
              ELSE AGREE=FALSE
        ELSEIF (PC.EQ.SF)
              THEN compute (PC+SF)/2 for updating beam's power sum
              ELSEIF (PC.GT.PF)
                    THEN   POWERSUMUPDATE = (PC+SF)/2
                    ELSE   POWERSUMUPDATE = (PF+SF)/2
  ELSEIF (SC.EQ.SF)
        THENIF (PC.GE.PF)
              THEN  POWERSUMUPDATE = (PC+SC)/2
              ELSE  POWERSUMUPDATE = (PF+SF)/2
        ELSEIF (PC.GE.PF)
              THEN  POWERSUMUPDATE = (PC+SC)/2 for updating beam's power sum
              ELSEIF ((SC-PF).GE.-D)
                    THEN  POWERSUMUPDATE = (PF+SC)/2
                    ELSE AGREE=FALSE
RETURN
END   TESTPOWERS
```

8.3 Track Update

```
TASK  TRACK UPDATE
      IN/OUT [ZERO-BUFFER, track file]
REPEAT WHILE  (tracks remain in track file)
 | DT          = time elapsed since last scan
 | AGE         = AGE + 1
 | RANGESQ     = RANGESQ     + RANGESQRATE * DT   + RANGESQACC * DT^2/2
 | RANGESQRATE = RANGESQRATE +  RANGESQACC * DT
 | RANGE       = square root of RANGESQ
 | X           = X           +        XDOT * DT
 | Y           = Y           +        YDOT * DT
 | BEARING   = ARCTANGENT (Y/X)
REPEAT WHILE  (tracks remain in track file)
 | IF (TRACKAGE.LT.7)
 | | THEN RANGEWINDOW=[(7-TRACKAGE)+5+UPDOWNCOASTS/2+BIAS/2]*0.12  (range)
 | | ELSE RANGEWINDOW=              [5+UPDOWNCOASTS/2+BIAS/2]*0.12  (range)
 | IF (RANGEWINDOW.GT.15)*0.12  (range)
 | | THEN RANGEWINDOW=15*0.12  (range)
 | IF (this is the first track)
 | | THEN INWINDOW = 0
 | | ELSE INWINDOW  = the average RANGE of this, and previous track
 | IF  (INWINDOW.LT.(RANGE - RANGEWINDOW)
 | | THEN  INWINDOW = RANGE - RANGEWINDOW
 | IF (this is the last track)
 | | THEN OUTWINDOW = infinity
 | | ELSE OUTWINDOW = the average RANGE of this, and the next track
 | IF  (OUTWINDOW.GT.(RANGE + RANGEWINDOW)
 | | THEN OUTWINDOW = RANGE + RANGEWINDOW
 |       REPEAT WHILE (replies remain in the ZERO-BUFFER)
 |         IF (reply's range is between INWINDOW and OUTWINDOW)
 |         | THEN mark reply as INELIGIBLE for use in NEW TRACK FORMATION
 |                IF (reply's bearing is within 3 (bearing) of track BEARING
 |                | THENIF (no reply has been selected yet)
 |                |      | THEN select this reply as the correlating reply
 |                |      | ELSEIF (the bearing of this reply is more than
 |                |      |          1.5 (bearing) from that of the selected reply)
 |                |      |      THENIF (this reply is closer to RANGE than is
 |                |      |           the selected reply)
 |                |      |           | THEN "deselect" the previously selected reply
 |                |      |           |      and select this reply instead)
 | IF (a reply was selected)
 | | THEN CALL: TRACK SMOOTHING
 |       delete the selected reply from the ZERO-BUFFER
 |       UPDATES = UPDATES + 1
 |       UPDOWNCOASTS = UPDOWNCOASTS - 1 (lower bounded by zero)
 |       COASTS = 0
 | ELSE COASTS = COASTS + 1
 |       UPDOWNCOASTS = UPDOWNCOASTS + 1
 |       IF ((COASTS.GT.6).OR.((COASTS.GT.3).AND.(RANGEACC.LT.0.0)))
 |       | THEN delete the track
arrange the track file in increasing range order
END   TRACK UPDATE
```

8.3 Track Update (continued)

```
FUNCTION TRACK_SMOOTHING
         IN/OUT [selected reply, trackfile]
RESIDUAL = (reply's range squared) - (track's RANGESQ)
BIAS = BIAS * 0.5 + RESIDUAL * 0.5
ALPHA = ALPHA * 0.90 + 0.44  *  0.10
BETA  = BETA  * 0.79 + 0.114 *  0.21
GAMMA = GAMMA * 0.70 + 0.012 *  0.30
IF (track was coasted on the last scan)
| THEN ALPHA = ALPHA * 0.750 +  0.250
       BETA  = BETA  * 0.984 +  0.016
       GAMMA = GAMMA * 0.998 +  0.002
IF (BIAS.GT.(1.5 (range)).AND.(ALPHA.LT.0.58))
| THEN ALPHA = 0.58
       BETA  = 0.22
       GAMMA = 0.035
RANGESQ     = RANGESQ   + RESIDUAL * ALPHA
RANGESQRATE = RANGESQRATE + RESIDAUL * BETA
RANGESQACC  = RANGESQACC  + RESIDUAL * GAMMA
RANGE       = square root of RANGESQ
BEARTEST = 22.5 * 2 EXPONENT(BEARCOAST + 1)
IF (absolute value of reply's bearing - BEARING is less than BEARTEST)
| THEN XRESIDUAL = (reply's range * cosine of reply's bearing) - (X)
       YRESIDUAL = (reply's range *   sine of reply's bearing) - (Y)
       IF (track age.LE.8)
       | THEN INDEX = track AGE
       | ELSE INDEX = 8
       XYALPHA = XYALPHA(INDEX)
       XYBETA  = XYBETA(INDEX)
       X    = X    + XYALPHA * XRESIDUAL
       XDOT = XDOT + XYBETA  * XRESIDUAL
       Y    = Y    + XYALPHA * YRESIDUAL
       YDOT = YDOT + XYBETA  * YRESIDUAL
       BEARING = ACRTANGENT (Y/X)
RETURN
END TRACK_SMOOTHING
```

INDEX	XYALPHA	XYBETA
3	0.700	0.300
4	0.600	0.200
5	0.524	0.143
6	0.464	0.107
7	0.417	0.083
8	0.400	0.067

8.4 Formation of New Tracks

```
TASK   FORMATION_OF_NEW_TRACKS
       IN/OUT [ZERO-BUFFER, trackfile]
REPEAT WHILE (ELEGIBLE replies remain in the 1 second old ZERO-BUFFER)
|  RANGE1    = range of the next reply
|  BEARING1 = bearing of the next reply
|  HIGHWINDOW = RANGE1 + 2100 feet
|  LOWWINDOW  = RANGE1 - 2100 feet
|  REPEAT WHILE (ELIGIBLE replies remain in the 2 sec. old ZERO-BUFFER)
|    RANGE0 = range of the next reply
|    BEARING0 = bearing of the next reply
|    IF ((RANGE0.GT.LOWWINDOW).AND.(RANGE0.LT.HIGHWINDOW))
|    |  THEN CENTERWINDOW = RANGE1 + (RANGE1-RANGE2)/2
|           OUTWINDOW     = CENTERWINDOW + 3 (range)
|           INWINDOW      = CENTERWINDOW - 3 (range)
|           REPEAT WHILE (ELIGIBLE replies remain in the current ZERO-BUFFER)
|             RANGE3 = range of the next reply
|             BEARING3 = bearing of the next reply
|             IF ((RANGE3.GT.INWINDOW).AND.(RANGE3.LT.OUTWINDOW)
|              |THENIF (|BEARING0-BEARING1|.LT.2 (bearing))
|                 |  THENIF (|BEARING0-BEARING2|.LT.2 (bearing))
|                       THENIF (|BEARING2-BEARING3|.LT.2 (bearing))
|                          |  THEN RANGE    = (RANGE0)
|                                   RANGESQ0 = (RANGE0)^2
|                                   RANGESQ1 = (RANGE1)^2
|                                   RANGESQ2 = (RANGE2)^2
|                                   RANGESQ     = RANGESQ2
|                                   RANGESQRATE=(3*RANGESQ2-4*RANGESQ1+RANGESQ0)/2
|                                   RANGESQACC =(RANGESQ2-2*RANGESQ1+RANGESQ0)
|                                   AGE         = 3
|                                   UPDATES     = 3
|                                   BIAS        = 0
|                                   ALPHA       = 1.0067
|                                   BETA        = 1.2355
|                                   GAMMA       = 0.7091
|                                   CALL: INITIALIZE TRACK BEARING
merge the new tracks into the track file, in increasing range order
END    FORMATION_OF_NEW_TRACKS
```

```
FUNCTION  INITIALIZE_TRACK_BEARING
          IN/OUT [ZERO-BUFFERS, track fiile]
IF (BEARING0 is valid)
 THEN X0 = RANGE0 * COSINE(BEARING0)
      Y0 = RANGE0 *   SINE(BEARING0)
      IF (BEARING1 is valid)
       THEN  X1 = RANGE1 * COSINE(BEARING1)
             Y1 = RANGE1 *   SINE(BEARING1)
             IF (BEARING2 is valid)
              THEN X2 = RANGE2 * COSINE(BEARING2)
                   Y2 = RANGE2 *   SINE(BEARING2)
                    X = 5/6 * X2 + 2/6 * X1 - 1/6 * X0  XDOT = X2/2 - X0/2
                    Y = 5/6 * Y2 + 2/6 * Y1 - 1/6 * Y0  YDOT = Y2/2 - Y0/2
              ELSE  X  = 2 * X1 - X0                    XDOT = X1 - X0
                    Y  = 2 * Y1 - Y0                    YDOT = Y1 - Y0
       ELSEIF (BEARING2 is valid)
              THEN X2 = RANGE2 * COSINE(BEARING2)
                   Y2 = RANGE2 *   SINE(BEARING2)
                    X = 5/6 * X2 + 1/6 *X0          XDOT = X2/2 - X0/2
                    Y = 5/6 * Y2 + 1/6 *Y0          YDOT = Y2/2 - Y0/2
              ELSE  X = X0                          XDOT = 0
                    Y = Y0                          YDOT = 0
 ELSEIF (BEARING 1 is valid)
      THEN X1 = RANGE1 * COSINE(BEARING1)
           Y1 = RANGE1 *   SINE(BEARING1)
           IF (BEARING2 is valid)
            THEN X2 = RANGE2 * COSINE(BEARING2)
                 Y2 = RANGE2 *   SINE(BEARING2)
                  X = X2                            XDOT = X2 - X1
                  Y = Y2                            YDOT = Y2 - Y1
             ELSE X = X1                            XDOT = 0
                  Y = Y1                            YDOT = 0
       ELSEIF (BEARING2 is valid)
            THEN X2 = RANGE2 * COSINE(BEARING2)
                 Y2 = RANGE2 *   SINE(BEARING2)
                 X = X2                             XDOT = 0
                 Y = Y2                             YDOT = 0
            ELSE X = 0                              XDOT = 0
                 Y = 0                              YDOT = 0
RETURN
END   INITIALIZE_TRACK_BEARING
```

8.5 Merge Tracks

```
TASK    MERGE_TRACKS
        IN/OUT [track file]
REPEAT WHILE (tracks remain in the track file)
|  refer to this track as the "inrange" track
|  RANGEIN = RANGE of the inrange track
|  REPEAT WHILE (tracks remain in the track file)
|    refer to this track as the "outrange" track
|    RANGEOUT = RANGE of the outrange track
|    IF ((RANGEOUT-RANGEIN).LT.500 feet)
|    |  THENIF (neither track is established)
|    |     |  THENIF (the tracks have equal UPDATES)
|    |     |     |  THENIF ((inrange track COASTS).EQ.0)
|    |     |     |     |  THEN delete the outrange track
|    |     |     |  ELSEIF (the track with greater UPDATES has COASTS.EQ.0)
|    |     |     |     |  THEN delete the track with lesser UPDATES
|    |     |  ELSEIF (only one track is not established)
|    |     |     |  THEN delete the non-established track
|
RETURN
END     MERGE_TRACKS
```

8.6 Image Rejection

The IMAGE REJECTION FUNCTION OPERATES ON THE COMBINED SET OF AR AND NAR TRACKS, AND IS DESCRIBED IN SECTION 7.7.

8.7 Establish Tracks

```
TASK ESTABLISH_TRACKS
     IN/OUT [track file]
REPEAT WHILE (tracks remain in the track file)
|  IF ((track's AGE).GE.4).AND.(track's COASTS).EQ.0)
|  |  THEN establish the track
RETURN
END ESTABLISH_TRACKS
```

REFERENCES

1) Radio Technical Commision for Aeronautics, "Minimum Operational Performance Standards for Traffic Alert and Collision Avoidance System (TCAS) Airborne Equipment," RTCA/DO-185, September 1983.

2) N.A. Spencer et al., "Assessment of the Performance of an Active ATCRBS Mode for Beacon Collision Avoidance," FAA-RD-77-151, (MITRE Corp. MTR-7645), October 28, 1977.

3) W.H. Harman and R.S. Kennedy, "TCAS II: Design and Validation of the High-Traffic-Density Surveillance Subsystem," Project Report ATC-126, Lincoln Laboratory, M.I.T., to be published, DOT/FAA/PM-84/5.