

General Disclaimer

One or more of the Following Statements may affect this Document

- This document has been reproduced from the best copy furnished by the organizational source. It is being released in the interest of making available as much information as possible.
- This document may contain data, which exceeds the sheet parameters. It was furnished in this condition by the organizational source and is the best copy available.
- This document may contain tone-on-tone or color graphs, charts and/or pictures, which have been reproduced in black and white.
- This document is paginated as submitted by the original source.
- Portions of this document are not fully legible due to the historical nature of some of the material. However, it is the best reproduction available from the original submission.

final report

ERROR STUDY For
SS-1090 ATM SENSOR SYSTEM

FACILITY FORM 602

N71-14098
(ACCESSION NUMBER)

159
(PAGES)

CR-102957
(NASA CR OR TMX OR AD NUMBER)

63
(THRU)

30
(CODE)

30
(CATEGORY)



BALL BROTHERS RESEARCH CORPORATION
SUBSIDIARY OF BALL CORPORATION
BOULDER, COLORADO

CR 102957

FINAL REPORT

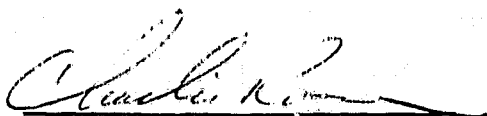
ERROR STUDY
for
SS1090 ATM SENSOR SYSTEM

Prepared for:
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
George C. Marshall Space Flight Center
Huntsville, Alabama 35812

Contract # NAS8-26269

Prepared by:
Attitude Sensors Section
Ball Brothers Research Corporation
Boulder, Colorado 80302

12 November 1970


C. E. Rose (Author)
Attitude Sensor Design


J. H. Spotts (Approval)
Project Manager
Attitude Sensing Systems



BALL BROTHERS RESEARCH CORPORATION
SUBSIDIARY OF BALL CORPORATION
BOULDER, COLORADO



F70-09

SUMMARY

An analytical study has been conducted to determine the effects of atmospheric refraction and attenuation, and earth albedo on the ATM acquisition sun sensor during certain modes of operation. This study involves no new technology.

At the beginning of the project, a literature search was conducted to determine if information gleaned from recent earth orbit experiments showed that an updating of the various mathematical models used in the analysis was necessary. The result of this search was a reconfirmation of the validity of the previous assumptions.

The theoretical background and mathematical models are developed in Section 2.

The computer implementation of the albedo equations is presented in Section 3. It is not necessary to read this section to understand the remainder of the report.

The angular response characteristics of acquisition sun sensor¹ number 007 were measured as part of the albedo mathematical model. This effort is documented in Section 4.

The results of the study are presented in Section 5. In essence, they are as follows:

- The worst pointing error attributable to earth albedo in the sun pointing mode is 34 arc-seconds.

¹Ref.: "Apollo Telescope Mount Acquisition Sun Sensor, Operational Requirements and Descriptions," NASA/MSFC Astrionics Laboratory, 50M23013, 24 February 1970.



F70-09

- Misalignments due to atmospheric refraction can reach one degree, but presumably the ATM fine sun sensor will be affected to the same extent. At this point the sunlight is attenuated 94%.
- In the event that a random search is performed to reacquire the sun, the acquisition sun sensor will generate no more than ± 1.2 volts (10% of maximum output) as it scans across the earth. The output of the sun presence detector will not exceed 1.5% of its maximum value - far below the 50% point at which trigger levels have been set.



F70-09

CONTENTS

Section		Page
	SUMMARY	iii
	ILLUSTRATIONS	vii
1	INTRODUCTION	1-1
2	THEORY	2-1
	2.1 Analysis of Albedo	2-1
	2.2 Problem Geometry	2-1
	2.3 Refraction Analysis	2-15
	2.4 Attenuation	2-22
3	COMPUTER IMPLEMENTATION	3-1
	3.1 Computer Used	3-1
	3.2 Features of the Analysis System	3-2
	3.3 Program Usage	3-9
4	MEASUREMENT OF SS1090 RESPONSE CHARACTERISTICS	4-1
5	RESULTS	5-1
	5.1 Albedo Error Near Terminator in Sun Pointing Mode	5-1
	5.2 Refraction and Attenuation Effects in Sun Pointing Mode	5-4
	5.3 Sensor Output Due to Earth Albedo Alone	5-7
	5.4 Sun Presence Detector Output Due to Albedo Alone	5-9
Appendix		Page
A	MATHEMATICAL NOTATION	A-1
B	PROGRAM LISTINGS	B-1
C	TYPICAL OUTPUT SHEETS	C-1
D	TWO-DIMENSIONAL INTERPOLATION	D-1
E	A NOTE ON LAMBERTIAN RADIATORS	E-1



F70-09

ILLUSTRATIONS

Figure		Page
2-1a	General Problem Geometry	2-3
2-1b	Detailed Problem Geometry	2-4
2-2	Definition of Sensor Illumination Entrance Angles	2-7
2-3	Atmospheric Refraction Geometry	2-19
4-1	SS1090 Test Fixture Light Shield	4-3
4-2	SS1090 Test Fixture Light Shield	4-4
4-3	SS1090 Test Fixture Elevation Axis Protractor and Data Acquisition Equipment	4-5
4-4	SS1090 Test Fixture: Operational Configuration, with Shroud	4-6
4-5	Sensor Response Versus Sensitive Axis Rotation	4-7
4-6	Isometric View of Sensor Response Function	4-8
5-1	Albedo-Induced Pointing Error of ATM Acquisition Sun Sensor Versus Angle ξ Between Terminator and Satellite Radius Vector	5-2
5-2	Pointing Error and Relative Response Due to Atmospheric Refraction and Attenuation Versus Angle ξ Between Terminator and Satellite Radius Vector	5-5
5-3	Sensor Output Due to Albedo Alone Versus Null Axis-Solar Vector Angle α (albedo factor = 0.6)	5-10
5-4	Geometry for Approximate Albedo Analysis	5-12
5-5	Sun Presence Detector Angular Response Characteristic	5-14
D-1	Two-Dimensional Interpolating Surface	D-6



F70-09

Section 1 INTRODUCTION

Part of the Apollo Manned Spacecraft Program involves the operation in space of several experiments contained in the Apollo Telescope Mount (ATM). The experiments are housed in the ATM canister, which is connected to the rest of the ATM structure by means of a large roll bearing and orthogonal limited-rotation pitch and yaw gimbals. The ATM structure is, in turn, "hard mounted" to the S-IVB workshop. The ATM structure with its connecting members is commonly referred to as the "rack" to differentiate between it and the gimballed experiment canister.

The pitch and yaw attitude of the ATM canister with respect to the sun line is sensed by a fine sun sensor located on the canister. The attitude is nominally controlled to an accuracy of a few arc-seconds. The pitch and yaw attitude of the rack with respect to the sun line is sensed by the BBRC-built SS-1090 acquisition sun sensor located on the rack. It provides signals to the ATM digital computer, which generates error signals for the control moment gyros and gas jets which control the attitude of the rack to an accuracy of a few arc minutes.

The ATM canister is mechanically limited to excursions in azimuth and elevation of only $\pm 2^\circ$ with respect to the rack. Thus, it is important that accumulated relative errors between the rack and the ATM canister control systems do not exceed this limit.

The current study program came to pass, partly, as a result of this restriction. It is obviously desirable to operate the experiments over as large a portion of each orbit as possible. During spacecraft dawn and dusk, however, there will be periods when the



F70-09

acquisition sensor, with its wide field of view, will be affected by earth albedo while the fine sun sensor will not. There was some concern that the albedo-induced misalignment of the rack might drive the ATM canister against its stops. If this indeed were a problem, where in orbit would it occur? If it were not a problem, then the experiments could take data until the murky atmosphere or the earth itself shut off the sun. Again the question, Where in orbit would this occur?

BBRC undertook the task of performing an analytical study to answer these questions. In doing so, use could be made of an existing BBRC computer program to compute atmospheric refraction of light rays and their attendant attenuation. The albedo problem, however, had to be attacked from scratch: a computer program had to be developed. The greater portion of this report is devoted to documenting that effort.

The program was made quite general and can handle any kind of sensor in any orbital position and attitude. It was therefore able to analyze, by simply altering a few parameters, another problem associated with the overall mission; if the rack loses its alignment for any reason and a tumbling search mode is instituted to find and reacquire the sun, is there a possibility of getting such a large signal as the sensor field of view sweeps across the brightly illuminated earth that it might appear that the sensor had swept by the sun? This would present a confusing situation.

These were the questions posed at the beginning of the program. The answers and the methods used to obtain them are presented in this report.



F70-09

Section 2 THEORY

The notation used in developing the mathematical model is concise, consistent, and an abomination. It is suggested that you go through Appendix A thoroughly, now, if you are serious about reading this section.

2.1 ANALYSIS OF ALBEDO

2.1.1 Assumptions

The earth is assumed to be spherical, of unit radius, and illuminated by an infinitely distant source. Thus, the terminator appears as a sharp, distinct line.

The atmosphere is treated as a Lambertian radiator; that is, the radiation emitted from each differential element of area varies as the cosine of the angle between the light ray and the area's normal vector. Furthermore, the spectral contents of the radiated and incident energies are assumed to be identical.

2.2 PROBLEM GEOMETRY

The sensor is placed a distance h above the surface of the earth (assumed spherical). The vector from the center of the earth to the sensor is called Q and the angle between Q and the earth's terminator is called ξ . If the sensor is over the "day side" of the earth, ξ is positive; if over the "night side," it is negative. Thus, as the satellite moves in its orbit from night into day, ξ goes from negative to positive values and is zero when the vehicle is situated just over the terminator.



F70-09

Note that ξ is not the angle traced out by Q as the satellite traverses its orbit, except in the special case where the orbital plane is perpendicular to the terminator.

The angle ξ is shown pictorially in Fig. 2-1a for several positions of the satellite. All computations were made in terms of ξ . The reason for this parameterization is that the data can be applied to orbits of any inclination.

The basis for the albedo analysis is shown in Fig. 2-1b. The sun lies in the direction of the unit vector S , N is the outward unit normal to dA (the differential element of area), and P^* is the vector from dA to the satellite. S is parallel to E_1^R and the (E_2^R, E_3^R) plane contains the terminator. The vehicle is in the (E_1^R, E_3^R) plane, lies along the vector Q , and is a distance $|Q| = q > 1$ from the center of the earth.

For purposes of this discussion, the terms roll, elevation, and azimuth have been associated with rotations about the vehicle 1, 2, and 3 axes respectively. Furthermore, the two detector pairs on the sun sensor which detect the latter two rotations are called the *elevation* and *azimuth* sensors rather than *pitch* and *yaw* sensors as they are commonly called.

2.2.1 Development of Equations

The earth is centered on the reference coordinate system E^R , which is aligned so the sun lies along E_1^R and the vehicle is in the (E_1^R, E_3^R) plane. This choice of reference allows the solar vector S and the vehicle position vector Q to be written in the simple form

$$S^R = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad Q^R = q \begin{bmatrix} s\xi \\ 0 \\ c\xi \end{bmatrix} \quad (1)$$



F70-09

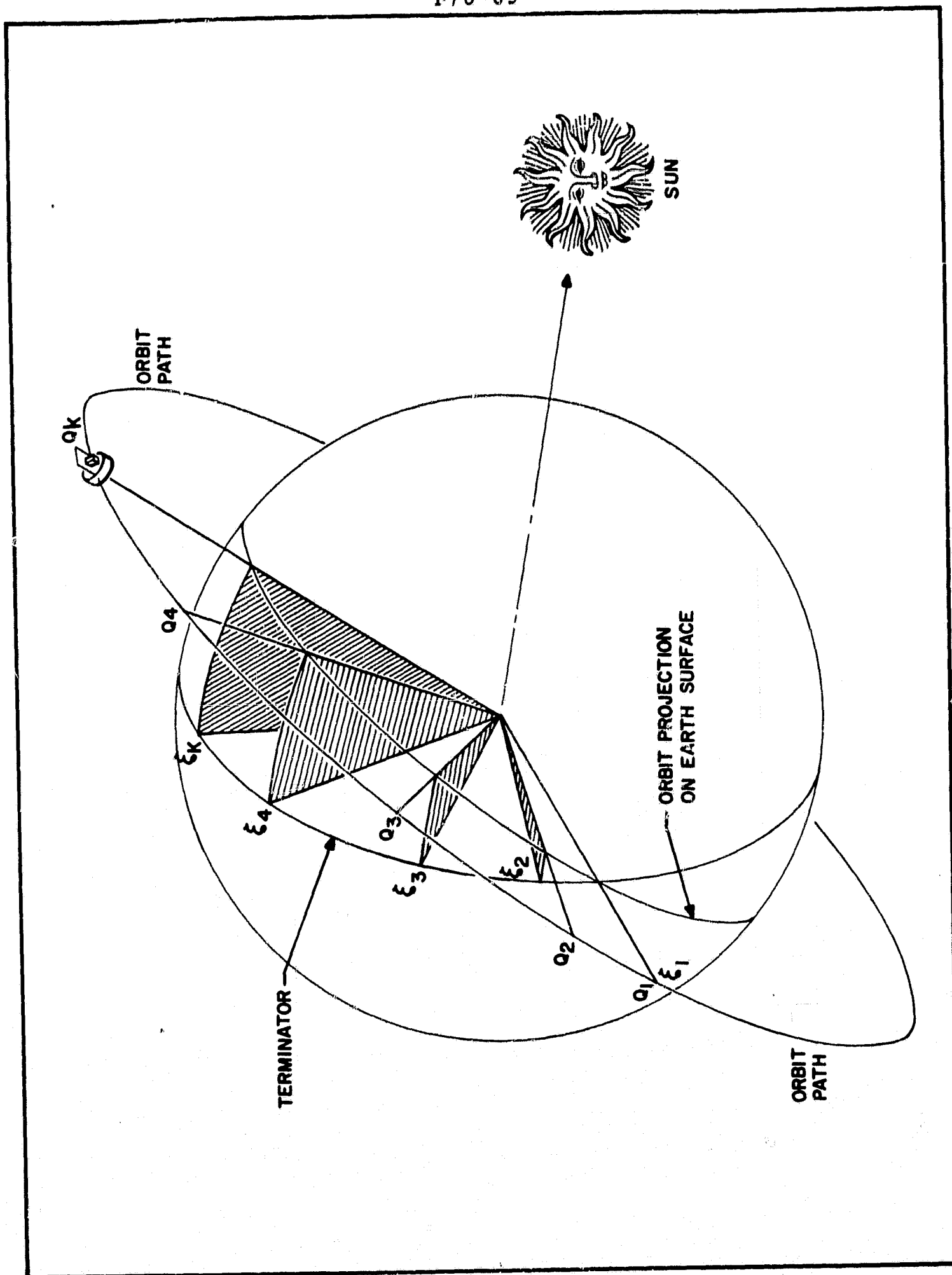


Fig. 2-1a General Problem Geometry



F70-09

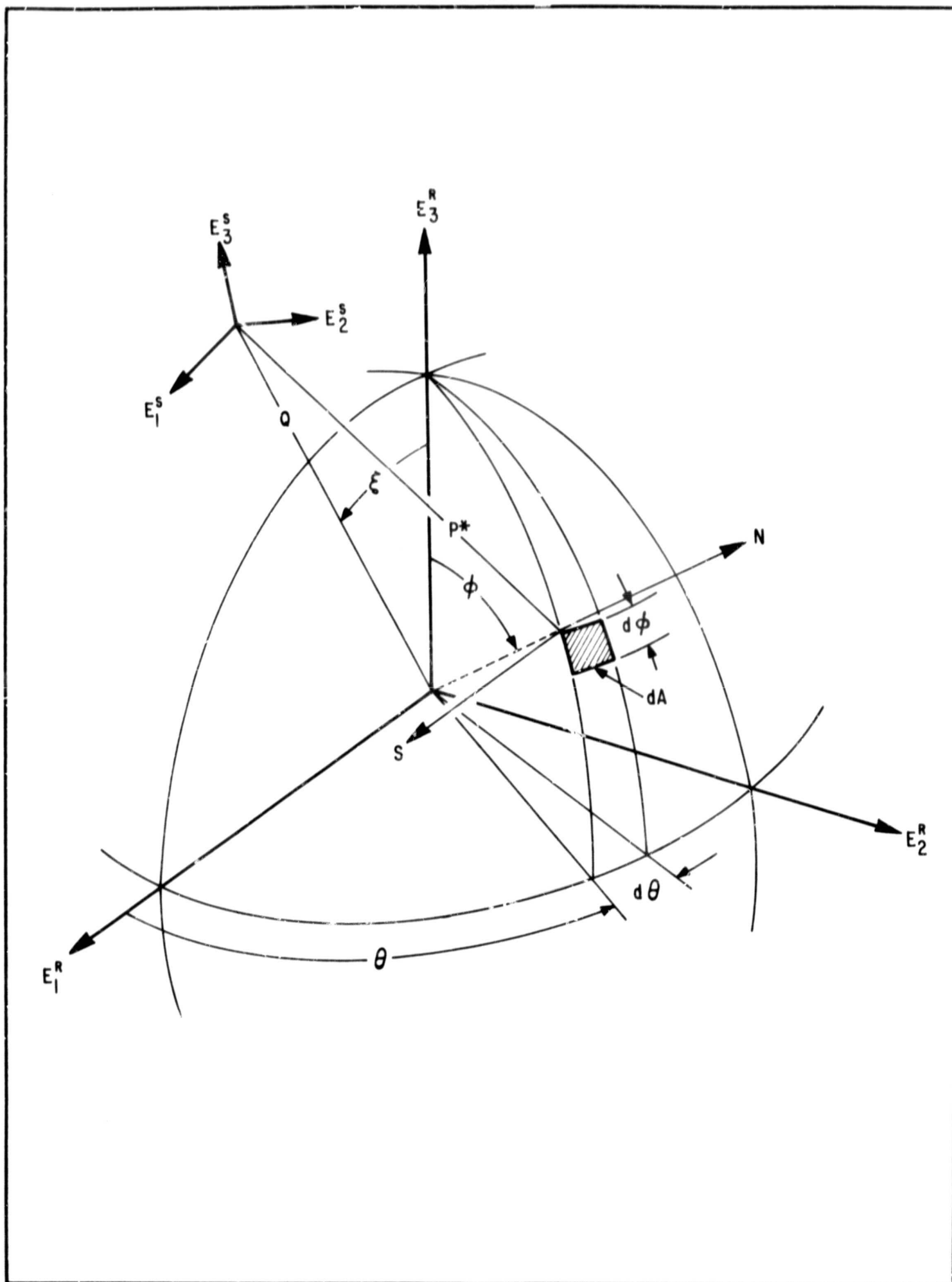


Fig. 2-1b Detailed Problem Geometry



F70-09

Points on the surface of the earth have coordinates $(1, \phi, \theta)$, where ϕ is the polar angle measured from E_3^R and θ is the longitudinal angle measured from E_1^R . Thus, the unit normal N at such a point is

$$N^R = \begin{bmatrix} s\phi c\theta \\ s\phi s\theta \\ c\phi \end{bmatrix} \quad (2)$$

Earth-reflected sunlight emanating from this point and reaching the satellite travels along P^* ($= Q - N$) or

$$P^{*R} = \begin{bmatrix} qs\xi - s\phi c\theta \\ -s\phi s\theta \\ qc\xi - c\phi \end{bmatrix} \quad (3)$$

Sunlight reflected from each of these points which reaches the sensor will produce an output dependent on several quantities:

1. The size of the radiating area dA .
2. The effective intensity of the light incident upon dA , which is the cosine of the angle between S and N or $\langle S, N \rangle$.
3. The cosine of the angle between N and P^* or $\langle N, P^* \rangle / p^*$ (Lambert's law).
4. The albedo factor $a(\phi, \theta)$ (reflectivity efficiency) applicable at the point.
5. The square of the distance between dA and the sensor.



F70-09

6. The angles at which the light from dA enters the sensor.
7. The response of the sensor to the incoming light, which is a function of the on- and off-axis entrance angles σ and ω .

Thus, the output of the sensor v_a due only to albedo is given by

$$v_a = \frac{1}{\pi} \int_{A_i \cap A_v} \frac{v(\sigma, \omega) a(\phi, \theta) \langle S, N \rangle \langle N, P^* \rangle dA}{(p^*)^3} \quad (4)$$

Equation (4) is the well-known "albedo equation," modified only in notation and the fact that the albedo factor and sensor response functions appear within the integral as weighting functions.

The terms σ and ω are the on- and off-axis angles of the light entering the sensor (ref. Fig. 2-2) in sensor coordinates E^S . Expressions for these quantities will be developed later in this section.

The factor $1/\pi$ is a result of the assumed Lambertian radiator law; a heuristic justification for the factor appears in Appendix E.

One of the p^* 's appearing in the denominator of Eq. (4) is associated with the $\langle N, P^* \rangle$ term and is not a misprint.



F70-09

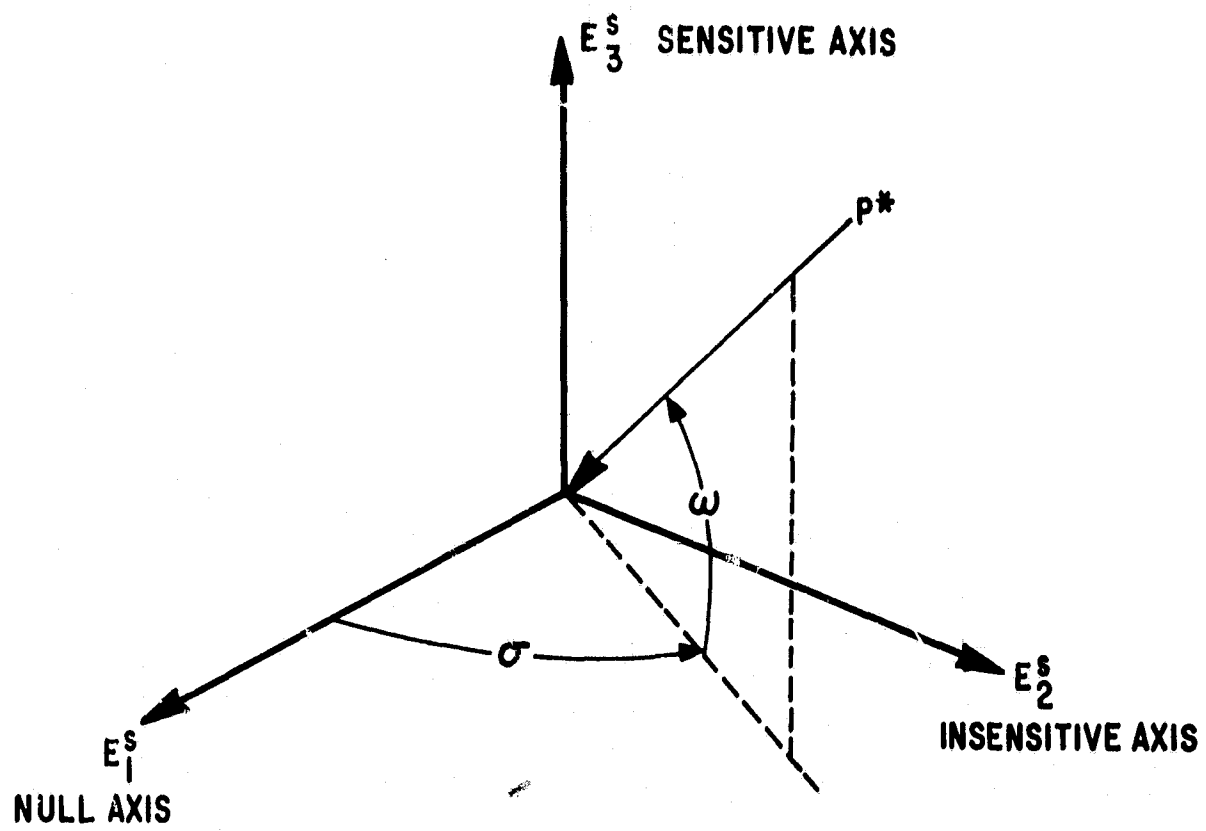


Fig. 2-2 Definition of Sensor Illumination Entrance Angles



F70-09

The area over which the integration is performed is the intersection of two general regions: A_i , the illuminated portion of the earth and A_v , that area of the earth visible from the satellite. Thus,

$$A_i = \{(1, \phi, \theta) \mid \langle N, S \rangle \geq 0\} \quad (5a)$$

$$A_v = \{(1, \phi, \theta) \mid \langle N, P^* \rangle \geq 0\} \quad (5b)$$

These conditions will now be used to derive analytical expressions for the boundaries of $A_i \cap A_v$.

The differential area dA appearing in Eq. (4) is given by

$$dA = s \phi \, d\theta \, d\phi \quad (6)$$

and the computer program which evaluates Eq. (4) integrates in just that order; it steps ϕ between minimum and maximum values and, for a given ϕ , allows θ to range between its bounds. The problem, therefore, is to determine the bounds on ϕ and θ such that

$$\phi_{\min} \leq \phi \leq \phi_{\max} \quad (7)$$

$$\theta_{\min}(\phi) \leq \theta \leq \theta_{\max}(\phi)$$

Condition (5a) gives

$$s \phi \cos \theta \geq 0 \quad (8)$$



F70-09

which implies the double constraint

$$0 \leq \phi < \pi \quad (9a)$$

$$-\pi/2 \leq \theta \leq \pi/2 \quad (9b)$$

Recalling that

$$P^* = Q - N \quad (10)$$

allows condition (5b) to be written as

$$\langle N, (Q - N) \rangle \geq 0 \quad (11)$$

or

$$\langle N, Q \rangle = (s\phi c\theta s\xi + c\phi c\xi)q \geq 1 \quad (12)$$

Due to symmetry, the extreme values of ϕ occur when $\theta = 0$, so Eq. (12) becomes

$$c(\phi - \xi) = c(\xi - \phi) \geq 1/q \quad (13)$$

Taking inverse cosines of both sides of Eq. (13), remembering that this operation reverses the direction of the inequality, gives

$$\xi - \cos^{-1}(1/q) \leq \phi \leq \xi + \cos^{-1}(1/q) \quad (14)$$

Thus,

$$\phi_{\max} \triangleq \xi + \cos^{-1}(1/q) \quad (15a)$$

$$\phi_{\min} \triangleq \xi - \cos^{-1}(1/q) \quad (15b)$$



F70-09

Condition (9a) must still be satisfied, so Eqs. (15a) and (15b) should be modified as follows:

$$\phi_{\max} = \text{MIN} (\xi + \cos^{-1}(1/q), \pi) \quad (16a)$$

$$\phi_{\min} = \text{MAX} (\xi - \cos^{-1}(1/q), 0) \quad (16b)$$

Next, the bounds on θ are developed. Returning to Eq. (12) and assuming that ϕ satisfies Eq. (14), we have

$$c\theta > \frac{1 - c\phi c\xi}{s\phi s\xi} \quad (17)$$

It is desirable to express Eq. (17) in terms of an arcsin function². Rewriting Eq. (17) gives

$$\theta \leq \sin^{-1} \left\{ \frac{(s^2\xi - c^2\phi + 2c\phi c\xi/q - 1/q^2)^{1/2}}{s\phi s\xi} \right\} \triangleq \phi_{\max} \quad (18)$$

Note that θ must also conform to Eq. (9b). Assuming that $\theta = \pm\pi/2$ in Eq. (12) gives

$$c\phi \geq \frac{1}{qc\xi} \quad (19)$$

So,

$$-\pi/2 \leq \theta < \pi/2 \quad \text{for } c\phi \geq \frac{1}{qc\xi} \quad (20a)$$

$$-\theta_{\max} \leq \theta \leq \theta_{\max} \quad \text{for } c\phi > \frac{1}{qc\xi} \quad (20b)$$

²The calculation of $\phi_{\max}$, $\phi_{\min}$, and $\theta_{\max}$ should be performed using $\sin^{-1}$ or $\cos^{-1}$ functions, depending on the arguments of the inverse functions. Optimum sensitivity can be obtained in this way.



F70-09

Thus, Eqs. (16) and (20) define the boundaries of the region $A_i \cap A_v$.

We now turn to the problem of orienting the spacecraft with respect to the earth and determining the angles at which light enters the azimuth and elevation sensors.

The spacecraft (and its coordinate system E^S) is placed in an arbitrary orientation by aligning it initially to E^R and then performing three ordered rotations:

- 1st rotation: ρ degrees about the roll axis.
- 2nd rotation: ϵ degrees about the elevation axis.
- 3rd rotation: α degrees about the azimuth axis.

Thus,

$$E^S = T_3(\alpha) T_2(\epsilon) T_1(\rho) E^R \triangleq TE^R \quad (21)$$

Additional reference frames, E^a and E^e , are associated with the azimuth and elevation sensors respectively and are related to E^S as follows:

$$E^a \equiv E^S \quad (22a)$$

$$E^e = T_1(-\pi/2) E^S \quad (22b)$$

The choice of $T_1(-\pi/2)$ results in $E_3^e = E_2^S$; that is, the elevation sensor's sensitive axis (E_3^e) is aligned to the vehicle elevation axis E_2^S .



F70-09

P^* is first transformed into spacecraft coordinates by means of Eq. (21); that is,

$$P^{*S} = \begin{bmatrix} p_1^S \\ p_2^S \\ p_3^S \end{bmatrix} = TP^{*R} \equiv P^{*a} \quad (23)$$

The on- and off-axis light-entrance angles, σ and ω , can be then obtained by writing P^* as follows (see Fig. 2-2):

$$-P^{*a} = \begin{bmatrix} c\sigma_a c\omega_a \\ s\sigma_a c\omega_a \\ s\omega_a \end{bmatrix} = \begin{bmatrix} -p_1^S \\ -p_2^S \\ -p_3^S \end{bmatrix} \quad (24)$$

so

$$\sigma_a = \tan^{-1} \left\{ \frac{-p_2^S}{-p_3^S} \right\} \quad (25a)$$

$$\omega_a = \sin^{-1} \left\{ -p_3^S \right\} \quad (25b)$$

These are the entrance angles associated with the azimuth sensor. For the elevation sensor, P^* is first transformed into E^e to give

$$P^{*e} = T_1(-\pi/2) P^{*S} \quad (26a)$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} p_1^S \\ p_2^S \\ p_3^S \end{bmatrix} \quad (26b)$$



F70-09

$$= \begin{vmatrix} p_1^S \\ -p_3^S \\ p_2^S \end{vmatrix} \quad (26c)$$

As in Eq. (24), we have

$$-p^*e = \begin{bmatrix} c\sigma_e c\omega_e \\ s\sigma_e c\omega_e \\ s\omega_e \end{bmatrix} = \begin{vmatrix} -p_1^S \\ p_3^S \\ -p_2^S \end{vmatrix} \quad (27)$$

so

$$\sigma_e = \tan^{-1}\{p_3^S / -p_1^S\} \quad (28a)$$

$$\omega_e = \sin^{-1}\{-p_2^S\} \quad (28b)$$

These are the entrance angles associated with the elevation sensor.

One further simplification can be made in the integrand of Eq. (4). The quantity $\langle S, N \rangle$, when evaluated, becomes simply

$$\langle S, N \rangle = s\phi c\theta \quad (29)$$

Equation (4) can now be rewritten in a more explicit form. Remember that the integral will have to be evaluated twice: once for the elevation sensor and once for the azimuth sensor. This is reflected by the subscript j ($= a, e$) appearing in Eq. (30)



F70-09

2.2.2 Synopsis of Equations

The albedo integral to be evaluated is

$$j^v_a = \frac{1}{\pi} \int_{\phi_{\min}}^{\phi_{\max}} \int_{-\theta(\phi)}^{\theta(\phi)} \frac{v(\sigma_j, \omega_j) a(\phi, \theta) \langle N, P^* \rangle \cos^2 \phi \, d\theta \, d\phi}{(p^*)^3} \quad (30)$$

$j = a, e$ (for azimuth, elevation)

where

$$\phi_{\max} = \text{MIN}(\xi + \cos^{-1}\{1/q\}, \pi) \quad (31a)$$

$$\phi_{\min} = \text{MAX}(\xi - \cos^{-1}\{1/q\}, 0) \quad (31b)$$

$$\theta_{\max} = \begin{cases} \sin^{-1} \left\{ \frac{(s^2 \xi - c^2 \phi + 2c\phi c\xi/q - 1/q^2)^{1/2}}{s\phi s\xi} \right\} & \text{for } c\phi < \frac{1}{qc\xi} \\ \pi/2 & \text{for } c\phi \geq \frac{1}{qc\xi} \end{cases} \quad (31c)$$

$$\sigma_a = \tan^{-1} \left\{ \frac{-p_2^S}{-p_3^S} \right\} \quad (31d)$$

$$\omega_a = \sin^{-1} \left\{ -p_3^S \right\} \quad (31e)$$

$$\sigma_e = \tan^{-1} \left\{ \frac{p_3^S}{-p_1^S} \right\} \quad (31f)$$

$$\omega_e = \sin^{-1} \left\{ -p_2^S \right\} \quad (31g)$$



F70-09

2.3 REFRACTION ANALYSIS

2.3.1 Assumptions

The analysis of atmospheric refraction was simplified, but not to the point of producing invalid results in the large, by assuming that (1) the index of refraction of the atmosphere is a function only of radial position; that is, it varies only with height off the earth's surface, and (2) the atmosphere is dispersionless; that is, the index of refraction is not a function of wavelength.

2.3.2 Introductory Mathematics

The following analysis of refraction is based on variational calculus. For those who are not familiar with the subject, these few paragraphs will serve as a brief introduction to the basic ideas.

Variational calculus and differential calculus are related in that they can be applied to solving minimization problems. It is only the mathematical object being minimized which is different.

In the case of the differential calculus, those things being minimized are called *functions*. A function is something which returns a number if one or more other numbers are supplied to it, and the minimization problem in differential calculus is to find what number(s) must be fed into the function of interest to cause it to generate a smallest³ number in return.

³Or for that matter, a largest; the only difference between minimization and maximization problems is the algebraic sign.



F70-09

The means of determining these minimizing numbers is well known. For a function f of the simple independent variable x , one sets the derivative of f equal to zero, thereby obtaining an algebraic equation whose solution is the extremizing number. Thus,

$$\frac{d}{dx} f(x) \stackrel{\text{set}}{=} 0 \quad (32)$$

and the resultant solution will produce an "extremum" - either a maximum or a minimum value for f .

The variational calculus is concerned with a very similar problem. The only difference is that in this case *functionals*, rather than functions, are being minimized. And what is a functional? It is a mathematical object which generates a number when it is given, as input, one or more functions. Thus, functions generate numbers from numbers; functionals generate numbers from functions. An example: Given two points A and B, there are an infinity of functions whose pictorial representations (graphs) pass through A and B. Singling out any of these curves, the length of the line between A and B can be computed.⁴ The formula used to compute this length is a *functional* because the number it generates (arc length) depends on a *function*.

The determination of what function extremizes a given functional is what the calculus of variations is all about, and rules analogous to Eq. (32) have been developed.

⁴I refuse, in this report, to be concerned with the behavior of pathological functions, whose length is a mystery.



F70-09

For instance, many common functionals occur in the form

$$L(y) = \int f\left(y, \frac{dy}{dx}, x\right) dx \quad (33)$$

and in this case, a necessary condition for L to achieve an extreme value is to choose the y which satisfies the following equation [which is the analog of Eq. (32)]:

$$\frac{\partial f}{\partial y} - \frac{d}{dx} \left(\frac{\partial f}{\partial y'} \right) = 0 \quad (34)$$

where $y' = dy/dx$.

The situation here is somewhat more difficult than in the calculus of functions, for whereas Eq. (32) results in algebraic equations which must be solved, Eq. (34) produces differential equations which, almost invariably, are hopelessly nonlinear. Nonetheless, solutions to Eq. (34) can be found (sometimes analytically but probably only numerically), and when this is done, what we have is a good candidate for a function which makes Eq. (33) produce an extreme value. To be sure, sufficiency tests must be made (as in differential calculus) or physical reasoning must be used.

The latter technique was used in the refraction analysis which follows.

2.3.3 Geometry

The assumption that the atmosphere's index of refraction is a function only of the distances above the earth's surface reduces the analytical problem to one of two dimensions. Thus, the refracted ray always lies in a single plane which passes through the center of the earth.



F70-09

In Fig. 2-3, the light ray is depicted by the curve C and a differential element of arc length along it is ds . The vector from the earth's center to ds is R and it is displaced from the reference line by an angle ξ . Thus, the path traced out by the light ray is given as $R(\xi)$.

2.3.4 Development of Equations

Fermat's principle states that a light ray traveling between two points in space will follow that path which results in the transit time being minimized.⁵

Suppose a light ray is propagating through the earth's atmosphere. The index of refraction of the surroundings is a function of position, and the time of propagation is obviously a functional of the path.

The time required to traverse the element of arc-length ds on C is

$$dt = \frac{ds}{v} \quad (35)$$

where v is velocity. Hence, the functional (time) to be minimized is

$$t = \int_C \frac{ds}{v} \quad (36)$$

⁵M. Born and E. Wolf, *Principles of Optics*, Second Revised Edition, MacMillan, New York, 1964, Chapter 3, page 128.



F70-09

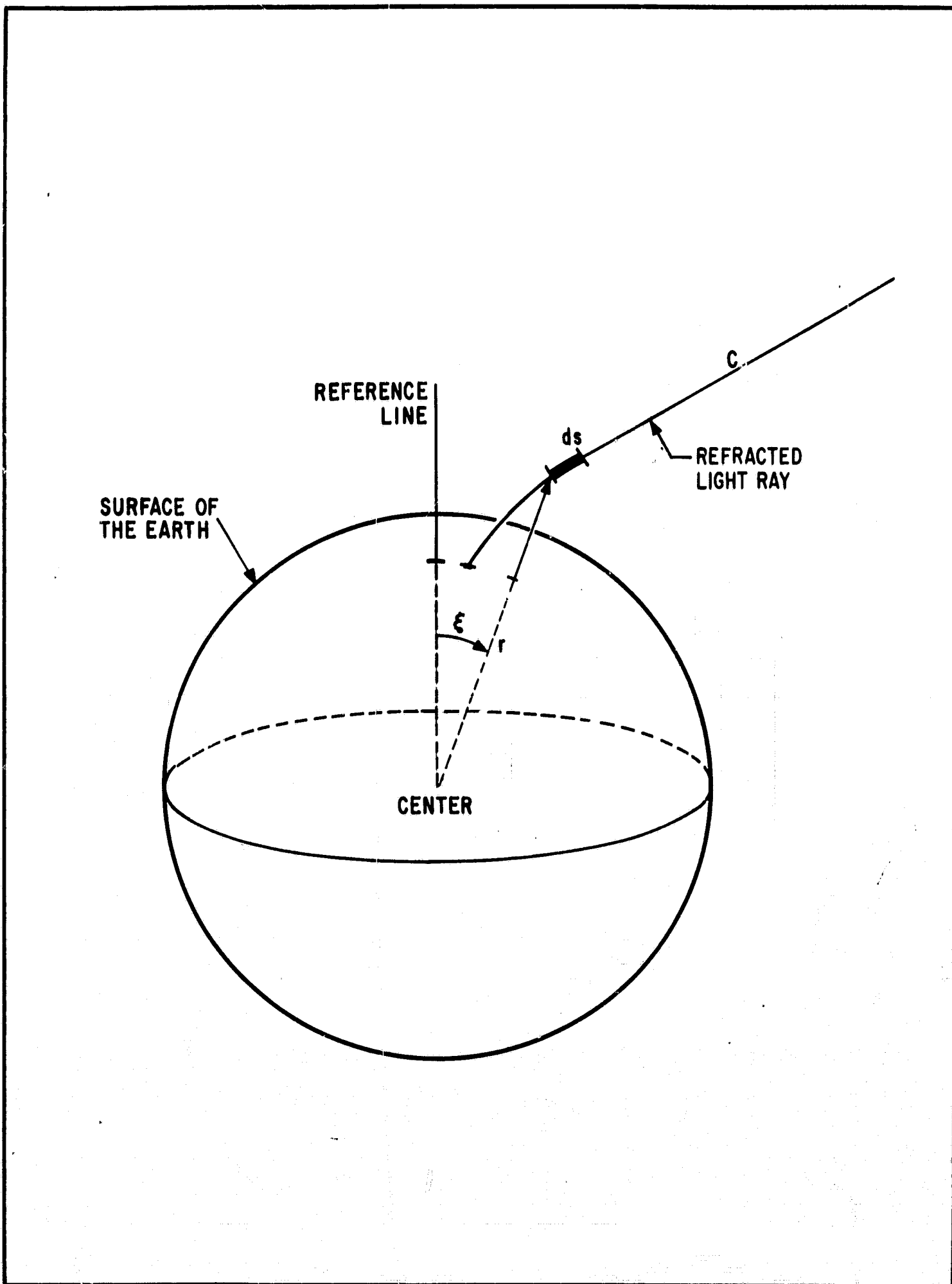


Fig. 2-3 Atmospheric Refraction Geometry



F70-09

Now assume that the index of refraction of the atmosphere is a function of radial distance r ($=|R|$) only. Denoting this by $n(r)$ gives

$$v = \frac{c}{n(r)} \quad (37)$$

where c is the speed of light in vacuum. Equation (36) now becomes

$$t = 1/c \int_C n(r) ds \quad (38)$$

In the coordinate system shown in Fig. 2-3, ds is expressed as

$$(ds)^2 = (r d\xi)^2 + dr^2 \quad (39)$$

so Eq. (38) takes the form

$$t = 1/c \int_C n(r) [r^2 d\xi^2 + dr^2]^{1/2} \quad (40)$$

Factoring out $d\xi$ and using the notation

$$\dot{r} = \frac{dr}{d\xi} \quad (41)$$

permits Eq. (40) to be rewritten as

$$t = 1/c \int_C n(r) \{r^2 + (\dot{r})^2\}^{1/2} d\xi \quad (42)$$

which is a particular case of the general form

$$t = \int f(r, \dot{r}, \xi) d\xi \quad (43)$$



F70-09

A necessary condition⁶ to enable Eq. (43) to attain an extremum is that

$$\frac{\partial f}{\partial r} - \frac{d}{d\xi} \left(\frac{\partial f}{\partial \dot{r}} \right) = 0 \quad (44)$$

Now, with $f = n(r)\{r^2 + (\dot{r})^2\}$, application of Eq. (44) results in the following differential equation for extremizing curve C:

$$\frac{d^2 r}{d\xi^2} = \frac{1}{r^2 n} \left\{ \left[r^2 + \left(\frac{dr}{d\xi} \right)^2 \right] \left[r^2 \frac{dn}{dr} + rn \right] + r^2 \frac{dr}{d\xi} n \right\} \quad (45)$$

There is little hope of solving Eq. (45) analytically, even for well-behaved $n(r)$ functions, so the job was done numerically. The algorithm used to solve the differential equations (fourth-order Runge-Kutta) requires that equations of order p be expressed as p first-order simultaneous equations.

Defining

$$r_1 = r \quad (46a)$$

$$r_2 = \frac{dr}{d\xi} \quad (46b)$$

permits Eq. (45) to be written as

$$\frac{dr_1}{d\xi} = r_2 \quad (47a)$$

$$\frac{dr_2}{d\xi} = \frac{1}{r_1^2 n} \left\{ \left(r_1^2 + r_2^2 \right) \left(r_1^2 \frac{dn}{dr_1} + r_1 n \right) + r_1^2 r_2 n \right\} \quad (47b)$$

⁶L. E. Elsgolts, *Calculus of Variations*, Permagon Press, Oxford and New York, 1962.



2.4 ATTENUATION

2.4.1 Assumptions

The atmosphere is assumed to be a single absorber which attenuates light in the same manner as a colored glass filter. The extinction (or absorption) coefficient is assumed to be a point function of position.

2.4.2 Development of Equations

Let B be the energy incident upon an infinitesimally thin sheet of absorbing material. The change in energy dB through the sheet is assumed to be proportional to the incident energy and the path length ds . Thus,

$$dB = -pB \, ds \quad (48)$$

where p is the proportionality factor. The negative sign indicates an energy loss. Rearranging gives

$$\int_{B_1}^{B_2} \frac{dB}{B} = - \int_{s_1}^{s_2} p \, ds \quad (49)$$

Now if p is constant, Eq. (49) can be integrated to give the common expression for transmission through the medium:

$$\frac{B_2}{B_1} = e^{-p(s_2 - s_1)} \quad (50)$$



F70-09

If, however, p is a function of position and, as in the present application, wavelength (λ), it must remain inside the integral. Doing so gives the transmission along the path C (Fig. 2-3) as

$$T = \frac{B_2}{B_1} = \exp \left\{ - \int_C p(\bar{r}, \lambda) ds \right\} \quad (51)$$



F70-09

Section 3 COMPUTER IMPLEMENTATION

This section presents a detailed description of the computer program which was written to solve the albedo equations. A fair degree of computer language expertise is assumed on the part of the reader; a good knowledge of FORTRAN plus some familiarity with operating systems is a prerequisite for digesting the material.

A reading of this section will in no way make the results of the study (presented in Section 5) clearer to the reader.

3.1 COMPUTER USED

The computer that was used to develop the albedo analysis program and to run it, as well as the already existing refraction and attenuation programs, was an IBM 1130.

BBRC's 1130 configuration has an 8192-word memory, a type 1132 line printer, type 1442 card reader, type 1134 paper tape reader, type 1627 plotter, and a single disk drive capable of storing an additional 512,000 words.

Residing on the disk, and callable via control cards, are FORTRAN and RPG compilers, an assembler with macro capabilities, utility programs which facilitate disk housekeeping, and a supervisor program which watches over the whole operation.



F70-09

Although the 1130 is a small and relatively slow machine, it was chosen from the many available in the area for several reasons.

- It is the computer the author is most familiar with.
- By using the software to its fullest capabilities, core overflow problems can be circumvented.
- It is quite inexpensive.
- It is operated in an "open-shop" environment. The engineers and mathematicians operate the computer themselves, thus effecting virtually instantaneous turn-around - a most valuable asset during the program debug phases.

The IBM 1130 programming system restricts program and subprogram names, as well as variables and addresses, to a maximum of five characters. The resulting names are often curiosities which make sense only after a unique phonic facility has been developed by the reader.

3.2 FEATURES OF THE ANALYSIS SYSTEM

The albedo analysis system consists of several main programs which perform analysis and housekeeping functions and a host of associated subprograms. All of the programs and subprograms are written in FORTRAN except for one of the minor mainlines which is in assembler language.



F70-09

3.2.1 Primary Program: "LBDO"

The basic task of "LBDO" is to solve Eq. (30) in Section 2. A two-axis sensing system composed of orthogonal, identical sensors is assumed. In the process of evaluating Eq. (30) the program

- Determines the boundaries of integration [Eqs. (31a) to (31c)].
- Performs the necessary coordinate transformations [Eq. (21)].
- Computes the entrance angles, for both sensors, of light radiated from dA [Eqs. (31d) to (31g)].
- Computes the resulting output from each sensor.
- Computes the albedo factor for that particular dA.
- Calculates pertinent angles and distances.
- Accumulates incremental contributions during the numerical integration.

When the albedo-induced sensor outputs have been computed, the program transfers to "null-seeking" mode. That is, it determines the solar-point offset which would produce the same signal as the albedo just calculated and alters the attitude of the sensor accordingly. Of course, the sensor now views a slightly different portion of the earth, so the program loops back to recalculate the albedo output again. This process is continued until the net alignment correction is less than a specified amount. (For the problem treated in this study, a value of 3 arc-seconds was used.)



F70-09

In addition to the arithmetic, the program performs other mundane chores such as reading data, printing results, detecting certain errors, storing pertinent data on the disk, and so on.

Some of the more salient features of the "LBDO" program are the following:

1. Data can be entered either from the console keyboard or via the card reader, thus providing on-line control by the operator or batch-mode operation.
2. The sensor angular response characteristic is defined within a subroutine, as is the spatial albedo factor. This permits easy, rapid substitution of functions for different applications.
3. The maximum number of iterations of the null-seeking loop is specified at the beginning of the run. By setting this parameter to 1, the null operation can be bypassed in those applications where sensor output alone, rather than pointing errors, is desired.
4. There is no restriction on sensor alignment with respect to the earth-bound reference coordinates.
5. A given run can be aborted in two ways: fatal and nonfatal. The fatal abort is used when the run is patently invalid. All data is scrapped and new data is requested. The nonfatal abort was incorporated because of the long run time (>30 minutes) per iteration. Another computer



F70-09

user may have a very important, but short, job, and it would be unfair to make him wait for several hours until the "LBDO" solution converged. When a designated sense switch is thrown, the program will store all the parameters pertinent to the run on a disk data file. These parameters (the "state vector") include the fact that a non-fatal abort was performed and where in the program the sense switch was thrown. By retrieving the state vector at the beginning of the next run, computations can be resumed where they were broken off.

6. Two core-saving techniques are used due to the limited size of memory. One of these is the "subroutine overlay" feature of the 1130 monitor program. Normally, all the subroutines required by the program are placed in core with the main program. This takes up space, but the subroutine is there when you need it. The overlay feature permits certain specified subroutines to reside on the disk until they are needed, at which time they are read into a special "overlay area" in core, wiping out the last routine placed there. Thus, a great deal of space is saved at the expense of fetch time.

Another space-saving trick is to store, on the disk, the majority of the many titles, headers, and messages printed by the program and then read them into a single buffer as required. This technique has the added advantage of making it possible to alter the messages without recompiling the main program.



F70-09

7. An array of eight general-purpose storage locations is assigned within the program and can be referenced by any of the subroutines and used as required. For instance, a circular angular sensor response function could be defined using one of the constants for the radius and another for the output level. The subroutines can be made even more general in this way.

3.2.2 Associated Utility Programs

As was mentioned earlier, several main programs have been written to support "LBDO". Their basic functions will now be described.

"LBDOOD". This is a special program which handles data processing for the SS1090 angular response characteristic. It performs raw-data conversion, stores the data in a disk data file in an orderly manner, and plots the stored data. The data can be stored and/or plotted either as individual records or as a full block. The data file used is named "LBDO1".

The program was designed to handle SS1090 data only and would apply to other tabulated functions only by coincidence.

"LBDOOM". This program manipulates the data file which holds the various titles, messages, and headers required by "LBDO". It permits given messages to be added, altered, or moved within the file and can print the entire file on the line printer. The data file used is "LBDO3".

"LBDOOQ". During the evolution of "LBDO" various capabilities were added which were not a part of the original program concept. Sequential modification level numbers were assigned to the program as these changes were incorporated, and one of the disk-stored headers prints the current "mod" level.



F70-09

The computer uses many different data codes for character representation. In this particular case, the mod level is stored in EBCDIC,⁷ which is not particularly well suited to alteration via arithmetic operations. Thus, "LBDOQ" was written in assembler language to read the new mod level number from the console keyboard, convert the quantity from keyboard code to EBCDIC, and then patch the data file.

3.2.3 Subroutines

The following function subprograms and subroutines are used by the albedo analysis system. Their descriptions include the type of subprogram it is, its name and calling sequence, and the function it performs.

Function RSPNS (ALPHA, OMEGA, KROUT). If $K \neq 0$, the function prints an identifying title on the line printer. If $K = 0$, the sensor response is computed for light entering at the on- and off-axis angles ALPHA and OMEGA respectively. These angles are in degrees. The units of the sensor output are whatever makes sense to the user at the time.

Function TWEEK (V, ANGLE). Given the output SIGNAL of the sensor and the off-axis angle OMEGA of the sun, TWEEK computes the on-axis angle (in degrees) required for the sun to produce an output equal to SIGNAL.

Subroutine MESAG (MESNM, NDVIC). This subroutine reads a selected literal field (message, header, title, etc.), specified by NUM, from data file 4 (= LBD03) and outputs this field to I/O device number NDVIC. NDVIC = 1 corresponds to the console typewriter; NDVIC = 3 corresponds to the line printer.

⁷Extended Binary Coded Decimal Interchange Code.



F70-09

Subroutine LBDOZ (NUMBR). This subroutine reads an eight-character routing word from the console keyboard or the card reader, depending on whether "LBDO" is in on-line or batch input mode. The subroutine compares the routing word to a list containing the literals "KEYBOARD", "CARD", and "QUIT". If a match is found, the variable NUMBR is set equal to 1, 2, or 3 respectively. These numbers are then used by "LBDO" to alter the input mode. If no match is found, the routing word is re-entered or NUMBR is set equal to 4, depending on whether the keyboard or the card reader is in control.

Function ALBDO (THETA, PHI). This routine returns the albedo factor as a function of earth surface coordinates θ and ϕ (see Fig. 2-1). θ and ϕ are assumed to be in degrees.

Subroutine CRDMP. CRDMP produces core dumps of real common for debugging purposes. Its action is controlled by data switches 6 and 7. If data switch 6 is up, a core dump is obtained if either sensor output is not 0.0; if data switch 7 is up, a core dump is obtained regardless of the sensor output.

Subroutine ROT1 (VECTR, ANGLE). This subroutine performs an ortho-normal rotation about the 1 axis of the vector VECTR through an angle $-(\text{ANGLE})$. ANGLE must be supplied in radians.

Subroutines ROT2 (VECTR, ANGLE), ROT3 (VECTR, ANGLE). These are identical to ROT1 except that rotations are made about the 2 axis and 3 axis.

Subroutine XNORM (VECTR, LNGTH). Given the 3-vector VECTR, this routine transforms it to unit length and also returns LNGTH, the magnitude of the original, untransformed vector.



F70-09

Real Function INNER (A, B). Given the two 3-vectors A and B, this routine computes their inner (or "dot") product. That is,

$$\text{INNER} = \sum_{j=1}^3 a_j b_j$$

Subroutine GRAF (MAX, TITLE). This subroutine initializes, scales, and then draws the coordinate axes for program "LBDOZ". MAX is the maximum y-axis value, TITLE is an array containing title information to be written along the y axis.

3.3 PROGRAM USAGE

In the following directions on program usage, samples of execution decks are given. The data appearing on these cards must be placed in particular columns. No attempt is made to specify such placement in this write-up; the reader is referred to

IBM 1130 Disk Monitor System, Version 2
Programming and Operator's Guide
GC26-3717-6 (or later)

for such particulars.

3.3.1 "LBDO"

Before the program can be executed, the general-purpose subprograms INNER, MESAG, LBDOZ, XNORM, ROT1, ROT2, and ROT3 must be stored on the disk. The subprograms which are changed from application to application, RSPNS, TWEEK, and ALBDO, or *their equivalents* must be on the disk as well.



F70-09

Normally, to change one of the subprograms, e.g., ALBDO, the old routine would have to be deleted from the disk and the new version stored. This is a time-consuming and annoying process which can be circumvented by using the *EQUAT facility of the 1130 monitor. At the time of program execution, subprogram substitutions can be made by including an *EQUAT (SUB1, SUB2) card in the execution deck. Then wherever the main program calls SUB1, subroutine SUB2 will be called instead. Thus, a host of albedo-factor-defining functions can be stored on the disk simultaneously, all under different names, and the desired one specified on an *EQUAT record at run time. The same holds for the RSPNS and TWEEK routines of course.

The required execution deck is shown below.

```
// JOB          01
*EQUAT (RSPNS, RSPN1), (ALBDO, SUBX)
// XEQ  LBDO  02
*FILES (1,LBD01),(2,LBD01),(4,LBD03),(5,LBD04)
*LOCALLBDO,ROT1,ROT2,ROT3,RSPNS,TWEEK,MESAG,LBDOZ
```

If no subroutine substitutions are to be made, no *EQUAT card is needed and the 01 is deleted from the // JOB card.

As execution begins, the console typewriter prints a header giving the program name and current modification level. It then commences listing the required input variables. As this is a somewhat lengthy process, the printout can be suppressed at any time by turning data switch 0 on.



F70-09

Next comes a short dialog between the computer (via the typewriter) and the operator (via the keyboard). The date, operator's name, and I/O routing word are requested and must be supplied when called for. Up to 20 characters can be used for the date; up to 30 for the operator's name. The routing word must be KEYBOARD, CARD, or QUIT.

A QUIT entry tidies up the program, stores the state vector, and exits to the monitor.

A KEYBOARD entry continues the machine-operator conversation, since now all the data for the run must be entered through the keyboard. First, a run title is requested. Eighty characters are allotted, not all of which need be used, of course. Next, initial position and alignment data is requested. Enter separately, on F10.3 format, the following parameters.

- h: Vehicle altitude (kilometers).
- ξ : Vehicle-terminator angle (degrees).
- α : Sensor azimuth angle (degrees).
- ϵ : Sensor elevation angle (degrees).
- ρ : Sensor roll angle (degrees).

ERRMX: Upper bound on sensor alignment correction
(arc-minutes)

The next request is for the maximum number of null-seeking iterations to be permitted. Enter this on I3 format.

Finally, the eight arbitrary constants C_j , $1 \leq j \leq 8$, which the subroutines may need can be defined. This is done by first entering the subscript j on I1 followed by the value for C_j on F10.3. The order of j 's is immaterial, and you can change your



F70-09

mind and redefine a value if desired; the values you get are the last values entered. If you fail to define a particular constant, it retains the value it had during the last run. In the beginning, I created all the constants to be zero, and this is what they will be if they have never been changed. To leave this loop, enter a value for j which is outside the range specified above.

The program now goes into computation mode. When completed, it loops back and asks, via the typewriter, for a new routing word.

A CARD entry transfers control immediately to the card reader, where data cards must be waiting. These data cards are grouped into "data packages" of three or four cards; one such data package is required for each run to be made. A data package consists of the following cards:

Card 1: TITLE (80 characters).

Card 2: H , ξ , α , ϵ , ρ , ERRMX, ITMAX, J (6F10.3, 13, 1X, I1).

Card 3: C_j ($j = 1, \dots, 8$) (8F10.3).

Card 4: Routing word (KEYBOARD, CARD, or QUIT).

Note 1: ITMAX is the maximum number of null-seeking iterations permitted.

Note 2: J is a flag which indicates whether constants are to be defined on Card 3 ($J \neq 0$ or blank) or whether Card 3 is omitted ($J = 0$ or blank). In the latter case, Card 4 becomes Card 3.

At the completion of a card-generated run, the routing-word card is read and the appropriate action is taken.



F70-69

During the computation phase of the program, there are several useful options the operator can exercise.

Percent Completion. Data switch 5: The current iteration number and its degree of completion is typed. This information is useful in calming a queue of anxious users.

Core Dumps. These are limited-area core dumps; only the real (as opposed to integer) variables pertinent to the program are dumped. The format is decimal rather than the more common hexadecimal (base 16).

Data Switch 6: A dump occurs only if one of the sensors had a nonzero output for the last dA processed.

Data Switch 7: A core dump occurs regardless of the sensor's output.

Fatal Abort. Data Switch 10: Computation stops immediately and a new routing word is read, either from the keyboard or card reader depending on the current data-input mode.

Nonfatal Abort. Data Switch 13: Computation stops and a flag is set indicating this action as well as where in the program the abort occurred. All the pertinent data (the "state vector") in core is then stored in a disk data file, and the program exits to the monitor.

The next time the program is run, the first thing that happens is the state vector is retrieved from the disk and the abort flag is checked. If it indicates that the last job was halted by a data switch 13 abort, the operator is asked if he wants to continue that run or not. He types YES or NO accordingly.



F70-09

3.3.2 "LBDOM"

The execution deck for this program is very simple.

```
// JOB  
// XEQ LBDOM
```

This program is stored on the disk in core-image format, and all data file correspondences have been predefined.

The program performs four functions, and the operator types ALTER, MOVE, LIST, or EMPTY after he has been asked (via typewriter) what he wants to do. If a word other than one of the above four is entered, the program goes into a short housekeeping loop and then exits to the monitor.

ALTER. When requested, enter j , the number (or address) of the message to be redefined on I2. $1 \leq j \leq 32$. Then type in the new message. The program loops back and asks for a new "record-to-be-altered" number. If j lies outside the valid range, a new function is requested.

MOVE. The typewriter prints XX TO YY. Enter XX (the address of the record to be moved) and YY (the destination address) on I2, 4X, I2 format (e.g., 27 TO 04 can be entered, although the TO is superfluous.) In the process, the original YY record is lost, the original XX record is retained. Both XX and YY must lie in the range $1 \leq XX, YY \leq 32$. If an invalid address is specified, a new function is requested.

LIST. The contents of the entire message file is printed on the 1132 printer. Blank messages are omitted from the listing.



F70-09

EMPTY. The entire file is filled with blank messages. This function is used only during file initialization or by vandals.

A new function automatically is requested upon completion of the latter two options.

3.3.3 "LBDOQ"

The execution deck for this program is

```
// JOB  
// XEQ LBDOQ
```

The following message is typed out:

ENTER NEW MODIFICATION LEVEL (SINGLE DIGIT)

The program is now ready to accept the new mod-level number, but since the program is written in assembler language, the keyboard and typewriter behave differently than if coded in FORTRAN. A carriage return does not occur, only one digit can be entered, and when it is, the EOF button need not be pushed.

The modification level is altered and program "LBDOM" is executed automatically; no execution deck is needed for it.

3.3.4 "LBDOD"

This is a particularly specialized program whose only function is to load SS1090 sensor response data to a data file and plot it. The program is not general enough to be used to load other response characteristics defined as tabulated points, so a discussion of its operation is not included. If necessary, its structure and use can be easily deduced from the program listing which appears in Appendix B.



F70-09

Section 4

MEASUREMENT OF SS1090 RESPONSE CHARACTERISTICS

At the time the acquisition sensors were being built and tested, the knowledge of their susceptibility to albedo had not yet become a requirement. Since they were to be used only as null-seeking devices, only their on-axis characteristics were measured.

With the advent of the current study, however, the sensor's behavior to light entering at large off-axis angles had to be known. Fortunately, one of the SS1090 units (serial number 007) was at BBRC for rework. It was taken to the BBRC sun-test facility and its output was mapped over a large region.

Since the response function was to be used in a digital computer, the sensor output was sampled at discrete points by a digital voltmeter and the readings printed by a digital printer.

Data was taken on only the azimuth sensor (yaw output channel) and only for positive on- and off-axis angles. It was assumed that variations between sensors and individual quadrants of sensors were minor and would produce only second-order effects.

As the data was taken, the sensor was oriented by the same series of rotations used in the analysis; that is, the sensor was offset a given angle in elevation and then rotated in azimuth (ref. Eq. 21, Section 2).

The sensor output was measured over the range 0 to 45° in both axes at 0.5° increments. Thus 8281 samples were taken. To facilitate the gathering of this much data the process was partially mechanized. The elevation axis had to be positioned manually, but the azimuth axis was motor driven through a worm reduction gear.



F70-09

Attached to the motor shaft was a dimpled wheel which tripped a microswitch every 0.5° of azimuth axis rotation. The microswitch in turn triggered a digital voltmeter which sampled the AGCed sensor output and printed the result. It is estimated that the data was taken within 3 arc-minutes of the desired orientations.

To prevent skylight from affecting the readings, a large baffle was constructed and mounted to the sun-tracking fixture. A black felt shroud covered the entire setup.

The test arrangement - baffle, shroud, trigger wheel, etc. - is shown in Figs. 4-1 through 4-4.

The resultant data was streamed into the computer, processed, and plotted. These curves are also included. Figure 4-5 shows a family of response curves with sensor output given as a function of azimuth angle. Each curve is for a different, constant elevation angle. It can be considered as depicting slices out of a three-dimensional surface. Thus, sensor output is the vertical axis, azimuth angle is the horizontal axis, and the elevation angle axis comes out of the page. The slices are made parallel to the sensor output-azimuth angle plane.

Figure 4-6 is an isometric view of the response surface. For clarity, it is plotted only over the range $0^\circ \leq \text{azimuth} \leq 12^\circ$ and $0^\circ \leq \text{elevation} \leq 30^\circ$.

The angular response characteristic of the sun presence detector, before the level-sensing electronics, was also measured. Since, for the purpose of this study, the characteristic could be considered radially symmetrical, only one curve was taken.



F70-09

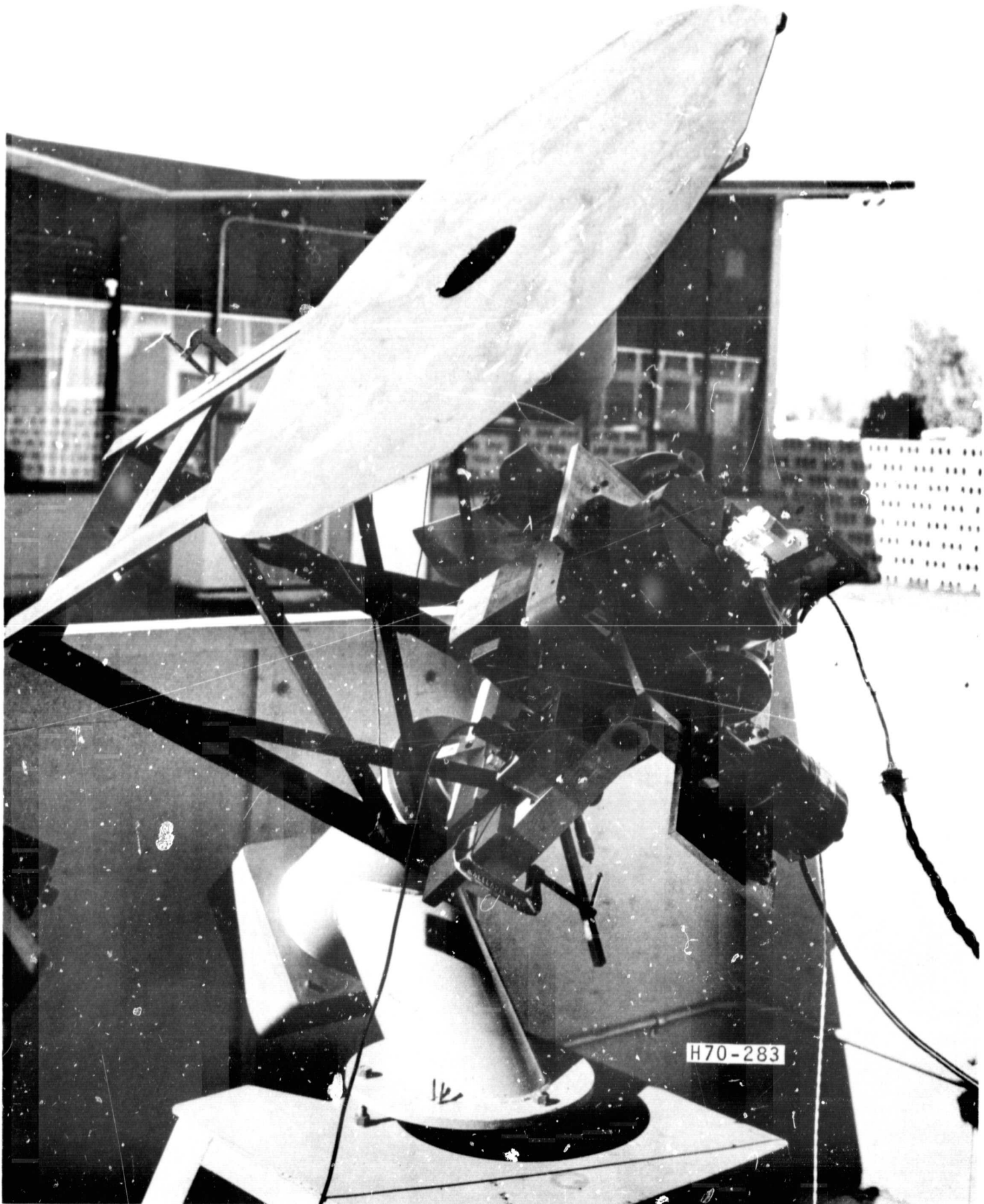


Fig. 4-1 SS1090 Test Fixture Light Shield



F70-09

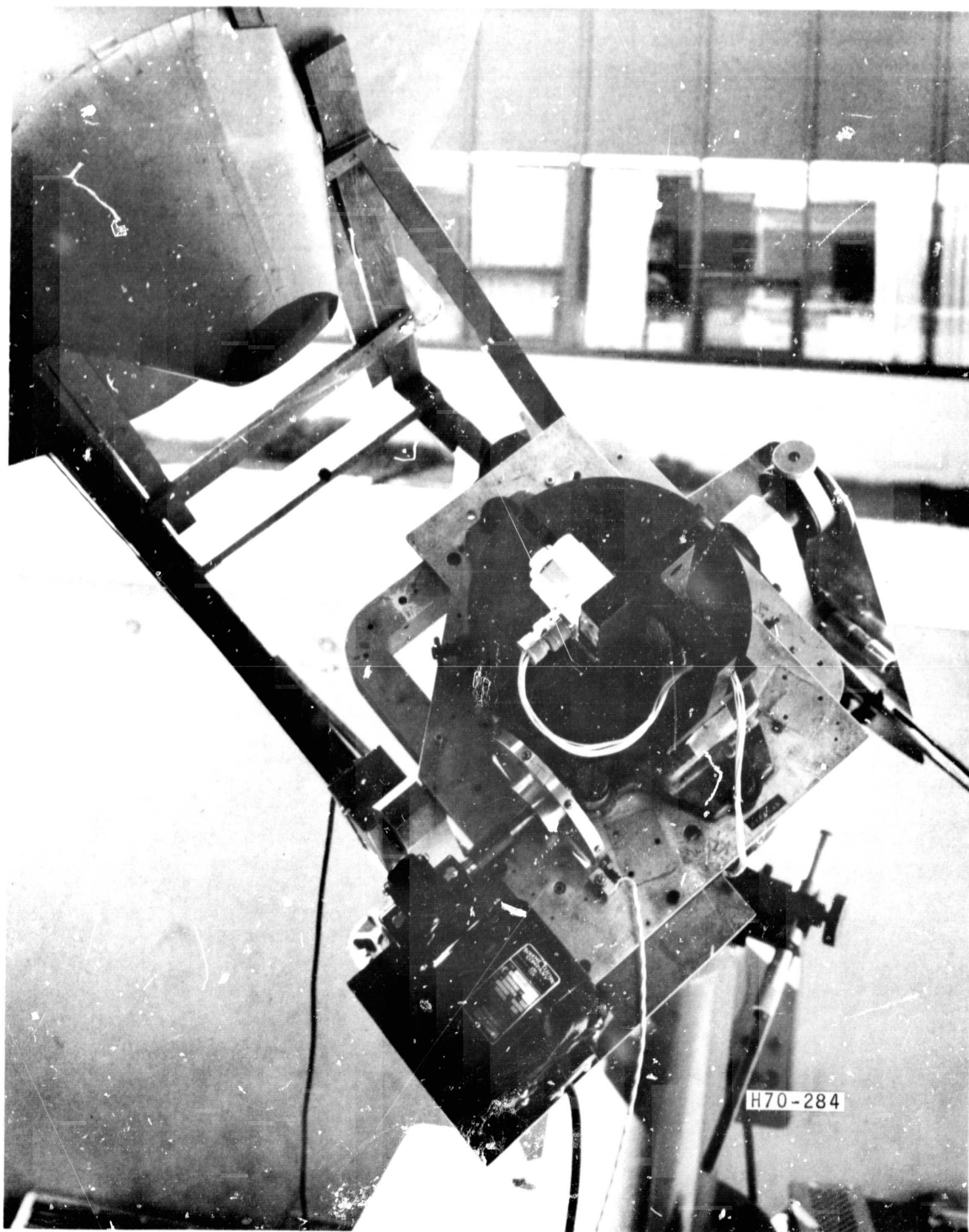


Fig. 4-2 SS1090 Test Fixture Light Shield



F70-09

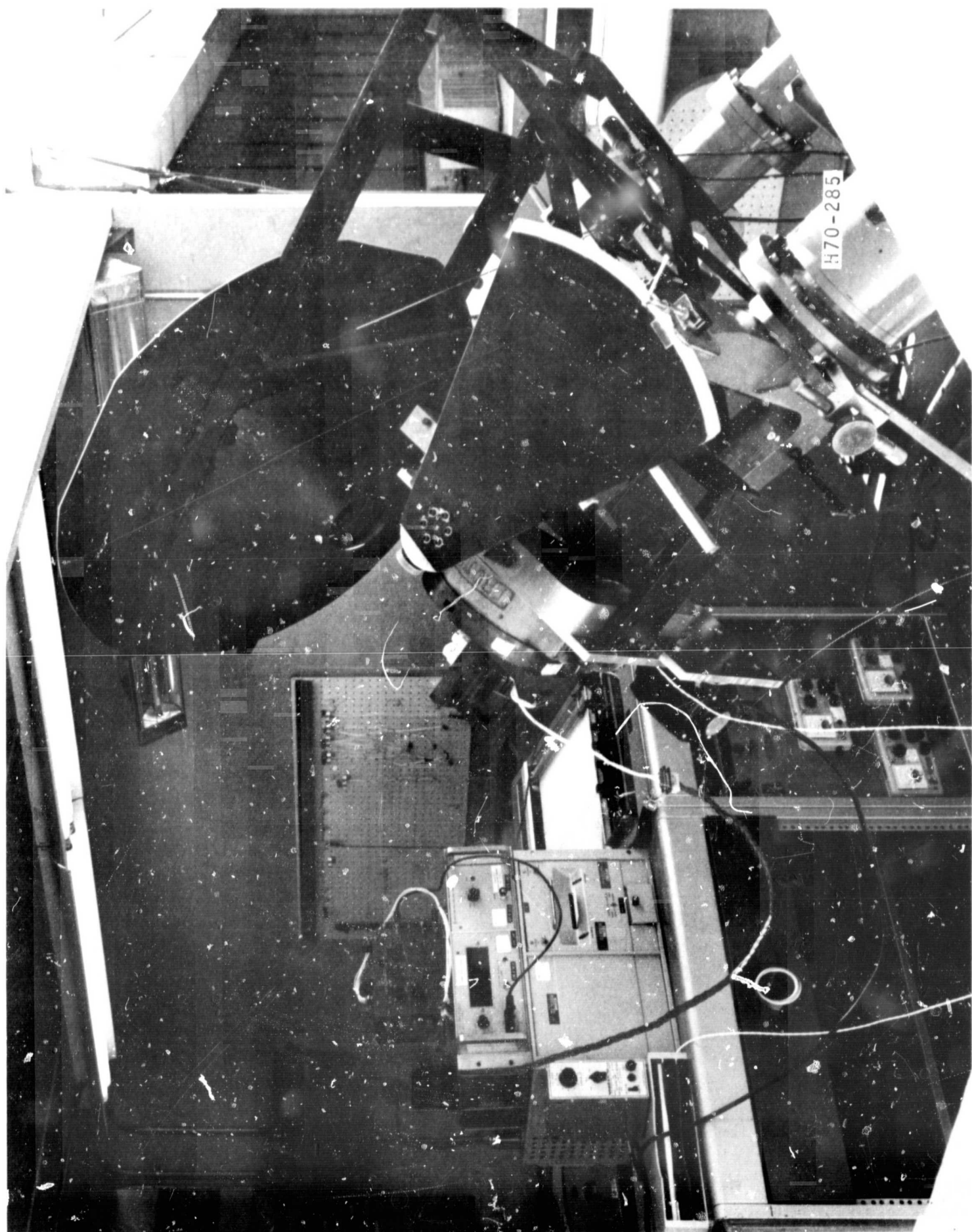


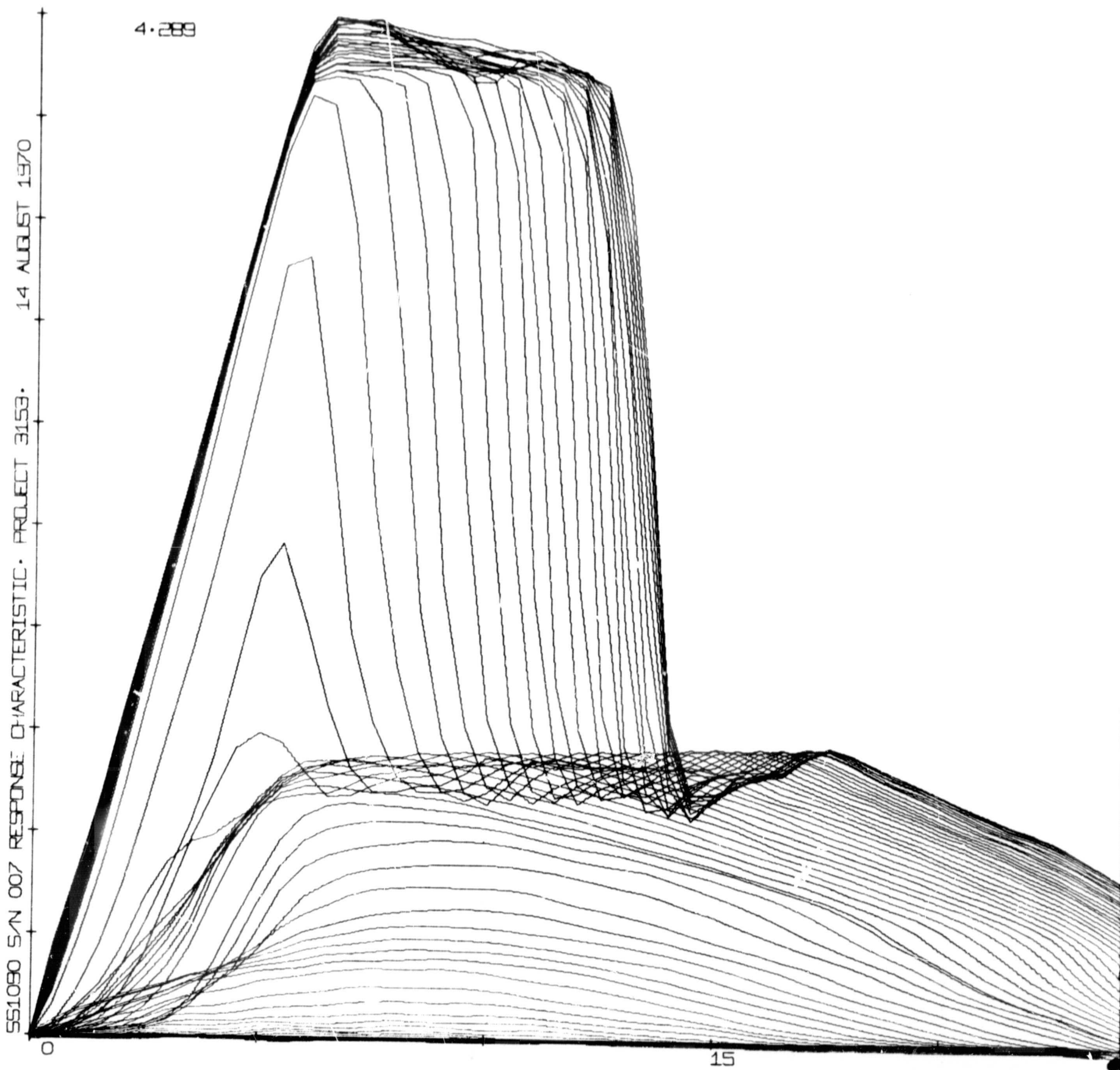
Fig. 4-3 SS1090 Test Fixture Elevation Axis Protractor and Data Acquisition Equipment



F70-09



Fig. 4-4 SS1090 Test Fixture: Operational Configuration, with Shroud



FOLDOUT FRAME



F70-09

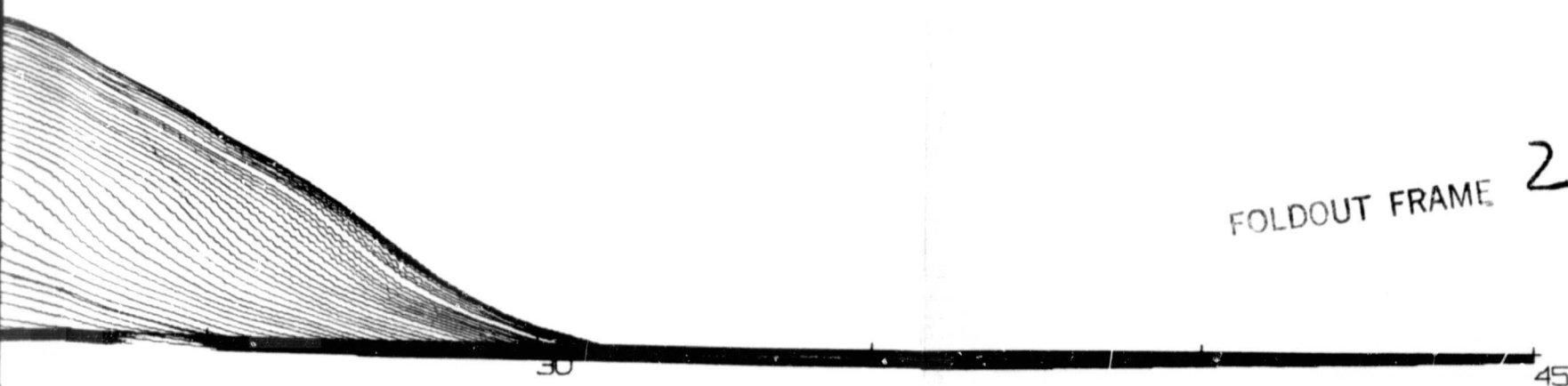


Fig. 4-5 Sensor Response Versus Sensitive Axis Rotation



F70-09

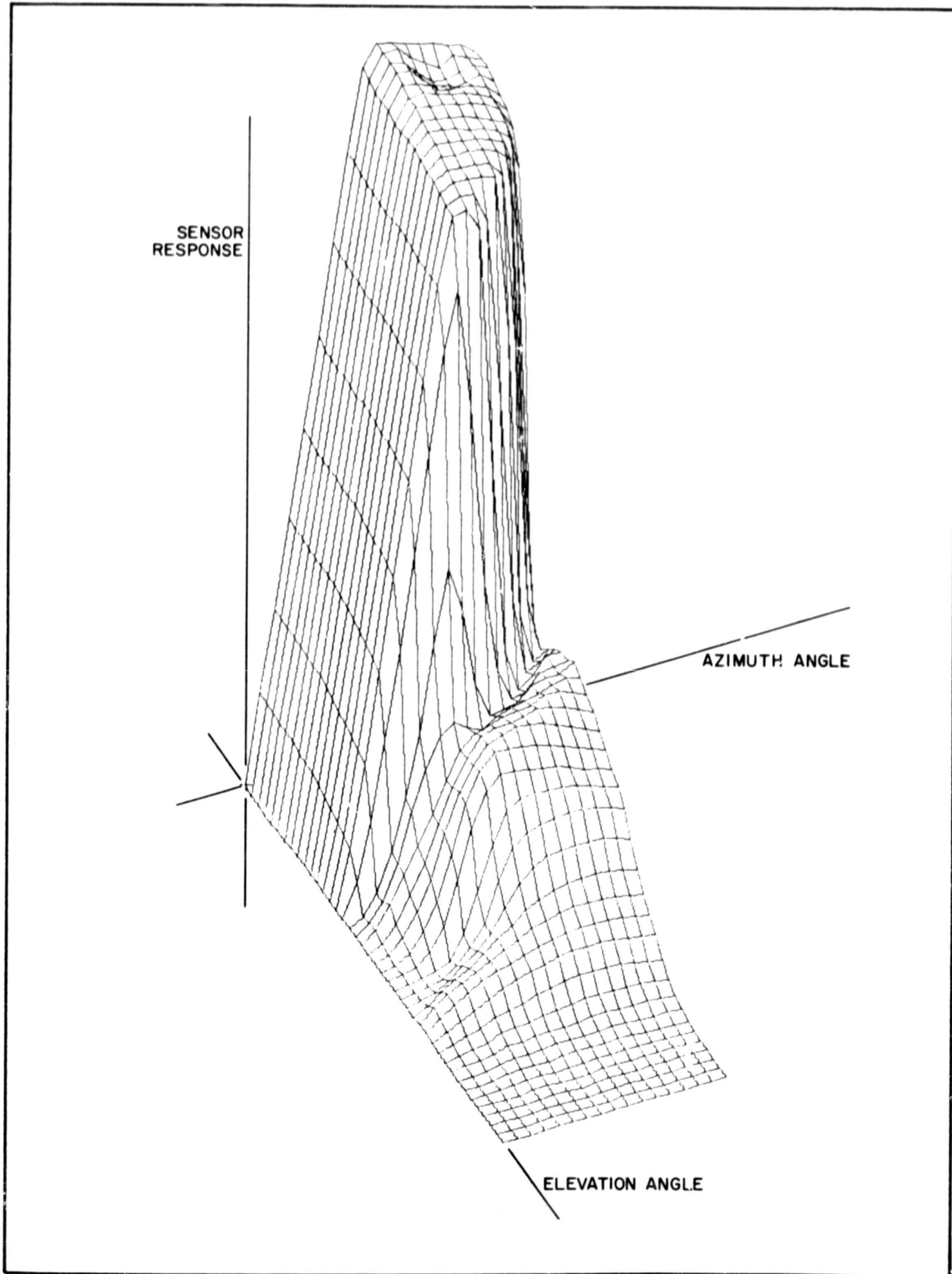


Fig. 4-6 Isometric View of Sensor Response Function



F70-09

Section 5 RESULTS

5.1 ALBEDO ERROR NEAR TERMINATOR IN SUN POINTING MODE

The satellite was placed 435.5 km above the surface of the earth and at angular displacements from the terminator varying over the range $-15^\circ \leq \xi \leq +10^\circ$. At each position the null point between solar and albedo outputs was found by directing the computer to perform iterations until the angular corrections were less than 3 arc-seconds.

A constant albedo factor, independent of earth coordinates (ϕ, θ) , of 0.6 was used, and the sensor was aligned to concentrate maximum albedo effect in the elevation sensor. The azimuth sensor was unaffected because of symmetry. (See Section 2 for a definition of "azimuth" and "elevation" sensors.)

This elevation-sensor pointing error is plotted as a function of ξ in Fig. 5-1. The worst error is found to occur over the range $-11^\circ \leq \xi \leq +1^\circ$, where the error ranges approximately between 22 and 34 arc-seconds.

A curious dip occurs in the error curve in the vicinity of $\xi = -7^\circ$ and is a result of the unique shape of the SS1090 angular response surface. Figures 4-5 and 4-6 show this surface to be composed of two primary parts: a high-output center region of limited width and breadth, and a relatively low-output region of extended size.



F70-09

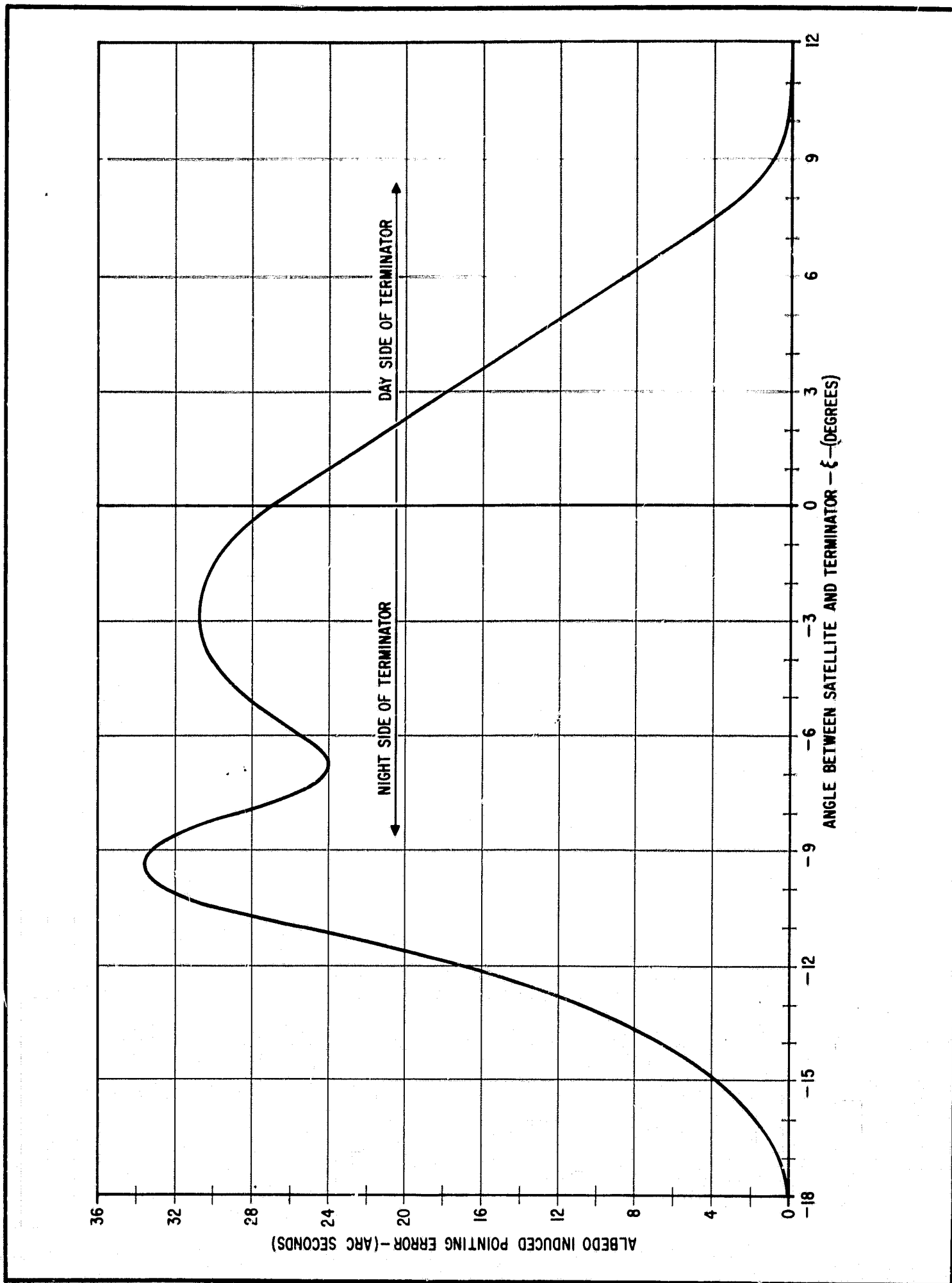


Fig. 5-1 Albedo-Induced Pointing Error of ATM Acquisition
Sun Sensor Versus Angle ξ Between Terminator and
Satellite Radius Vector



F70-09

The peak occurring at $\xi = -9.5^\circ$ is due to albedo entering the sensor in its high-output region. The satellite is coming up on the terminator, the sun is appearing over the horizon, and the narrow band of illuminated cloud cover is just below the null axis. Thus, it produces a high output because the sensor is very sensitive in this region, despite the limited narrow band of this sensitivity. The low-sensitivity region, which responds to light sources far off the null axis, is still viewing the dark portion of the earth.

As the vehicle moves toward the terminator, the band of sunlit clouds drops away from the null axis and its effect, and hence the pointing error, begins to decrease.

Soon, however, the low-sensitivity area of the sensor becomes more and more affected; its wide field of view more than makes up for its limited response, and the net error increases again.

Finally, as the vehicle crosses the terminator and continues on its way, smaller and smaller areas of earth fall into the sensor's field, the albedo-induced output decreases, and the error diminishes to zero at $\xi = 10^\circ$.

The actual response data for the acquisition sensor (Figs. 4-5 and 4-6) was not used to generate the error curve, Fig. 5-1, because of the excessive computation times required. The data was stored in 91 rows of elevation angles, each holding the outputs at 91 azimuth angles. Given the azimuth and elevation angles (σ and ω) of an incident light ray, the bracketing elevation-angle rows, and then the bracketing azimuth angles had to be computed. The four tabulated outputs which surround the desired part then had to be fetched from the disk and a two-dimensional interpolation operation performed (see Appendix D) to get the sensor output. The process was then repeated for the orthogonal sensor. The



F70-09

result of these machinations was a three-hour iteration period - six hours per data point.

The measured response functions (Figs. 4-5 and 4-6) were instead used to model an approximate analytical substitute. This model is included in Appendix B. As a check, the counterfeit was checked against the real data function in the area $\xi = -9.5'$. The analytical model produced a slightly smaller error - by about 1 arc-second.

5.2 REFRACTION AND ATTENUATION EFFECTS IN SUN POINTING MODE

Equations (47a) and (47b) were solved using a fourth-order Runge-Kutta algorithm; the integration increment ds was 100 meters. The function used for $n(h)$ was

$$n(h) = 1.0 + 0.000338 e^{-0.12542h} \quad (52)$$

where h is the height from the earth's surface in kilometers. This function was derived by least-squares techniques operating on data given in

"The Effect of Refraction on the Setting Sun
As Seen from Space in Theory and Observation,"
NASA Technical Note D1721 (W. S. Cameron,
J. H. Glenn, M. S. Carpenter, and J. A. O'Keefe)

Different runs were made for light rays of grazing elevations ranging between 4 and 45 kilometers and the total refraction angle determined. An extrapolation program then computed where in orbit the satellite had to be when the refracted light ray reached 435.5 km. Thus the error curve shown in Fig. 5-2 was constructed; it gives the total refraction angle as a function of satellite angular position ξ from the terminator.



F70-09

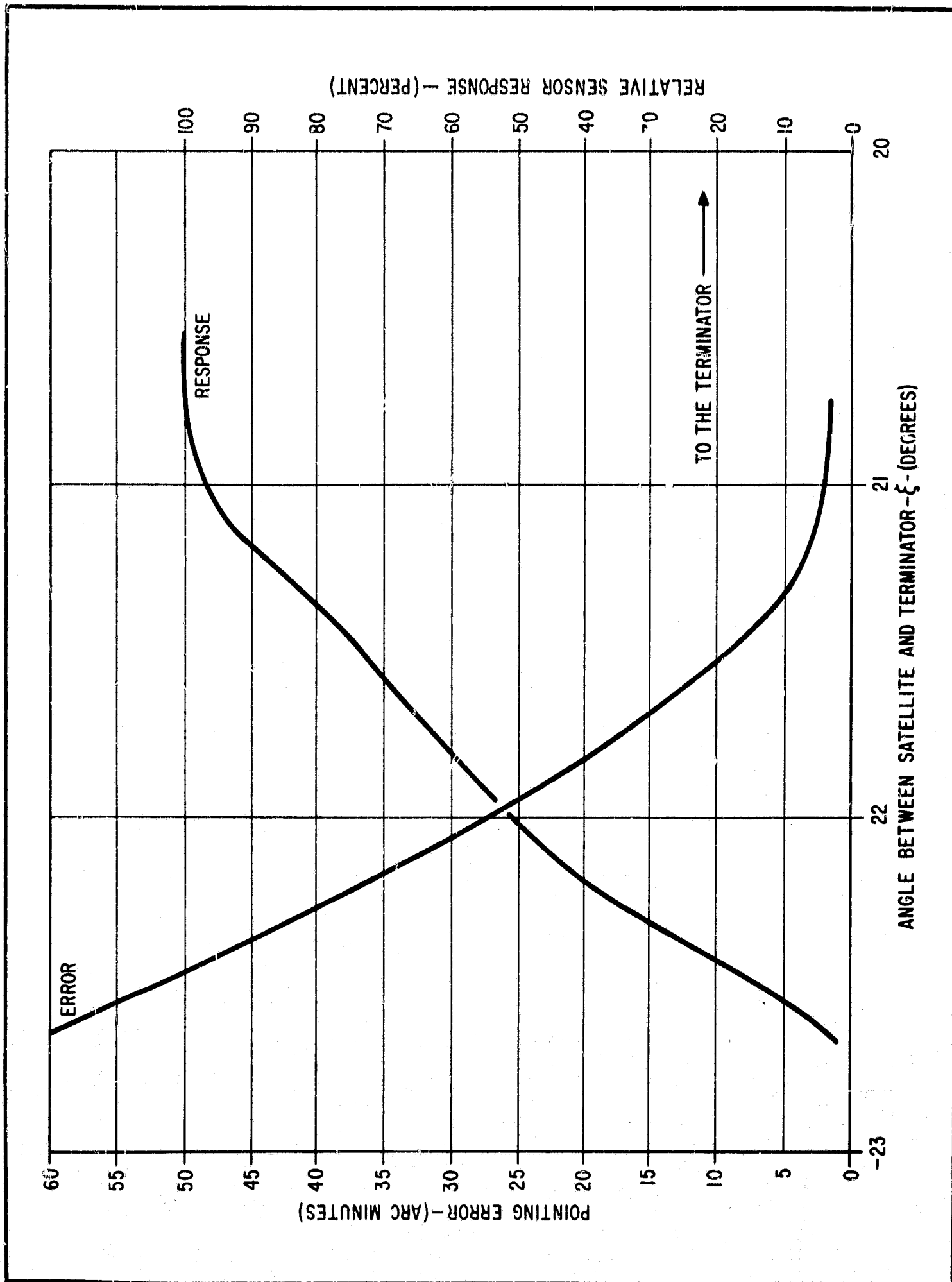


Fig. 5-2 Pointing Error and Relative Response Due to Atmospheric Refraction and Attenuation Versus Angle ξ Between Terminator and Satellite Radius Vector



F70-09

A minor philosophical point regarding the term "error" should be made. Throughout the greater part of this report, the word "error" has meant a deviation from a desired or intended condition. Thus, the slight angular displacement caused by albedo is an "error" because the sensor is supposed to point at the sun. In the case of refraction effects, however, that is precisely what the sensor is doing. It can't help it if the sun has appeared to move. Hence, "error" is really a misnomer, but to invent a really good describer would be a bother. Let us agree on this special interpretation of the term and worry the subject no further.

After each refracted ray was traced, the data was used to evaluate the line integral of transmission [Eq. (51), Section 2]. The extinction coefficient, $p(r, \lambda)$ was taken from

"Handbook of Geophysics and Space Environments,"
Air Force Cambridge Laboratories, Chapter 7,
'Atmospheric Optics', (L. Edterman and R. B.
Toolin)

The data is tabulated as functions of wavelength and altitude, so a two-dimensional interpolation scheme was included in the transmission program (see Appendix D).

For each light ray, then, an associated transmission value was obtained (for $\lambda = 0.8 \mu$), and this data is plotted in Fig. 5-2 along with the error curve.

Since the gain (or scale factor in volts/degree) of the acquisition sensor in the vicinity of null is directly proportional to the peak output of the sensor, and this in turn is directly proportional to the brightness of the source, the transmission curve indicates how the gain of the sensor varies with position.



F70-09

The curve also shows that the sun presence signal, set to trigger at 50% of full sun intensity, will drop out at approximately $\xi = -22^\circ$.

In the event the alignment state of the ATM is maintained inertially, during nighttime, at the value it had as it moved behind the terminator, the data presented in Fig. 5-2 will be useful in analyzing the dawn perturbations the vehicle will experience. It will see approximately a two-degree offset in the solar position, and the attitude control loop will attempt to correct this with a servo made sloppy by the atmospheric attenuation of sunlight.

5.3 SENSOR OUTPUT DUE TO EARTH ALBEDO ALONE

If, for some reason, the ATM loses solar pointing, the system will shift into "search mode" and the vehicle will be augered about until the sun is found again.

The question is: If, during this process, the sensor is pointed at the earth, could the SS1090 sensor output be high enough to instigate an invalid acquisition?

The answer is: No.

The "LBDO" program was used to analyze this problem. The ATM was placed 435.5 km above the earth with $\xi = 90^\circ$. This placed it on a line between the sun and the center of the earth, directly over the most brightly illuminated region of the surface.



F70-09

The roll and elevation angles of the sensor were both set to zero and a series of runs was made for sensor azimuth angles varying between 90 and 180° (0° points the sensor at the sun, 180° points it directly at the earth). The maximum number of iterations per run was set at 1, keeping the program out of "null-seek" mode. Only the sensor outputs due to albedo were desired.

The results are plotted in Fig. 5-3. The output rises from zero to a maximum of 1.2 volts (10% of peak output for solar illumination) and then decreases toward zero as both halves of the azimuth sensor become illuminated more or less equally by the albedo.

For the region $135^\circ \leq \alpha \leq 180^\circ$, the curve is seen to go berserk; there is a good reason for this, although the data in that region is totally invalid. The SS1090 angular response characteristic was measured using the sun as a target - a circle with a 0.5° diameter. The optical system of the SS1090 detectors distorts the target image greatly, and small variations in target size are completely masked. Very large targets are guaranteed to cause discrepancies, however. In the albedo mathematical model, the radiating element dA is $\sin \phi \, d\phi \, d\theta$, where ϕ and θ are the earth coordinates of latitude and longitude respectively. Within "LBDC" $d\phi$ and $d\theta$ are held constant at 0.5° each. Now, where dA is far from the sensor, e.g., near the horizon, dA appears small, partly due to the great distance, but primarily because it is viewed on edge. When dA is located almost directly under the sensor, however, it subtends a huge angle *at the sensor* (about 7.4°), and the sensor output computed by subroutine "RSPNS" can in no way be correct.



F70-09

As a check, the albedo response curve was recomputed using a larger dA : $1 \times 0.5^\circ$. There was no deviation between results over the range $90^\circ < \alpha < 125^\circ$, a slight deviation crept in over $125^\circ < \alpha < 135^\circ$, and an entirely different disaster was produced over $135^\circ < \alpha < 180^\circ$.

This indicates that although the computer simulation produces erroneous results under certain circumstances, the problem is confined to areas of little or no interest in this specific application. The calculation of peak sensor output during an earth scan is virtually independent of dA , as it should be, in the vicinity of maximum output.

The output at 180° , incidentally, is correct. Although the incremental outputs that were computed were totally wrong, they cancelled because of the symmetry of the situation, as indeed they should.

For clarity, only one half the response curve is given in Fig. 5-3. The remainder is identical to that shown, except that it is reflected about the point $(0, 180^\circ)$. In other words, if the curve shown is called $f(\alpha)$, then $f(-\alpha) = -f(\alpha)$.

Incidentally, during computations for the study phase termed "Albedo Error near Terminator in Sun Pointing Mode," dA was again located very near the horizon, where the simulation produces valid results.

5.4 SUN PRESENCE DETECTOR OUTPUT DUE TO ALBEDO ALONE

During the reacquisition search mode, the sun presence detector (which forms a part of the acquisition sun sensor) will produce an output as it scans the earth. It is of interest to know the maximum such output which can be expected.



F70-09

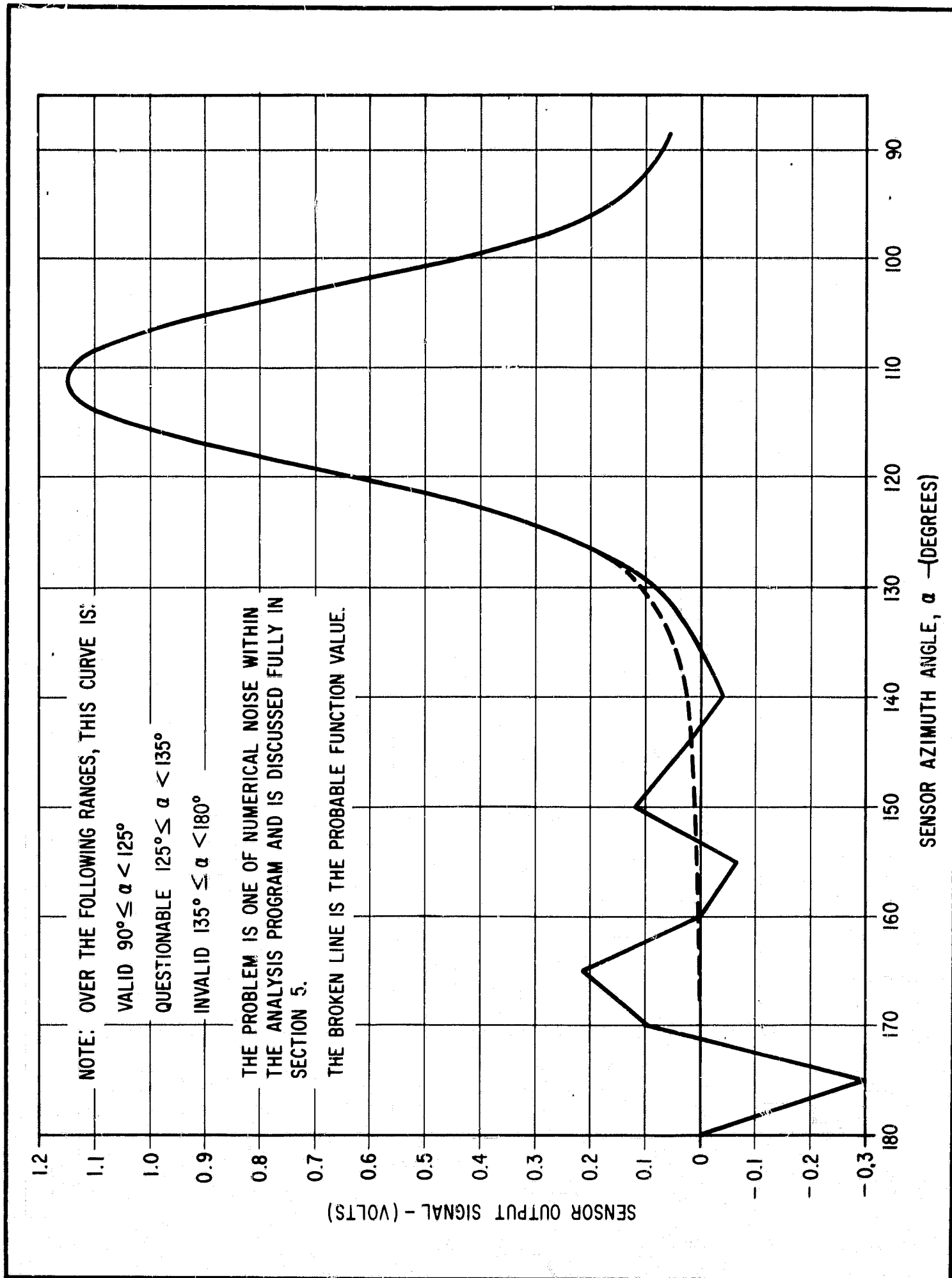


Fig. 5-3 Sensor Output Due to Albedo Alone Versus Null Axis-Solar Vector Angle α (albedo factor = 0.6)



F70-09

The situation which will produce the greatest signal from this detector is that of the vehicle directly between the sun and the center of the earth and with the detector pointed directly at the earth. These are precisely the conditions, however, for which the LBDO program does not perform well⁸.

Fortunately, this case is simple enough to be handled analytically. Certain simplifying assumptions, which tend to give a conservative result, are made to render the problem tractable.

In the coordinate system shown in Fig. 5-4, the (2,3) plane lies on the surface of the earth, which is assumed flat, and the 1 axis points at the sun. The ATM is also located on the 1 axis a distance h above the earth. Light radiated from dA leaves at an angle γ with respect to the earth's normal and travels a distance p up to the vehicle. Let "a" denote the albedo factor, assumed to be constant. Then, the output u of the detector, relative to its sun induced output is

$$u = \frac{a}{\pi} \int_0^{2\pi} \int_0^{r_1} \frac{\cos \gamma \, r \, dr \, d\beta}{p^2} \quad (53)$$

where r_1 is the radius of the circular area on the earth visible to the detector.

⁸The incremental radiating areas dA could be chosen small enough to produce good results, but the execution time would increase greatly.



F70-09

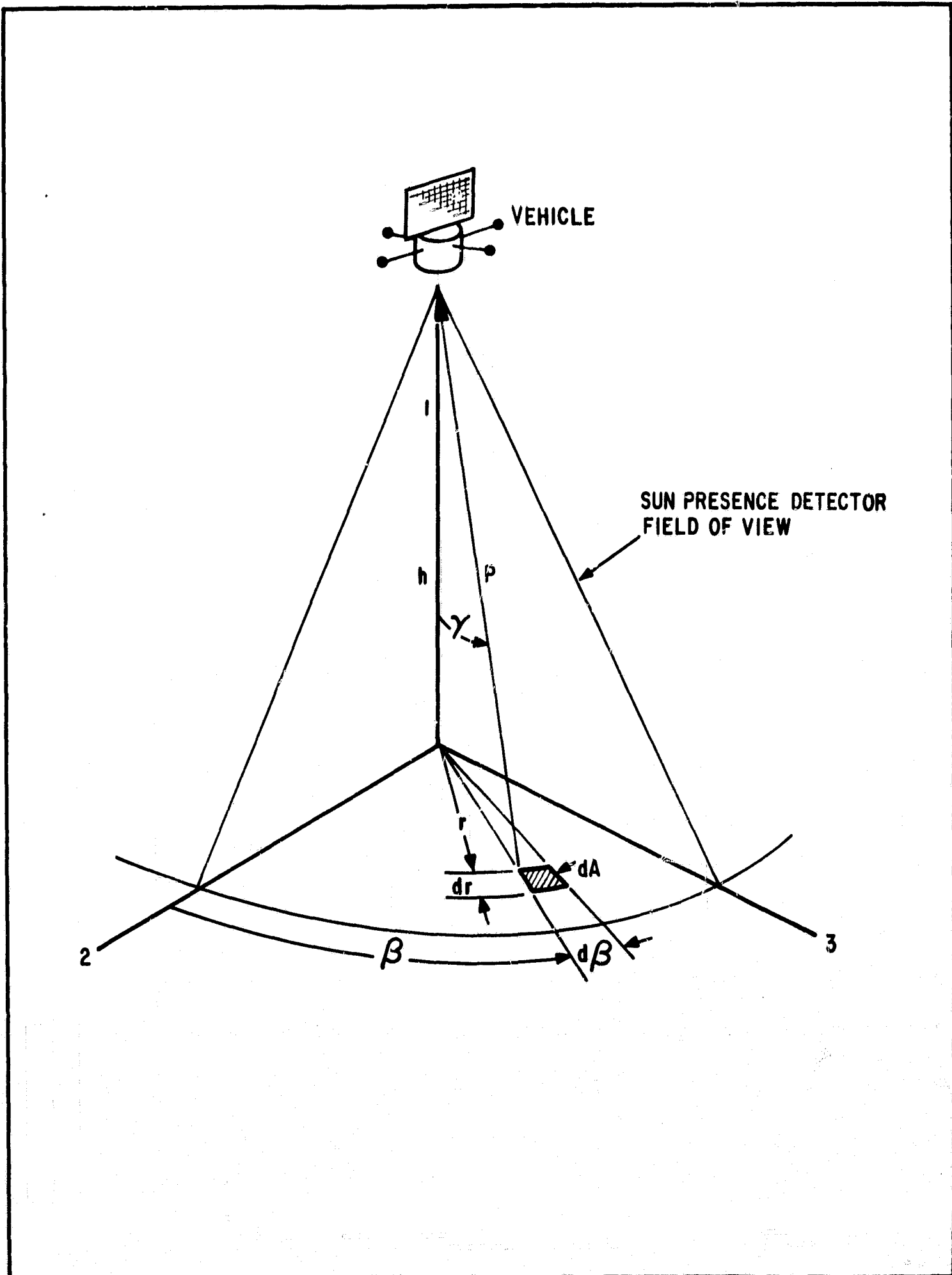


Fig. 5-4 Geometry for Approximate Albedo Analysis



F70-09

The output of the sun presence detector has been assumed to be constant over the range $0 \leq \gamma \leq \gamma_{\max}$, independent of β , and zero elsewhere. This is reasonable as evidenced by the detector angular characteristic shown in Fig. 5-5.

Equation (53) can be simplified considerably by

- Replacing p with the constant h .
- Replacing $\cos \gamma$ with 1.0.

These simplifications produce a larger integrand, as does the assumption of a flat earth. Equation (53) now becomes trivial:

$$u = \frac{a}{\pi h^2} \int_0^{2\pi} \int_0^{r_1} r \, dr \, d\beta \quad (54)$$

$$= \frac{a}{h^2} r_1^2 \quad (55)$$

As seen in Fig. 5-4,

$$r_1 = h \tan(\gamma_{\max}) \quad (56)$$

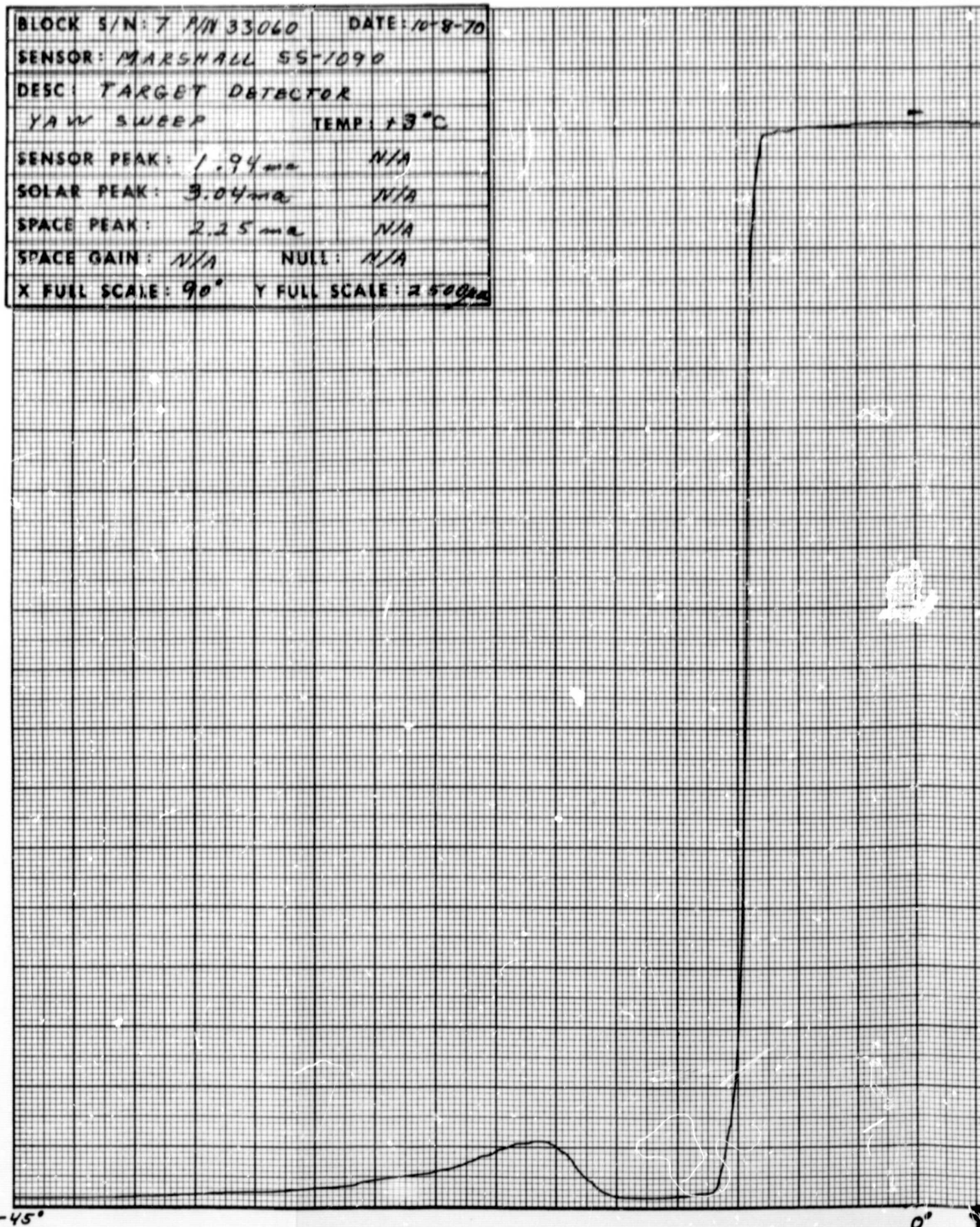
Thus,

$$u = a \tan^2(\gamma_{\max}) \quad (57)$$



SENSOR DATA BLOCK

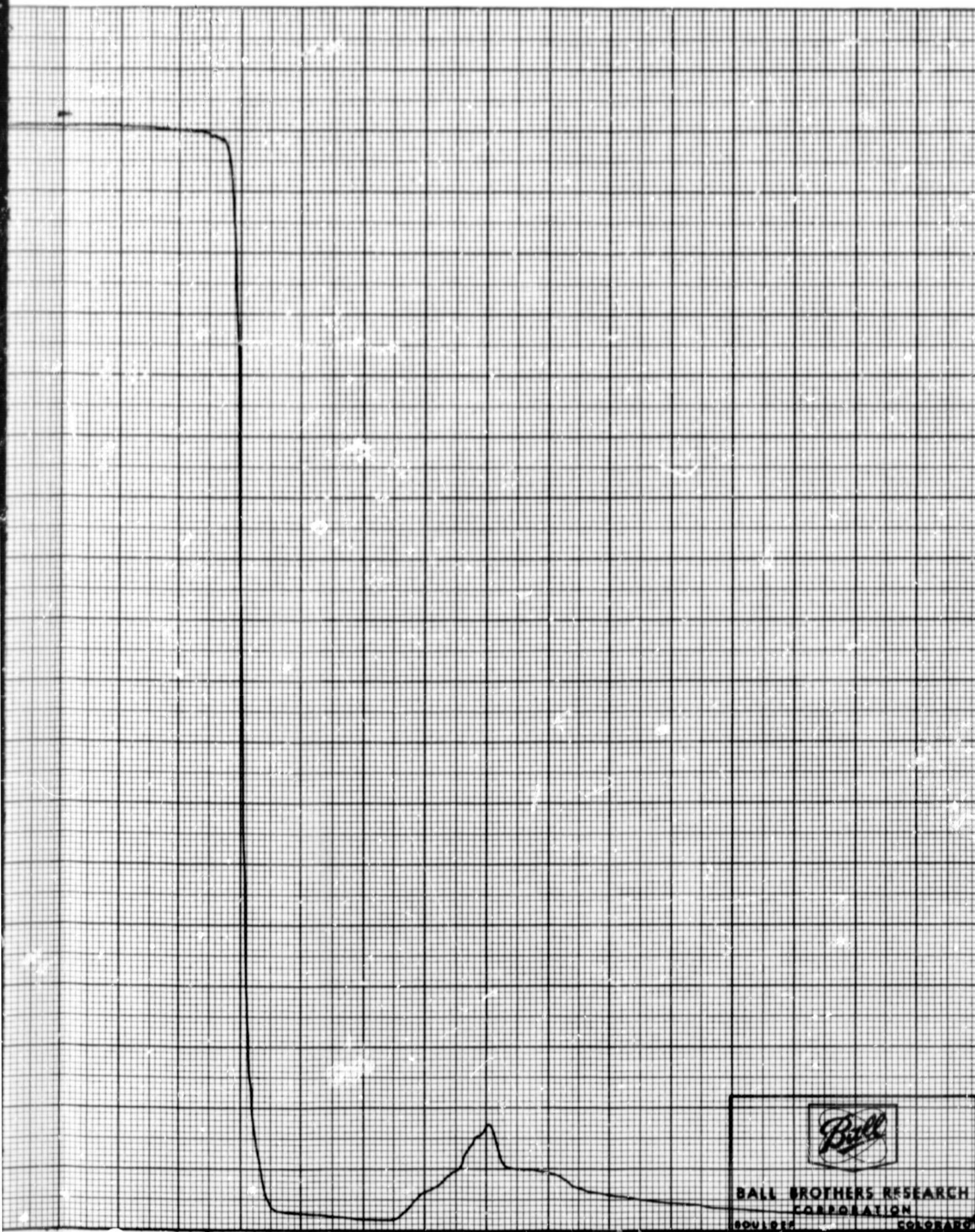
BLOCK S/N: 7 P/N 33060	DATE: 10-8-70
SENSOR: MARSHALL 55-1090	
DESC: TARGET DETECTOR	
YAW SWEEP	TEMP: +3°C
SENSOR PEAK: 1.94 ma	N/A
SOLAR PEAK: 3.04 ma	N/A
SPACE PEAK: 2.25 ma	N/A
SPACE GAIN: N/A	NULL: N/A
X FULL SCALE: 90°	Y FULL SCALE: 2500 ma



FOLDOUT FRAME

Fig. 5-5 Sun Presence Detector Angular Response Characteristic

TA BLOCK



0°

RETAIN ORIGINAL-RECORD COPY



BR-147 8-69
+45°

FOLDOUT FRAME 2



F70-09

Appendix A
MATHEMATICAL NOTATION



F70-09

This appendix is intended to explain and clarify the notation used in manipulating vectors and matrices. The discussion is limited to operations in three dimensions.

Matrices and vectors (which are invariably written as matrices) are denoted by upper-case English letters; their components by lower-case English letters. Angles are given by lower-case Greek letters. Matrix transpose is denoted by the letter *t*.

E is the generic term for right-handed orthogonal coordinate systems. Individual systems are differentiated from each other by means of superscripts (e.g., E^C). Rather than give letter names to the various coordinate axes (*x*, *y*, *z* for instance), they are numbered. For example, the base vectors for E^C are given as follows:

$$E_1^C = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad E_2^C = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \quad E_3^C = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

Vectors are hallowed creatures whose characteristics, both intrinsic and in relation to other vectors, are not affected by choice (or even total absence) of coordinate systems. For computational purposes, however, the vector must be associated with some reference. When this is done, the vector is superscripted the same as the reference. For example, the vector *V*, when referenced to E^C , is written as V^C . The components of V^C are

$$V^C = \begin{bmatrix} v_1^C & v_2^C & v_3^C \end{bmatrix}^t$$



F70-09

Most situations utilize a multitude of coordinate systems, and the transformation of one into another is required for studying various pertinent relationships between them. This is accomplished by performing a series of coordinate-system rotations - first about one axis, then about another, and so on until the first system has been made to coincide with the second. These transformations are pure rotations; there are no lineal distortions of the space. Thus, three orthonormal transformations are available. They are denoted by $T_j(\zeta)$ where j ($= 1, 2, 3$) is the coordinate axis about which the rotation is made and ζ is the actual angle of rotation. Thus,

$$T_1(\zeta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c\zeta & s\zeta \\ 0 & -s\zeta & c\zeta \end{bmatrix} \quad T_2(\zeta) = \begin{bmatrix} c\zeta & 0 & -s\zeta \\ 0 & 1 & 0 \\ s\zeta & 0 & c\zeta \end{bmatrix}$$

$$T_3(\zeta) = \begin{bmatrix} c\zeta & s\zeta & 0 \\ -s\zeta & c\zeta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where s and c denote sine and cosine operations respectively. Note that $\det|T_j(\zeta)| = 1$ and that $T_j^{-1}(\zeta) = T_j^t(\zeta)$.

Multiple transformations such as

$$E^b = T_{j_n}(\zeta_n) \cdots T_{j_2}(\zeta_2) T_{j_1}(\zeta_1) E^a$$

where $j_k = 1, 2, \text{ or } 3$, $k = 1, \dots, n$, are, on occasion, coalesced into a single omnipotent transformation denoted simply by T , superscripted or otherwise depending on the level of confusion. Thus, the above can be written

$$E^b = TE^a$$



F70-09

and the components of a vector V in E^b can be derived, if V^a is known, simply by

$$V^b = \begin{bmatrix} v_1^b \\ v_2^b \\ v_3^b \end{bmatrix} = T \begin{bmatrix} v_1^a \\ v_2^a \\ v_3^a \end{bmatrix}$$

A common and useful operation between two vectors is their inner (or "dot") product. It is written and defined as

$$\langle A, B \rangle \triangleq \sum_{j=1}^3 a_j b_j$$



F70-09

Appendix B
PROGRAM LISTINGS

F70-09

PAGE 1 27OCT70

// JOB

27OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 OCEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR MAINLINE PROGRAM ... LBDO

*LIST ALL

*ONE WORD INTEGERS

*NAME LBDO

*IOCS(TYPEWRITER, KEYBOARD, 1132 PRINTER, CARD, DISK)

*TWO AXIS NULL SENSOR ALBEDO ANALYSIS PROGRAM 'LBDO'

VIM03

F70 69

```

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
PROGRAM NAME .....
'LBDO'      VERSION 1, MODIFICATION LEVEL 3
FUNCTION .....
THE PROGRAM COMPUTES THE ALBEDO PRODUCED RESPONSE OF A
TWO-AXIS SOLAR SENSOR AND THROUGH AN ITERATIVE NULL PROCESS,
BALANCES THIS OUTPUT AGAINST THE SUN-PRODUCED OUTPUT. HENCE,
THE POINTING ERROR DUE TO ALBEDO IS OBTAINED.

SUBPROGRAMS REQUIRED .....
RSPNS-COMPUTES THE OUTPUT OF THE SINGLE AXIS SENSOR GIVEN
THE ILLUMINATION INPUT ORIENTATION ANGLES.
TWEAK-GIVEN THE SENSOR OUTPUT, THIS ROUTINE COMPUTES THE
EQUIVALENT SOLAR VECTOR - SENSOR NULL AXIS MISALIGN-
MENT ANGLE, USED IN THE ITERATIVE NULLING SECTION.
ALBDO-COMPUTES THE ALBEDO FACTOR AT THE DIFFERENTIAL
RADIATING AREA'S LOCATION.
ROTJ- THESE ARE THREE ROUTINES WITH J = 1, 2, AND 3.
PERFORMS ORTHONORMAL COORDINATE TRANSFORMATIONS
ABOUT THE 1, 2, OR 3 AXES.
INNER-PERFORMS THE INNER PRODUCT OF TWO 3-VECTORS.
XNORM-NORMALIZES A 3-VECTOR TO UNITY AND RETURNS THE
ORIGINAL MAGNITUDE.
MESAG-RETRIEVES A MESSAGE STORED IN A DISK DATA-FILE AND
OUTPUTS IT ON THE SPECIFIED I/O DEVICE
LBDOZ-READS AN ALPHAMERIC WORD FROM THE KEYBOARD OR CARD
READER, RECOGNIZES THE WORDS 'CARD', 'KEYBOARD', AND
'QUIT'. SETS DATA-INPUT DEVICE ACCORDINGLY OR CAUSES
A PROGRAM EXIT.
ATAN2-FOUR-QUADRANT ARC-TANGENT.

INPUT VARIABLES .....
H  SATELLITE HEIGHT ABOVE EARTH (KILOMETERS)
XI  ANGLE BETWEEN TERMINATOR AND SATELLITE RADIUS
    VECTOR (DEGREES)
ALPHA-SENSOR AZIMUTH ANGLE (DEGREES)
EPSLN-SENSOR ELEVATION ANGLE (DEGREES)
RHO  SENSOR ROLL ANGLE (DEGREES)
ERRMX-MAXIMUM PERMISSIBLE CORRECTION ANGLE ... USED TO TURN
OFF ITERATION PROCESS (ARC-MINUTES)
ITMAX-MAXIMUM NUMBER OF ITERATIONS PERMITTED.
J  CARD INPUT ONLY ... J.NE. 0 IMPLIES A CONSTANTS
CARD IS INCLUDED IN THE DATA.
CONST-VECTOR OF REAL CONSTANTS FOR USE AS REQUIRED BY
VARIOUS SUBROUTINES ... 'RSPNS', 'TWEAK', AND
'ALBDO' FOR INSTANCE.
DATE THE DATE THE RUN(S) WERE MADE.
TITLE-RUN IDENTIFIER
NAME OPERATOR'S NAME.

```

VIM03

```

PAGE 3 27OCT70 TWO AXIS NULL SENSOR ALBEDO ANALYSIS PROGRAM 'LBDO'
C
C OUTPUT VARIABLES .....
C ITER NUMBER OF THE CURRENT ITERATION
C ALPHA-CORRECTED SENSOR AZIMUTH ANGLE (DEGREES).
C EPSLN-CORRECTED SENSOR ELEVATION ANGLE (DEGREES).
C ERROR-INCREMENTAL CORRECTION ANGLES IN AZIMUTH AND
C ELEVATION (DEGREES).
C TOTER-TOTAL NULL-AXIS INCREMENTAL DISPLACEMENT (ARC-MINUTES)
C SUM AZIMUTH AND ELEVATION CHANNEL SENSOR OUTPUTS DUE TO
C ALBEDO ONLY.
C
C STILL MISSING FROM THIS INFORMATION BLOCK IS A SECTION ON
C THE 'ATTRIBUTES' OF THE PROGRAM ... I.E., KEYBOARD OR CARD
C INPUT, NON-FATAL INTERRUPTION, CORE DUMP, DATA-SWITCH FUNCTIONS
C AND THE LIKE.
C
C CHARLES E. ROSE
C BALL BROTHERS RESEARCH CORPORATION
C BOULDER, COLORADO 80302
C 15 SEPTEMBER 1970
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
REAL N(3), INNER
C
C INTEGER ABORT, RUN, DATE(10), TITLE(40), YES, TYPE, CARD, P1132,
C 1 YESNO, HEADR(40), BLANK, DUMY1(40), STATE(258), COMON, PRCNT
C
C DATA YES, NO, TYPE, CARD, KEYBD, P1132, BLANK, PI, R, COMON, DA,
C 1 PRCNT
C 2/ 'YE', 'NO', 1, 2, 6, 3, 24040, 3.141593, 6378.166, 258,
C 3 7.61545E-5, Z6C40
C
C COMMON MODE, ABORT, RUN, ITER, ITMAX, DATE, NAME(15), TITLE
C 1, HEADR, DUMY1,
C 2 RHO, EPSLN, CE, SE, ALPHA, CA, SA, H, Q, XI, CXI, SXI, P(3),
C 3 PSTAR, N, A, SUM(2), CURNT(2), ERROR(2), TOTER, ERRMX, THTMX,
C 4 THETA, CTHIA, STHIA, PHIMN, PHI, PHIMX, CPHI, SPHI, OMEGA(2),
C 5 SIGMA(2), CBETA, CONST(8), PHOLD(3), HOLD
C
C EQUIVALENCE (STATE(1), MODE), (YESNO, KEY)
C
C DEFINE FILE 1(91,182,U,KK), 2(320,1,U,KK),
C 1 4(32,40,U,KK), 5(1,320,U,KK)
C
C SOUND(X,Y) = (X + Y - FLOAT(IFIX(X - Y + 0.5)) - 0.5)/2.0
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C SET-UP AND INITIALIZATION SECTION
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

```

C
C
C
TYPE INTRODUCTORY INFORMATION ON CONSOLE TYPEWRITER.

DO 90 J=13,27

CALL MESAG(J,TYPE)

CALL DATSW(0, KEY)

GO TO(95,90), KEY

CONTINUE

90
C
95
FACTR = PI/180.0

READ(5,1) (STATE(J), J=1,COMON)

MODE = 1

GO TO (150, 140, 100, 100, 100), ABORT

C
C
C
PREVIOUS RUN ABORTED BY DATA-SWITCH 13

100
CALL MESAG(1, TYPE)

110
READ(KEYBD, 1010) YESNO

IF(YES-YESNO) 120,130,120

120
IF(NO - YESNO) 110,150,110

C
C
C
RESUME OPERATION A PREVIOUSLY ABORTED RUN.

130
WRITE(P1132,1110) RUN, (DATE(J), J=1,10)

WRITE(P1132,1120)

GO TO(100, 100, 396, 330, 470), ABORT

C
C
C
PREVIOUS RUN ABORTED BECAUSE OF FAULTY DATA CARD

140
CALL MESAG(2, TYPE)

C
C
C
NORMAL EXIT ON PREVIOUS RUN. BEGIN A FRESH RUN.

150
CONTINUE

C
C
C
ENTER TODAY'S DATE

170
CALL MESAG(3, TYPE)

READ(KEYBD, 1000) (DUMY1(J), J=1,10)

C
C
C
TEST FOR A BLANK ENTRY

DO 180 J=1,10

IF(DUMY1(J) - BLANK) 220,180,220

180
CONTINUE

C
C
C
BLANK RECORD FOUND. PRINT INFORMATION ON LAST RUN.

WRITE(P1132, 1020) RUN, (DATE(J), J=1,10), (NAME(J), J=1,15),

1 (TITLE(J), J=1,40)

190
CALL MESAG(4, TYPE)

READ(KEYBD,1010) YESNO

IF(YESNO - NO) 200, 170, 200

200
IF(YESNO - YES) 190, 210, 190

PAGE 5 27OCT70 TWO AXIS NULL SENSOR ALBEDO ANALYSIS PROGRAM 'LBDO'

QUIT. WRITE MESSAGE OF GOODBYE.

```
CALL MESAG(5, TYPE)
CALL EXIT
```

'DATE' NOT BLANK. BEGIN A FRESH RUN.

DO 230 J=1,15

DATE(J) = DUMY1(J)

CONTINUE

CALL MESAG(6, TYPE)

```
READ(KEYBD, 1000) (NAME(J), J=1,15)
```

DATA INPUT SECTION

ENTER DATA-INPUT MODE

CALL LBDOZ(J)

GO TO(250,260,520), J

KEYBOARD INPUT SELECTED

MODE = 1

```
CALL MESAG(10, TYPE)
READ(KEYBD, 1000) (TITLE(J), J=1,40)
```

CALL MESAG'11, TYPE)

READ(KEYBD, 1050) H

READ(KEYBD, 1050) XI

READ(KEYBD, 1050) ALPHA

READ(KEYBD, 1050) EPSLN

```
READ(KEYBD, 1050) RHO
```

READ(KEYBD, 1050) ERRMX
CALL MSGAC(13, TYPE)

```
CALL MSGAG(12, TYPE)
READ(KEYVBD, 1060) ITMAX
```

```
READ(KEYBD, LOGO) 1 MAX
CALL MSGAG(30, TYPE)
```

CALL MSGAG(30,TYPE)

```
CALL MSGAB(STATYPE)
READ(KEYBD, 1130) /
```

IF(J*(9-J)) 270,270,254

READ(KEYBD,1050) CONST(

GO TO 252

CARD ENTRY SELECTED

MODE = 2

```
READ(CARD, 1000) (TITLE(J), J=1,40)
```

READ(CARD, 1070) H, XI, ALPHA, EPSLN, RHO, ERRMX, ITMAX, J

IF(J) 270,270,265

VIM03

PAGE 7 270CT70 TWO AXIS NULL SENSOR ALBEDO ANALYSIS PROGRAM 'LBDO'

334 PHI = BOUND(PHIMX, PHIMN)

336 PHI = PHI + 0.5

IF(PHI - PHIMX) 338,338,410

338 SPHI = SIN(PHI*FACTR)

CPHI = COS(PHI*FACTR)

C

C COMPUTE BOUNDARIES OF INTEGRATION REGION ... THETA-MAX

C

IF(1.0/(Q*CXI) - ABS(CPHI)) 340,340,342

340 THTMX = 90.0

GO TO 344

342 THTMX = ASIN(SQRT(SXI*SXI + CPHI*(2.0*CXI/Q-CPHI) - 1.0/(Q*Q))/
1 (SPHI*SXI))/FACTR

C

C BEGIN INTEGRATION BETWEEN -(THETA-MAX) AND +(THETA-MAX)

C

344 THETA = BOUND(THTMX, -THTMX)

346 THETA = THETA + 0.5

IF(THTMX - THETA) 336,360,360

C

C EVALUATE SENSOR OUTPUT IN TWO AXES. J = 1 FOR AZIMUTH SENSOR

J = 2 FOR ELEVATION SENSOR

C

C COMPUTE ALBEDO FACTOR

C

A = ALBDO(THETA, PHI)

IF(A) 346,346,370

C

370 CTHTA = COS(THETA*FACTR)

STHTA = SIN(THETA*FACTR)

C

C DEFINE 'N', THE VECTOR NORMAL TO THE DIFFERENTIAL RADIATING AREA

C

N(1) = SPMI*CTHTA

N(2) = SPMI*STHTA

N(3) = CPMI

C

C DEFINE AND NORMALIZE 'P', THE RADIATION VECTOR

C

P(1) = Q*CXI - SPMI*CTHTA

P(2) = -SPMI*STHTA

P(3) = Q*CXI - CPMI

CALL XNORM(P, PSTAR)

C

C TRANSFER 'P' TO 'PHOLD' ... 'PHOLD' TO REMAIN IN EARTH COORDINATES

C

DO 375 J=1,3

PHOLD(J) = P(J)

CONTINUE

375

C TRANSFORM 'P' INTO SENSOR COORDINATES

C

CALL ROT1(P, RHO*FACTR)

C

CALL ROT2(P, EPSLN*FACR)
CALL ROT3(P, ALPHA*FACR)

COMPUTE ENTRANCE ANGLES OF 'P' FOR ... (1), THE AZIMUTH SENSOR AND
(2), THE ELEVATION SENSOR

```
SIGMA(1) = ATAN2(-P(2), -P(1))/FACTR
OMEGA(1) = ASIN(-P(3))/FACTR
SIGMA(2) = ATAN2(+P(3), -P(1))/FACTR
OMEGA(2) = ASIN(-P(2))/FACTR
```

COMPUTE COS(BETA) ... 'BETA' = ANGLE BETWEEN 'P' AND 'N'

CBETA = INNER(N,PHOLD)

COMPUTE ALBEDO EFFECT AND ACCUMULATE FOR NEW OUTPUT.

DO 390 J=1.2

CURNT(J) = RSPNS(SIGMA(J), OMEGA(J), 0)

IF(CURNT(J)) 380,390,390

```

SUM(J) = SUM(J)
(PSTAR*PSTAR)
+ A*CURNT(J)*CBETA*CTHTA*SPHI*SPHI*DA/

```

CONTINUE

CORE-DUMP OF REAL COMMON ... CONTROLLED BY DSW 6 AND DSW 7

CALL CRDMP

B-10

170-09

ALIGNMENT CORRECTION AND BRANCHING SECTION

CHECK DATA SWITCHES FOR ABORT OR READOUT FUNCTIONS.

CALL DATSW(10, KEY)
GO TO(292,394), KEY
CALL MESAG(26, P1132)
GO TO 470

CALL DATSW(13,KEY)
GO TO(400, 396), KEY

```
CALL DATSW(5,KEY)
GO TO(398,346), KEY
HOLD = (PHI-PHIMN)*100.0/(PHIMX-PHIMN)
WRITE(1,1100) ITER, HOLD, PRCNT
GO TO 346
```

ABORT JOB ... SAVE STATE VECTOR.

```

PAGE 9 27OCT70 TWO AXIS NULL SENSOR ALBEDO ANALYSIS PROGRAM 'LBDO' VIM03

400 ABORT = 3
405 WRITE(5,1) (STATE(J), J=1,COMON)
      CALL MESAG(7,P1132)
      GO TO 530
C
C
C
410 COMPUTE SENSOR OUTPUT FROM SUN AND REQUIRED ROTATION ANGLES.
      OMEGA(1) = EPSLN
      SIGMA(1) = ALPHA
      CA = COS(ALPHA*FACTR)
      SA = SIN(ALPHA*FACTR)
      CE = COS(EPSLN*FACTR)
      SE = SIN(EPSLN*FACTR)
      OMEGA(2) = ASIN((-SA*CE)/FACTR)
      SIGMA(2) = ATAN2(SE, CA*CE)/FACTR
      TOTER = 0.0
      DO 420 J=1,2
      CURNT(J) = RSPNS(SIGMA(J), OMEGA(J), 0) + SUM(J)
      ERROR(J) = TWEAK(CURNT(J), OMEGA(J))
      TOTER = TOTER + ERROR(J)*ERROR(J)
      CONTINUE
      TOTER = SQRT(TOTER)*60.0
C
C
C
      ROTATE SENSOR
      ALPHA = ALPHA - ERROR(1)
      EPSLN = EPSLN - ERROR(2)
      WRITE(P1132,1040) IYER, ALPHA, EPSLN, (ERROR(J), J=1,2), TOTER
      WRITE(P1132,1090) (SUM(J), J=1,2)
C
C
C
      CHECK FOR CONVERGENCE
      IF(ERRMX - TOTER) 430, .50, 450
      NO CONVERGENCE
C
C
C
      CALL DATSW(13, KEY)
      GO TO (440, 300), KEY
      ABORT = 4
      GO TO 405
C
C
C
      CONVERGENCE
      CALL DATSW(13, KEY)
      GO TO (460, 470), KEY
      ABORT = 5
      GO TO 405
C
C
C
      SET UP FOR A NEW RUN
      GO TO (240, 500), MODE
C
C
      CURRENT INPUT MODE IS CARD

```

PAGE 10 27OCT70 TWO AXIS NULL SENSOR ANALYSIS PROGRAM 'LBDO03

C 500 CALL LBD0Z(J)
C GO TO(250,260,520,510), J
C
C FAULTY DATA CARD
C
C 510 ABORT = 2
CALL MESAG(8,P1132)
GO TO 530

QUIT

```

520  ABORT = 1
530  WRITE(5,1) (STATE(J), J=1,COMMON)
530  GO TO 210

```

[illegible]

FORMAT STATEMENTS

))))))

```

1000 FORMAT(40A2)
1010 FORMAT(A2)
1020 FORMAT(/' LAST RUN NUMBER...', I3/' DATE', I4X, 10A2/' OPERATOR',
1 10X, 15A2/' TITLE OF RUN', 6X, 40A2//)
1030 FORMAT('IRUN NUMBER', I4, T21, 40A2/20X, 15A2/20X, 10A2/' SENSOR
IDENTIFICATION ... ')

```

```

1035  FORMAT(
2 F6.3, 5X, 'XI =', F6.1, ' KILOMETERS', 5X, 'Q =',
3 ( ALPHA, EPSILON, RHO ) =, 3F10.1, ' ITERATION TURN-OFF ANGLE =',
4 F6.2, ' ARC-MINUTES' ) )

```

```

1040 FORMAT(I5, 5E20.4)
1050 FORMAT(F10.3)
1060 FORMAT(I3)
1070 FORMAT(6F10.3, I3, I3, I3, I1)
1080 FORMAT('ITER =', I4, 5X)
1090 FORMAT( 7X, 'ALBEDO OUTPUTS ... (AZIMUTH,ELEVATION)', 2E20.4/)
1100 FORMAT('ITER =', I3, F6.1, A2, 3X)
1110 FORMAT('CONTINUATION OF RUN', I4, ' ... BEGUN ', I0A2//)
1120 FORMAT(
1, I1X, 'ALPHA ERROR', 8X, 'EPSILON ERROR', 4X, 'NET ERROR (ARC-MIN)', //)

```

```

11130 FORMAT(1I1)
11140 FORMAT(8F10.3)
11150 FORMAT(' VECTOR OF COMMON CONSTANTS ...', 4E16.6/31X, 4E16.6/31X)

```

ONE

VARIABLE ALLOCATIONS	
MODE(IC)=7FFF	ABORT(IC)=7FFE
NAME(IC)=7FF0-7FE2	TITLE(IC)=7FE1-7FBA
CE(RC)=7F64	SE(RC)=7F62
Q(RC)=7F58	XI(RC)=7F56
N(RC)=7F48-7F44	A(RC)=7F42
ERRMX(RC)=7F32	THTMX(RC)=7F30
	RUN(IC)=7FFD
	HEADR(IC)=7FB9-7F92
	ALPHA(RC)=7F60
	CXI(RC)=7F54
	SUM(RC)=7F40-7F3E
	THETA(RC)=7F2E
	ITER(IC)=7FFC
	DUMY1(IC)=7F91-7F6A
	CA(RC)=7F5E
	SXI(RC)=7F52
	CURNT(RC)=7F3C-7F3A
	CIHTA(RC)=7F2C
	ITMAX(IC)=7FFB
	RHO(RC)=7F68
	SA(RC)=7F5C
	P(RC)=7F50-7F4C
	PSTAR(RC)=7F4A
	TOTER(RC)=7F34
	PHIMN(RC)=7F28
	DATE(IC)=7FFA-7FF1
	EPSLN(RC)=7F66
	H(RC)=7F5A

PHI(RC)=7F26 PHIMX(RC)=7F24 CPHI(RC)=7F22 SPHI(RC)=7F20 OMEGA(RC)=7F1E-7F1C SIGMA(RC)=7F1A-7F18
CBETA(RC)=7F16 CONST(RC)=7F14-7F06 PHOLD(RC)=7F04-7F00 HOLD(RC)=7EFE STATE(IC)=7FFF-7EFE YESNO(I)=001C
KEY(I)=001C FACTR(R)=001E PI(R)=0020 R(R)=0022 DA(R)=0024 YES(I)=0036
TYPE(I)=0037 CARD(I)=0038 P1132(I)=0039 BLANK(I)=003A COMON(I)=003B PRCNT(I)=003C
KK(I)=003D J(I)=003E KEYBD(I)=003F NO(I)=0040

STATEMENT ALLOCATIONS

BOUND=006D 1000=0094 1010=0097 1020=0099 1030=00C8 1035=00ED 1040=0142 1050=0146 1060=0148 1070=014A
1080=0150 1090=0157 1100=0170 1110=0179 1120=0191 1130=01BF 1140=01C1 1150=01C4 90=020A 95=0213
100=023C 110=0240 120=0248 130=0253 140=0279 150=027D 170=027D 180=02A6 190=02EE 200=02FD
210=0303 220=0308 230=0315 240=0338 250=0342 252=0393 254=03A1 260=03AD 265=03DE 270=03F4
300=048F 303=049F 305=04A5 310=04AB 320=04B9 322=04EC 324=04F6 326=04FD 328=0502 330=0506
332=050D 334=0511 336=0517 338=0524 340=05A 342=0550 344=0582 346=058D 360=059A 370=05A5
375=0605 380=06B3 390=06D2 392=06E7 394=06ED 396=06F7 398=0701 400=071D 405=0721 410=073E
420=07F0 430=084D 440=0857 450=085D 460=0867 470=086D 500=087E 510=0873 520=0888 530=088C

FEATURES SUPPORTED
ONE WORD INTEGERS
IOCS

CALLED SUBPROGRAMS

INNER	MESAG	DATSW	LBDCZ	FCOS	FSDN	RSPNS	ASIN	FSQRT	FABS	ALBDO	XNORM	ROT1	ROT2	ROT3
ATAN2	CRDMP	TWEEK	FADD	FADDX	FSDN	FSDN	FSDN	FSDN	FSDN	FSDN	FSDN	FSDN	FSDN	FSDN
FDVR	IFIX	FLOAT	TYPEZ	CARDZ	PRNTZ	SRED	SWRT	SCOMP	FDIV	FLD	FLDX	SIOF	SIOI	FSBR
SNR	SUBIN	SDFIO	SDRED	SDWRT	SDCOM	SDIX			SFIO	SIOFX	SIOIX			SUBSC

REAL CONSTANTS

•500000E 00=0048
•100000E 03=0054
•200000E 01=004A
•600000E 02=0056

INTEGER CONSTANTS

13=0058 27=0059
4=0062 6=0063
7=006C
0=005A
11=0064
5=005B
12=0065
1=005C
30=0066
10=005D
31=0067
2=005E
9=0068
3=005F
8=0069
15=0060
29=006A
40=0061
26=0068

CORE REQUIREMENTS FOR LBDO

COMMON 258 VARIABLES 72 PROGRAM 2142

END OF COMPILATION

PAGE 1 29OCT70 29OCT70 01

// JOB

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 OCEB OCEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

*EQUAT(RSPNS,RSPN3)

// DUP CORE-IMAGE GENERATION PACKAGE FOR PROGRAM LBDO

*DELETE LBDO
CART ID OCEB DB ADDR 4C70 DB CNT 01F0

*STORECI CD UA LBDO 0002
*LOCAL,TWEEK,RSPN3,ROT1,ROT2,ROT3,LBDOZ,MESAG,CRDMP
*FILES(1,LBDO1),(2,LBDO1),(4,LBDO3),(5,LBDO4)

FILES ALLOCATION
1 0178 005B OCEB LBDO1
2 0178 0001 OCEB LBDO1
4 01D7 0004 OCEB LBDO3
5 01D8 0001 OCEB LBDO4

STORAGE ALLOCATION
R 41 0128 (HEX) WDS UNUSED BY CORE LOAD
CALL TRANSFER VECTOR

FSIGN 1B02
FATAN 1A7E
OVERF 1A6C
PDUMP 18D4
INNER 18AA
ATAN2 1840
XNORM 17E4
ALBDO 17CE
FABS 17B6
ASIN 173F
FSQRT 16D0
FSIN 1656
FCOS 164E
DATSW 15DA
CRDMP 1BF5 LOCAL
MESAG 1C00 LOCAL
LBDOZ 1C34 LOCAL
ROT3 1C16 LOCAL
ROT2 1C16 LOCAL
ROT1 1C16 LOCAL
RSPN3 1C3C LOCAL
TWEEK 1BF6 LOCAL
LIBF TRANSFER VECTOR

HOLTB 1B57
SIOAF 0FB6
SIOAI 0FC3
FGETP 1A56
XMDS 1A3A

PAGE 2 29OCT70

EBCTB 1A37
HOLEZ 19C8
GETAD 19B8
PAUSE 19A2
FARC 1980
NORM 1956
SDI 0C37
FSUBX 0AFF
SDCOM 0CDA
SDWRT 0C63
FADDX 0808
FMPYX 161A
FLDX 0AE0
SNR 17C2
FDVR 17A0
FSTOX 0AC4
FMPY 161E
SIOFX 0FBA
SIOF 0F94
SCOMP 0F98
SIOIX 0F51
SWRT 0EBA
SIOI 0FBE
SRED 0EBF
SDIX 0C49
SUBSC 15FC
SDRED 0C5E
PRNTZ 1460
CARDZ 13B4
TYPEZ 134E
SFIO 0FD1
SDFIO 0CC2
FDIV 0BB2
FSBR 0B9A
FLOAT 0B8C
IFIX 0B60
FSUB 0AFA
FSTO 0AC8
FADD 0B05
FLD 0AE4
SUBIN 0AA4
SYSTEM SUBROUTINES
ILS04 00C4
ILS02 00B3
ILS01 1D00
ILS00 1D1B
FLIPR 1B8A

03DD (HEX) IS THE EXECUTION ADDR
CART ID 0CEB DB ADDR 4E90 DB CNT 01F0

PAGE 1 29OCT70
// JOB 29OCT70
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0CEB 0000
V2 M08 ACTUAL 8K CONFIG 8K
// FOR SENSOR RESPONSE DATA INPUT VEHICLE FOR LBDO SYSTEM.
*LIST ALL
*ONE WORD INTEGERS
*ORIGIN /0300
*NAME LBDO
*IOCS(TYPEWRITER, CARD, DISK, PLOTTER, KEYBOARD, 1132 PRINTER)
** SENSOR RESPONSE DATA CONVERTOR AND DISK-WRITING PROGRAM FOR 'LBDO'

PAGE 2 29OCT70 SENSOR RESPONSE DATA CONVERTOR AND DISK-WRITING PROGRAM FOR 'LBDO'

REAL MAX

```

INTEGER RAW(91,3), ZERO(320), TITLE(40), COUNT, ROUTE
DIMENSION DATA(91)
DEFINE FILE 1(91,182,U,K1), 2(91,320,U,K2), 3(320,1,U,K3)
DATA ZERO, K1, K2, K3, ROUTE /323*0, 1 /

```

```

THIS PROGRAM LOADS, CONVERTS, AND STORES-TO-DISK THE RAW DATA
WHICH DEFINES THE SENSOR ANGULAR CHARACTERISTIC FOR USE IN
PROGRAM 'LBDO' .

```

```

THE DATA IS STORED AS SENSOR RESPONSES AT EVENLY SPACED (EVERY
0.5 DEGREES) AZIMUTH AND ELEVATION ANGLES. THERE ARE 91 RECORDS
EACH HOLDING DATA TAKEN AT A CONSTANT ELEVATION ANGLE. EACH SUCH
RECORD CONTAINS UP TO 91 SENSOR RESPONSES TAKEN AT 0.5 DEGREE
AZIMUTH ANGLE INCREMENTS. THUS, THE SENSOR ANGULAR CHARACTERISTIC
IS MAPPED OVER THE REGION.....

```

```

0 .LE. BETA .LE. 45 DEGREES
0 .LE. GAMMA .LE. 45 DEGREES

```

```

THE THREE DEFINED FILES ARE ALL EXTERNALLY REFERENCED TO DISK-
DATA-FILE 'LBDO1' BY MEANS OF AN *FILES RECORD

```

```

THE DATA-PLOT PHASE CAN BE INTERRUPTED BY TURNING DATA SWITCH 15
ON. THE RECORD AND DATA POINT NUMBERS WILL BE PRINTED AND THE
PROGRAM WILL LOOP UNTIL SWITCH 15 IS TURNED OFF.

```

```

CHARLES E. ROSE
BALL BROTHERS RESEARCH CORPORATION
BOULDER, COLORADO 80302
4 AUGUST 1970
17 AUGUST 1970 ... REVISION A

```

```

FUNCTION SELECT POINT.

```

```

WRITE(1,1070)
WRITE(1,1080)

```

```

PAUSE 80
DO 90 J=1,5
CALL DATSW(J,KEY)
GO TO(95,90), KEY

```

```

CONTINUE

```

```

GO TO 80

```

```

GO TO(100,200,300,400,500), J

```

```

NULL 'LBDO1' DATA FILE

```

```

ROUTE = 1

```

```

WRITE(1,1000)

```

```

DO 101 J=1,91

```

```

PAGE 3 29OCT70 SENSOR RESPONSE DATA CONVERTOR AND DISK-WRITING PROGRAM FOR 'LBDO1'

101 WRITE(2,J) (ZERO(K), K=1,320)
C CONTINUE
C READ DATA HEADER, PLACE IN FIRST SECTOR OF 'LBDO1' FILE
C WRITE(1,1010)
C READ(2,1040) (TITLE(J), J=1,40)
DO 105 J=1,40
WRITE(3,J+280) TITLE(J)
105 CONTINUE
C DATA ENTRY
C DO 140 J=1,91
C READ(2,1020) (RAW(K,3), RAW(K,1), RAW(K,2), K=1,91)
C DATA CONVERSION
C DO 120 COUNT = 1,91
C DATA(COUNT) = FLOAT(RAW(COUNT,1))*10.0**(-RAW(COUNT,2))
C IF(RAW(COUNT,3)) 106,120,106
106 DATA(COUNT) = -DATA(COUNT)
120 CONTINUE
C CONVERTED DATA TO 'LBDO1' FILE
C WRITE(1,J) (DATA(K), K=1,91)
140 CONTINUE
C GO TO 80
C PLOT DATA, BY RECORDS, FOR VALIDITY CHECK
C
C WRITE(1,1050)
C MAX = -1.0E19
C DO 520 J=1,91
C READ(1,J) (DATA(K), K=1,91)
C DO 520 K=1,91
C IF(MAX - DATA(K)) 510,520,520
510 MAX = DATA(K)
520 CONTINUE
C WRITE(1,1140) MAX
C DO 530 J=1,40
C READ(3,J+280) TITLE(J)
530 CONTINUE
C CALL GRAF(MAX, TITLE)
C DO 170 J=1,91
C READ(1,J) (DATA(K), K=1,91)
C CALL FPLOT(1, 0.0, DATA(1))
C CALL DOWN
C DO 160 K=2,91
C CALL FPLOT(0, FLOAT(K-1)/2.0, DATA(K))
C CALL DATSW(15,KEY)
C GO TO(151,160), KEY

```

```

PAGE 4 29OCT70 SENSOR RESPONSE DATA CONVERTOR AND DISK-WRITING PROGRAM FOR 'LBDO'

151 WRITE(1,1060) J, K
152 CALL DATSW(15,KEY)
GO TO(152,160), KEY
160 CONTINUE
CALL UP
170 CONTINUE
CALL SCALF(1.0, 1.0, 0.0, 0.0)
CALL FPLOT(0, 5.0, 0.0)
GO TO 80

C SINGLE RECORD PLOT PHASE
C
C
200 GO TO(210,240), ROUTE
210 ROUTE = 2
MAX = -1.0E19
DO 230 J=1,91
READ(1,J) (DATA(K), K=1,91)
DO 230 K=1,91
IF(MAX - DATA(K)) 220,230,230
220 MAX = DATA(K)
230 CONTINUE
DO 235 J=1,40
READ(3,J+280) TITLE(J)
235 CONTINUE
CALL GRAF(MAX, TITLE)
240 WRITE(1,1090)
READ(6,1100) J
IF(J) 80,80,250
250 READ(1,J) (DATA(K), K=1,91)
CALL UP
CALL FPLOT(-2, 0.0, DATA(1))
DO 260 K=2,91
CALL FPLOT(0, FLOAT(K-1)/2.0, DATA(K))
260 CONTINUE
WRITE(3,1120) (TITLE(K), K=1,40), J
WRITE(3,1130) (DATA(K), K=1,91)
GO TO 240

C SINGLE RECORD RAW-DATA INPUT PHASE
C
C
300 WRITE(1,1110)
READ(6,1100) J
READ(2,1020) (RAW(K,1), RAW(K,2), K=1,91)
DO 320 K=1,91
DATA(K) = FLOAT(RAW(K,1))*10.0+*(-RAW(K,2))
IF(RAW(K,3)) 310,320,310
310 DATA(K) = -DATA(K)
320 CONTINUE
WRITE(1,J) (DATA(K), K=1,91)
GO TO 80

C EXIT PHASE
C
C

```

400 WRITE(1,1030)
CALL EXIT

C
1000 FORMAT('DATA FILE ', 7H'LBDO1', ' NOW BEING CLEARED', 5X)
1010 FORMAT('FILE CLEARED. READ, CONVERT, AND STORE RAW DATA.', 5X)
1020 FORMAT(13(I1, I4, I1))
1030 FORMAT('COMPLETE. RETURN TO MONITOR.', 5X)
1040 FORMAT(40A2)
1050 FORMAT('PLOT DATA', 5X)
1060 FORMAT('RECORD', 13, 5X, 'NUMBER', 13, 5X)
1070 FORMAT('DATA SWITCH 1 ... COMPLETE FILE INPUT AND PLOT', 12X, '2 ...
1 SINGLE RECORD PLOT', 12X, '3 ... SINGLE RECORD RAW-DATA INPUT', 12X
2, '4 ... EXIT TO MONITOR', 12X, '5 ... PLOT ENTIRE FILE')
1080 FORMAT('SET DATA SWITCH')
1090 FORMAT('ENTER RECORD NUMBER TO BE PLOTTED ... 12')
1100 FORMAT(12)
1110 FORMAT('ENTER RECORD NUMBER OF NEW DATA ... 12')
1120 FORMAT('1', 20X, 40A2//55X, 'RECORD', 13////)
1130 FORMAT(13(2X, F6.3, 1X))
1140 FORMAT('MAXIMUM VALUE =', F8.4, 5X)
END

VARIABLE ALLOCATIONS

DATA(R) = 03CA-0316 MAX(R) = 03CC RAW(I) = 04E4-03D4 ZERO(I) = 0624-04E5 TITLE(I) = 064C-0625 COUNT(I) = 064D
ROUTE(I) = 064E K1(I) = 064F K2(I) = 0650 K3(I) = 0651 J(I) = 0652 KEY(I) = 0653
K(I) = 0654

STATEMENT ALLOCATIONS

1000 = 0673 1010 = 068A 1020 = 06A5 1030 = 06AB 1040 = 06BC 1050 = 06BF 1060 = 06C7 1070 = 06D4 1080 = 072C 1090 = 0736
1100 = 074C 1110 = 074E 1120 = 0763 1130 = 0775 1140 = 077C 80 = 07A5 90 = 07C4 100 = 07CD 101 = 07F0
105 = 0828 106 = 0890 120 = 089A 140 = 08BA 500 = 08C5 510 = 08F8 520 = 0901 530 = 092D 151 = 098C 152 = 0994
160 = 099E 170 = 09A9 200 = 09BF 210 = 09C5 220 = 09F8 230 = 0A01 235 = 0A27 240 = 0A34 250 = 0A41 260 = 0A8A
300 = 0AC5 310 = 0B29 320 = 0B33 400 = 0B55

FEATURES SUPPORTED

ONE WORD INTEGERS
ORIGIN
IOCS

CALLED SUBPROGRAMS

DATSW GRAF FPLOT DOWN UP SCALF FSUBX FMPY FDIV FLD FLDX FSTO FSTOX FAXI FLOAT
TYPEZ CARDZ PRNTZ WCHR1 SRED SDCOM SDFX SWRT SDIX SFIO SIOF SIOI SUBSC PAUSE SNR
SDFIO SDRED SDWRT

REAL CONSTANTS

.100000E 02=065A .100000E 20=065C .000000E 00=065E .200000E 01=0660 .100000E 01=0662 .500000E 01=0664

INTEGER CONSTANTS

1=0666 80=0667 5=0668 91=0669 2=066A 320=066B 40=066C 3=066D 280=066E 0=066F
15=0670 6=0671 128=0672

CORE REQUIREMENTS FOR LBDO

COMMON 0 VARIABLES 858 PROGRAM 1280

END OF COMPILATION

// DUP

*DELETE LBD02
 CART ID OCEB DB ADDR 4C90 DB CNT 0200

*STORECI WS UA LBD0D 0001
 *FILES(1,LBD01),(2,LBD01),(3,LBD01)
 FILES ALLOCATION

1 017B 005B OCEB LBD01
 2 017B 005B OCEB LBD01
 3 017B 0001 OCEB LBD01

STORAGE ALLOCATION

R 40 0097 (HEX) ADDITIONAL CORE REQUIRED
 R 43 0176 (HEX) ARITH/FUNC SOCIAL WD CNT
 R 44 079C (HEX) FI/O, I/O SOCIAL WD CNT
 R 41 006A (HEX) WDS UNUSED BY CORE LOAD
 CALL TRANSFER VECTOR

POINT 1652

FCOS 17DC SOCIAL 1
 FSIN 17E4 SOCIAL 1

FGRID 1544

FCHAR 150B

TAP 12E0

SCALF 1316

UP 1301

DOWN 1306

FPLOT 12B6

GRAF 11CA

DATSW 18F2 SOCIAL 1

LIBF TRANSFER VECTOR

FGETP 1852 SOCIAL 1

FCHRX 0F1B

FSUB 112C

FMPYX 10F8

HOLTB 1E8B SOCIAL 2

FRULE 139B

PLOTX 159B

FMOVE 13C1

SUBIN 14DE

FARC 14BC

XMDS 14A0

FDVR 17C6 SOCIAL 1

NORM 1476

PLOTI 1464

XYPLT 13FC

FINC 13EB

FADD 1137

EBCTB 137F

HOLEZ 1E22 SOCIAL 2

GETAD 1E12 SOCIAL 2

PAGE	7	290CT70
IFIX	131A	
SIOFX	17F2	SOCAL 2
SIOI	17F6	SOCAL 2
FDIV	176E	SOCAL 1
SIOF	17CC	SOCAL 2
FSUBX	1131	
SDRED	08B4	
SDFX	089B	
SNR	1762	SOCAL 1
FLDX	10A0	
FSTOX	1084	
FMPY	10FC	
FAXI	1718	SOCAL 1
FLOAT	10EE	
SIOIX	1789	SOCAL 2
SRED	16F7	SOCAL 2
SDCOM	0C30	
SDIX	0B9F	
SUBSC	10D0	
SDWRT	0BB9	
PAUSE	10BA	
SCOMP	17D0	SOCAL 2
SWRT	16F2	SOCAL 2
FSTO	1088	
FLD	10A4	
WCHRI	0E10	
PRNTZ	1C98	SOCAL 2
CARDZ	1BEC	SOCAL 2
TYPEZ	1B86	SOCAL 2
SFIO	1809	SOCAL 2
SDFIO	0C18	
SYSTEM SUBROUTINES		
ILS04	00C4	
ILS03	1E93	
ILS02	00B3	
ILS01	1EB0	
ILS00	1ECB	
FLIPR	1688	
0788 (HEX) IS THE EXECUTION ADDR		
CART ID	0CEB	DB ADDR 4E80 DB CNT 0200

F70-09

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR PROGRAM LBDOM

*LIST ALL

*UNE WORD INTEGERS

*IOCS(TYPEWRITER, KEYBOARD, 1132 PRINTER, DISK)

*NAME LBDOM

'LBDOM' * MESSAGE FILE MANIPULATOR FOR THE 'LBDOM' ANALYSIS SYSTEM

*ORIGIN /0300

B-23

B-24

PAGE 3 290CT70 'LBDO' ... MESSAGE FILE MANIPULATOR FOR THE 'LBDO' ANALYSIS SYSTEM

```

220 IF(MODE - MOVE) 260,230,260
C
C MOVE MESSAGES
C
230 WRITE(1,1080)
READ(6,1090) M,N
IF(M*(33-M)) 100,100,240
IF(N*(33-N)) 100,100,250
240 READ(4,M) (MESAG(J), J=1,39), COUNT
250 WRITE(4,N) (MESAG(J), J=1,39), COUNT
GO TO 230
C
C COMPUTE RECORD LENGTHS ... THEN EXIT
C
260 DO 290 J=1,32
READ(4,J) (MESAG(N), N=1,39)
DO 270 N=1,39
COUNT = 40 - N
IF(MESAG(COUNT) - BLANK) 280,270,280
270 CONTINUE
COUNT = 0
280 WRITE(4,J) (MESAG(N), N=1,39), COUNT
290 CONTINUE
C
CALL EXIT
C

```

1000 FORMAT('DO YOU WISH TO ALTER, LIST, MOVE, OR EMPTY THE MESSAGE FILE', A2)

```

1010 FORMAT(A2)
1020 FORMAT('MESSAGE NUMBER', A2)
1030 FORMAT(I2)
1040 FORMAT('MESSAGE', A2)
1050 FORMAT(39A2)
1060 FORMAT('LISTING OF THE CURRENT CONTENTS OF DATA-FILE', 8H 'LBDO3',
1, //)
1070 FORMAT(1H0, I2, 5X, 39A2)
1080 FORMAT('XX TO YY')
1090 FORMAT(I2, 4X, I2)
END

```

VARIABLE ALLOCATIONS

```

MESAG(I) = 032E-0308 ALTER(I) = 032F
K4(I) = 0334 MODE(I) = 0335
MOVE(I) = 033A M(I) = 033B

```

EMPTY(I) = 0330
N(I) = 0336

QMARK(I) = 0331
J(I) = 0337

BLANK(I) = 0332
N39(I) = 0338

COUNT(I) = 0333
LIST(I) = 0339

STATEMENT ALLOCATIONS

```

1000 = 0347 1010 = 0368 1020 = 036A 1030 = 0374 1040 = 0376 1050 = 037D 1060 = 0380 1070 = 03A0 1080 = 03A7 1090 = 03AD
1100 = 03CA 1110 = 03DB 1120 = 03EF 1130 = 0426 1140 = 042C 1150 = 0450 1160 = 0469 1170 = 0474 1180 = 047A 1190 = 0487
1210 = 04B1 1220 = 04BC 1230 = 04C2 1240 = 04D6 1250 = 04DF 1260 = 0512 1270 = 0541 1280 = 054E 1290 = 0567

```

FEATURES SUPPORTED
ONE WORD INTEGERS
ORIGIN
IOCS

PAGE 4 29OCT70 'LBDOM' ... MESSAGE FILE MANIPULATOR FOR THE 'LBDOM' ANALYSIS SYSTEM

CALLED SUBPROGRAMS
FLD FSTO TYPEZ PRNTZ SRED SWRT SDCOM
SDIX SDI

INTEGER CONSTANTS
1=033E 6=033F 33=0340 39=0341 4=0342 3=0343 32=0344 0=0345 40=0346

CORE REQUIREMENTS FOR LBDOM
COMMON 0 VARIABLES 62 PROGRAM 564

END OF COMPILATION

// DUP

*DELETE LBDOM
CART ID OCEB DB ADDR 48D0 DB CNT 0080

*STORECI WS UA LBDOM 0001
*FILES(4,LBD03)
FILES ALLOCATION
4 01D7 0004 OCEB LBD03

STORAGE ALLOCATION
R 41 0F74 (HEX) WDS UNUSED BY CORE LOAD
LIBF TRANSFER VECTOR
HOLTB 101B
NORM OFBE
EBCTB OFBB
HOLEZ OF4C
GETAD OF3C
FLOAT OF32
IFIX OF06
PAUSE OFE0
SDRED 05CC
SDCOM 0648
SDI 05A5
SDIX 05B7
SDWRT 05D1
SIOIX 08BF
SUBSC 0ED2
SRED 082D
SCOMP 0906
SIOI 092C
SWRT 0828
FSTO 0EAO
FLD 0EBC
PRNTZ 0D22
TYPEZ 0CBC
SFIO 093F
SDFIO 0630

SYSTEM SUBROUTINES
ILS04 00C4
ILS02 00B3
ILS01 1020

F70-09

PAGE 5 29OCT70
03B1 (HEX) IS THE EXECUTION ADDR
CART ID 0CEB DB ADDR 4FDO DB CNT 0080

B-27

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// ASM

*LIST

*PRINT SYMBOL TABLE

*OVERFLOW SECTORS ,8

F70-09

B-28

TYPEWRITER INPUT-OUTPUT MACRO

PAGE 2 29OCT70

```

00001 *****
00002 *
00003 * THIS MACRO UTILIZES 'TYPE0' FOR
00004 * INPUT (X=1) OR OUTPUT (X=2) TO OR
00005 * FROM THE I/O AREA NAMED 'LIST'.
00006 *
00007 *****
00008 *
00009 *****
00010 MAC
00011 LABEL TYPE X,LIST
00012 LABEL LIBF TYPE0
00013 AIF (X EQ 1),INPUT TEST FOR INPUT
00014 AIF (X EQ 2),OUTPT TEST FOR OUTPUT
00015 DC /30FE TRAP INVALID 'X'
00016 MDX *-2
00017 INPUT ANOP
00018 DC /1000 READ FROM KEYBD
00019 AGO GEIST
00020 OUTPT ANOP
00021 DC /2000
00022 GEIST ANOP
00023 DC
00024 LIBF BUSY WATCH
00025 DC TYPE0
00026 MDX /0000
00027 *-3
00028 MEND

```

```

00027 *****
00028 *
00029 *
00030 *
00031 *
00032 *
00033 *
00034 *
00035 *
00036 *
00037 *
00038 *
00039 *
00040 *****
00041 *
00042 *****
00043 *
00044 *
00045 *
00046 *
00047 *
00048 *
00049 *
00050 *
00051 *
00052 *
00053 *
00054 *
00055 *
00056 *
00057 *
00058 *
00059 *
00060 *
00061 *
00062 *
00063 *
00064 *
00065 *
00066 *
00067 *
00068 *
00069 *
00070 *
00071 *
00072 *
00073 *
00074 *

THIS MACRO UTILIZES 'DISK1' FOR A
VARIETY OF READ/WRITE OPTIONS,
DEPENDING ON THE PARAMETER 'X' ...
X = 1 READ
X = 2 WRITE, NO READ-BACK CHECK
X = 3 WRITE, READ-BACK CHECK
X = 4 WRITE, IMMEDIATE
X = 5 SEEK
'AREA' IS THE I/O AREA, 'ERROR' IS
THE APPLICABLE ERROR ROUTINE.

*****
X,AREA,ERROR
DISK1
(X EQ 1),READ
(X EQ 2),NRBC
(X EQ 3),RBC
(X EQ 4),IMM
(X EQ 5),SEEK
/30FF TRAP INVALID 'X'
*-2
/1000
GEIST
/2000
GEIST
/3000
GEIST
/4000
GEIST
/5000
AREA
(X EQ 4),BUSY'
ERROR
DISK1
/0000
AREA
BUSY'
MEND

LABEL DISK
LABEL LIBF
AIF
AIF
AIF
AIF
AIF
AIF
DC
MDX
ANOP
DC
AGO
ANOP
DC
AGO
ANOP
DC
AGO
ANOP
DC
ANOP
DC
ANOP
DC
ANOP
DC
AIF
DC
BUSY' LIBF
DC
DC
DC
MDX
MEND

```


MOD-LEVEL CHANGER ... LBDO SYSTEM.

PAGE 5 29OCT70

0317	20	23A17170	00126	+	LIBF	TYPE0	
0318	0	0000	00127	+	DC	/0000	
0319	0	70FD	00128	+	MDX	*-3	
031A	20	225C5144	00129		LIBF	SPEED	CONVERT TO EBCDIC
031B	0	0010	00130		DC	/0010	
031C	0	0343	00131		DC	MODLV+1	
031D	0	0344	00132		DC	HOLD	
031E	0	0001	00133		DC	1	
031F	00	C4000344	00134		LD	L HOLD	ADJUST FORMAT
0321	0	1808	00135		SRA	8	
0322	00	84000341	00136		A	BLANK	
0324	00	D40003DF	00137		STO	SECTR+129	
0326	20	04262481	00138		DISK	4,SECTR:	
			00139	+	LIBF	DISK1	
			00140	+	AIF	(4 EQ 1),READ	
			00141	+	AIF	(4 EQ 2),NRBC	
			00142	+	AIF	(4 EQ 3),RBC	
			00143	+	AIF	(4 EQ 4),IMM	
			00144	+	ANOP		
0327	0	4000	00145	+	DC	/4000	
			00146	+	AGO	GEIST	
0328	0	035E	00147	+	ANOP		
			00148	+	DC	SECTR	
			00149	+	AIF	(4 EQ 4),BUSY2	
0329	20	04262481	00150	+	LIBF	DISK1	
032A	0	0000	00151	+	DC	/0000	
032B	0	035E	00152	+	DC	SECTR	
032C	0	70FC	00153	+	MDX	BUSY2	
032D	00	13084594	00154		LINK	LBDO	
			00155	*			
0331	0	0000	00156		ERROR	DC	*--*
0332	0	300F	00157		DC	/300F	
0333	00	4C800331	00158		BSC	I ERROR	

PAGE 6 29OCT70

Address	Operation	Source	Destination	Comments
00159	*			
00160	*			
00161	*			
00162	*			CONSTANTS AREA, BUFFERS, ETC.
00163	*			
00164	*			
00165	*			
00166	*			
00167	*			
00168	N320	DC	320	
00169	BLANK	DC	/4000	
00170	MODLV	DC	1	
00171	BSS		1	
00172	HOLD	BSS	1	
00173	MESAG	DC	24	
00174	DMES			REENTER NEW MODIFICATION LEVEL (SIN'
00175	DMES			GLE DIGIT)'45'E
00176	SECTR	DSA	LBDO3	
00177	BSS		319	
00178	END		BEGIN	

B-34

F70-09

PAGE 1 29OCT70
// JOB 29OCT70
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 OCEB 0000
V2 M08 ACTUAL 8K CONFIG 8K
// FOR FUNCTION RSPNS
*LIST ALL
*ONE WORD INTEGERS
** TWO-DIMENSIONAL INTERPOLATION ROUTINE FOR 'LBDD'

```

PAGE 2 29OCT70 TWO-DIMENSIONAL INTERPOLATION ROUTINE FOR 'LBDO'
      FUNCTION RSPNS(ALPHA, OMEGA, KROUT)
      CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      C THIS ROUTINE ACCEPTS, AS INPUT, THE ON AND OFF AXIS ILLUMINATION C
      C ANGLES (ALPHA AND OMEGA) FOR A SENSOR, BRACKETS THESE VALUES C
      C WITH TABULATED POINTS ON DISK-FILE 'LBDO1', AND THEN PERFORMS A C
      C LINEAR, TWO-DIMENSIONAL INTERPOLATION ON THE DATA TO ESTIMATE C
      C THE SENSOR RESPONSE AT (ALPHA, OMEGA). C
      C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C C
      C CHARLES E. ROSE C
      C BALL BROTHERS RESEARCH CORPORATION C
      C BOULDER, COLORADO 80302 C
      C 29 AUGUST 1970 C
      C CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      C INTEGER HEADR(40) C
      C DIMENSION DUMMY(91), Z(2,2), NDUMMY(40) C
      C COMMON IDUMMY(70), HEADR C
      C EQUIVALENCE (Z(1,1),Z11), (Z(1,2),Z12), (Z(2,1),Z21), (Z(2,2),Z22) C
      C 1, (HEADR(1), NDUMMY(1)) C
      C IF(KROUT) 70,90,70 C
      C DO 80 J=1,40 C
      C READ(2,J+280) NDUMMY(J) C
      C CONTINUE C
      C WRITE(3,1000) (HEADR(J), J=1,40) C
      C GO TO 160 C
      C X = ABS(ALPHA) C
      C Y = ABS(OMEGA) C
      C CHECK FOR OUT-OF-BOUNDS DATA C
      C IF(45.0 - X) 110,110,100 C
      C IF(45.0 - Y) 110,110,120 C
      C RSPNS = 0.0 C
      C GO TO 140 C
      C DATA IS WITHIN BOUNDS, PROCEED TO BRACKET ALPHA AND OMEGA C
      C NREC = IFIX(2.0*Y) + 1 C
      C NPT = IFIX(2.0*X) + 1 C
      C READ BRACKETING RESPONSE POINTS FROM DATA-FILE 'LBDO1' C
      C DO 130 J=1,2 C
      C READ(1,NREC+J-1) (DUMMY(K), K=1,91) C
      C DO 130 K=1,2 C
      C INDEX = NPT+K-1 C
      C Z(K,J) = DUMMY(INDEX) C
      C CONTINUE C
      C 130 C

```

```

PAGE 3 290CT00 TWO-DIMENSIONAL INTERPOLATION ROUTINE FOR 'LBDO'
C SHIFT ORIGIN OF ALPHA AND OMEGA, THEN PERFORM INTERPOLATION.
C
X = X - FLOAT(NPT-1)/2.0
Y = Y - FLOAT(NREC-1)/2.0
RSPNS = (Z21-Z11)*X/0.5 + (Z11-Z21-Z12+Z22)*X*Y/0.25 +
1 (Z12-Z11)*Y/0.5 + Z11
C SET ALGEBRAIC SIGN OF RESPONSE ACCORDING TO SIGN OF ALPHA
C
140 IF(ALPHA) 150,160,160
150 RSPNS = -RSPNS
160 RETURN
C
1000 FORMAT('+', 26X, 40A2/)
END
VARIABLE ALLOCATIONS
IDUMY(IC)=7FFF-7FBA HEADR(IC)=7FB9-7F92 NDUMY(IC)=7FB9-7F92 Z11(R)=0006 Z12(R)=0002
Z21(R)=0004 Z22(R)=C000 RSPNS(R)=C008 DUMMY(R)=008E-000A X(R)=00C0 Y(R)=00C2
J(I)=00C8 NREC(I)=00C9 NPT(I)=00CA K(I)=00CB INDEX(I)=00CC
STATEMENT ALLOCATIONS
1000 =00E0 70 =00F5 80 =0109 90 =012B 100 =013C 110 =0143 120 =0149 130 =01A3 140 =01FD 150 =0202
160 =0207
FEATURES SUPPORTED
ONE WORD INTEGERS
CALLED SUBPROGRAMS
FABS FADD FSUB FMPY FDIV FLD FLDX FSTO FSTOX FSBR IFIX FLOAT SWRT SCOMP SIOIX
SUBSC SNR SUBIN SDRED SDFX SDIX
REAL CONSTANTS
.450000E 02=00D0 .000000E 00=00D2 .200000E 01=00D4 .500000E 00=00D6 .250000E 00=00D8
INTEGER CONSTANTS
1=00DA 40=00DB 2=00DC 280=00DD 3=00DE 91=00DF
CORE REQUIREMENTS FOR RSPNS
COMMON 110 VARIABLES 208 PROGRAM 316
RELATIVE ENTRY POINT ADDRESS IS 00E7 (HEX)
END OF COMPILATION
// DUP
*DELETE RSPNS
CART ID 0CEB DB ADDR 4B44 DB CNT 0017
*STORE WS UA RSPNS
CART ID 0CEB DB ADDR 5060 DB CNT 0017

```

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 OCEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR FUNCTION RSPN3 PIECEWISE LINEAR APPROXIMATION

*LIST ALL

*ONE WORD INTEGERS

**RSPN3: ... PIECEWISE LINEAR APPROXIMATION OF SENSOR CHARACTERISTIC

```
FUNCTION RSPN3 (SIGMA, OMEGA, K)  
COMMON IDUMY(234), CONST(8)
```

```
IF(K) 80,90,80
WRITE(3,1000)
GO TO 360
```

```

C
200 X = ABS(OMEGA)
    IF(27.0-X) 170,170,210
210 IF(11.2-X) 220,220,230
220 W = (27.0-X)*1.25/15.8
    GO TO 300
230 W = 1.0
C
300 RSPN3 = W*RSPN3
    IF(SIGMA) 310,170,320
310 RSPN3 = -RSPN3

```

THIS SECTION PERMITS THE SENSOR RESPONSE FUNCTION TO BE SEPARATED. BY SETTING 'CONST(1)', 'GT.' OR '.LT. 0', THE SENSOR OUTPUT CAN BE MADE IDENTICALLY ZERO FOR NEGATIVE OR POSITIVE ANGLES RESPECTIVELY. IF 'CONST(2)' = 0, THE ENTIRE FUNCTION IS RETAINED.

```

320 IF(CONST(1)) 330,360,350
330 IF(RSPN3) 36Q,360,340
340 RSPN3 = 0.0
GO TO 360

```

```

PAGE 3 290CT70 'RSPN3' ... PIECEWISE LINEAR APPROXIMATION OF SENSOR CHARACTERISTIC

350 IF(RSPN3) 340,360,360
360 RETURN
1000 FORMAT('+', 26X, 'PIECEWISE LINEAR APPROXIMATION OF THE SS1090 SEN
150R CHARACTERISTIC')
END
VARIABLE ALLOCATIONS
IDUMY(1C)=7FFF-7F16 CONST(RC)=7F14-7F06 RSPN3(1R )=0000 X(1R )=0002 W(1R )=0004

STATEMENT ALLOCATIONS
1000 =0025 80 =005A 90 =0060 100 =006C 110 =0076 120 =007D 130 =0083 140 =008A 150 =00A0 160 =00A7
170 =0083 200 =008B 210 =00C7 220 =00CE 230 =00DA 300 =00DE 310 =00EB 320 =00F0 330 =00F9 340 =00FE
350 =0104 360 =0109

FEATURES SUPPORTED
ONE WORD INTEGERS

CALLED SUBPROGRAMS
FABS FADD FSUB FMPY FDIV FLD FLDX FSTO SWRT SCOMP SNR SUBIN

REAL CONSTANTS
.600000E 01=000A .500000E 01=000C .126000E 02=000E .146000E 02=0010 .125000E 01=0012 .200000E 01=0014
.316000E 02=0016 .170000E 02=0018 .000000E 90=001A .270000E 02=001C .112000E 02=001E .159000E 02=002C
.100000E 01=0022

INTEGER CONSTANTS
3=0024

CORE REQUIREMENTS FOR RSPN3
COMMON 250 VARIABLES 10 PROGRAM 260

RELATIVE ENTRY POINT ADDRESS IS 004C (HEX)

END OF COMPILATION

// DUP

*DELETE RSPN3
CART ID 0CEB DB ADDR 4844 DB CNT 0013

*STORE WS UA RSPN3
CART ID 0CEB DB ADDR 5067 DB CNT 0013

```

F70-09

PAGE 1 29OCT70
// JOB 29OCT70
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0000
V2 M08 ACTUAL 8K CONFIG 8K
// FOR FUNCTION TWEAK SIMPLIFIED LINEAR FUNCTION FOR CHECKOUT
*LIST ALL
*ONE WORD INTEGERS
**TWEAK! ... SIMPLIFIED LINEAR FUNCTION FOR CHECKOUT PURPOSES.

B-41

FUNCTION TWEAK(V, ANGLE)

```
TWEEK = 6.0*V/5.0
RETURN
END
```

VARIABLE ALLOCATIONS
TWEAK(R)=0000

FEATURES SUPPORTED
ONE WORD INTEGERS

CALLER SUBPROGRAMS

FMPY	FDIV	FLD	FSTO	SUBIN
------	------	-----	------	-------

REAL CONSTANTS

•600000E 01=0002 •500000E 01=0004

CORE REQUIREMENTS FOR TWEAK

COMMON	0	VARIABLES	2	PROGRAM	22
--------	---	-----------	---	---------	----

RELATIVE ENTRY POINT ADDRESS IS 0006 (HEX)

END OF COMPILATION

// DUP

*DELETE	CART ID	CEB	DB ADDR	460D	TWEEK	DB CNT	9003

*STORE	WS	UA	TWEEK	
CART ID	0CEB	DB ADDR	507A	DB CNT 0003

PAGE 1 29OCT70 29OCT70
// JOB
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0C00 0CEB 0000
V2 M08 ACTUAL 8K CONFIG 8K
// FOR FUNCTION ALBDO LBDO ANALYSIS SYSTEM
** FUNCTION 'ALBDO' ... CONSTANT ALBDO FACTOR VERSION.
#ONE WORD INTEGERS
#LIST ALL

```

PAGE 2 29OCT70 FUNCTION 'ALBDO' ... CONSTANT ALBDO FACTOR VERSION.

FUNCTION ALBDO(THETA, PHI)
COMMON IDUMY(234), CONST(8)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C THIS ROUTINE RETURNS A CONSTANT ALBDO FACTOR, INDEPENDENT OF C
C THE LOCATION OF THE DIFFERENTIAL RADIATING ELEMENT 'DA'. THE C
C VALUE OF THE ALBDO FACTOR IS ENTERED VIA THE CONSTANTS VECTOR C
C 'CONST' AND ONCE ENTERED, WILL REMAIN IN EFFECT (UNTIL MANUALLY C
C MODIFIED) SINCE IT WILL BE SAVED BY THE 'STATE' VECTOR. C
C CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C ALBDO = CONST(8)
C RETURN
C END

VARIABLE ALLOCATIONS
IDUMY(IC)=7FFF-7F16 CONST(RC)=7F14-7F06 ALBDO(R )=0000

FEATURES SUPPORTED
ONE WORD INTEGERS

CALLED SUBPROGRAMS
FLD FLDX FSTO SUBIN

CORE REQUIREMENTS FOR ALBDO
COMMON 250 VARIABLES 4 PROGRAM 16

RELATIVE ENTRY POINT ADDRESS IS 0004 (HEX)

END OF COMPILATION

// DUP

*DELETE ALBDO
CART ID 0CEB DB ADDR 48C0 DB CNT 0002

*STORE WS UA ALBDO
CART ID 0CEB DB ADDR 507D DB CNT 0002

```

F70-09

PAGE 1 29OCT70
// JOB 29OCT70
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 OCEB 0000
V2 M08 ACTUAL 8K CONFIG 8K
// FOR SUBROUTINE ROT1
*LIST ALL
*ONE WORD INTEGERS
*#1-AXIS COORDINATE TRANSFORMATION SUBROUTINE ROT1

B-45

'ROTI'

SUBROUTINE ROT1(VECTR, ANGLE)

[illegible]

```

DIMENSION VECTR(3), A(3,3), DUMMY(3)
DATA A / 1.0, 8*0.0 /

```

C = COS(ANGLE)
S = SIN(ANGLE)

C DEFINE TRANSFORMATION MATRIX ELEMENTS

$$\begin{aligned} A(2,2) &= C \\ A(2,3) &= S \\ A(3,2) &= -S \\ A(3,3) &= C \end{aligned}$$

C PERFORM MATRIX MULTIPLICATION

```

DO 100 J=1,3
  DUMMY(J) = 0.0
DO 100 K=1,3
  DUMMY(J) = DUMMY(J) + A(J,K)*VECTR(K)
CONTINUE
100

```

```
C      SET 'VECTR' = 'DUMMY'
```

```
DO 110 J=1,3
  VECTR(J) = DUMMY(J)
  CONTINUE
110
```

```

VARIABLE ALLOCATIONS
  A(R )=0010-0000 DUMMY(R )=0016-0012      C(R )=0018      S(R )=000

```

STATEMENT ALLOCATIONS
100 =0082 110 =00A1

FEATURES SUPPORTED
ONE WORD INTEGERS

CALLED SUBPROGRAMS						
FCOS	FSIN	FADDX	FMPY%	FLD	FLDX	FSTO
						FSTOX
						SUBSC

REAL CONSTANTS
•000000E 00-5322

INTEGER CONSTANTS 3=0025
1=0024

PAGE 3 29OCT70 1-AXIS COORDINATE TRANSFORMATION SUBROUTINE 'ROT1'

CORE REQUIREMENTS FOR ROT1
COMMON 0 VARIABLES 34 PROGRAM 138

RELATIVE ENTRY POINT ADDRESS IS 0026 (HEX)

END OF COMPILATION

// DUP

*DELETE ROT1
CART ID OCEB DB ADDR 460D DB CNT 000C

*STORE WS UA ROT1
CART ID OCEB DB ADDR 506F DB CNT 000C

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR SUBROUTINE ROT2

*LIST ALL

*ONE WORD INTEGERS

**2-AXIS COORDINATE TRANSFORMATION SUBROUTINE

'ROT2'

```

PAGE 2    290CT70   2-AXIS COORDINATE TRANSFORMATION SUBROUTINE      'ROT2'
SUBROUTINE ROT2(VECTR, ANGLE)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C THIS SUBROUTINE PERFORMS AN ORTHONORMAL TRANSFORMATION OF THE C
C VECTOR DENOTED BY 'VECTR' ABOUT THE 2-AXIS THROUGH AN ANGLE C
C DENOTED BY 'ANGLE'. C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C DIMENSION VECTR(3), A(3,3), DUMMY(3)
DATA A / 4*0.0, 1.0, 4*0.0 /
C C = COS(ANGLE)
C S = SIN(ANGLE)
C DEFINE TRANSFORMATION MATRIX ELEMENTS
A(1,1) = C
A(1,3) = -S
A(3,1) = S
A(3,3) = C
C PERFORM MATRIX MULTIPLICATION
DO 100 J=1,3
DUMMY(J) = 0.0
DO 100 K=1,3
DUMMY(J) = DUMMY(J) + A(J,K)*VECTR(K)
CONTINUE
100 C
C SET 'VECTR' = 'DUMMY'
DO 110 J=1,3
VECTR(J) = DUMMY(J)
CONTINUE
RETURN
END
VARIABLE ALLOCATIONS
AIR )=0010-0000 DUMMY(R )=0016-0012 C(R )=0018 SIR )=001A J(I )=001C K(I )=001D
STATEMENT ALLOCATIONS
100 =0082 110 =00A1
FEATURES SUPPORTED
ONE WORD INTEGERS
CALLED SUBPROGRAMS
FCOS FFSN FADDX FMPYX FLD FLDX FSTO FSTOX SUBSC SNR SUBIN
REAL CONSTANTS
•000000•00=0022

```

'ROT2'

PAGE 3 29OCT70 2-AXIS COORDINATE TRANSFORMATION SUBROUTINE

INTEGER CONSTANTS

1=0024 3=0025

CORE REQUIREMENTS FOR ROT2
COMMON 0 VARIABLES 34 PROGRAM 138

RELATIVE ENTRY POINT ADDRESS IS 0026 (HEX)

END OF COMPILATION

// DUP

*DELETE ROT2
CART ID OCEB DB ADDR 460D DB CNT 000C*STORE WS UA ROT2
CART ID OCEB DB ADDR 506B DB CNT 000C

PAGE 1 290CT70

// JOB

290CT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0CEB 0000

V2 M08 ACTUAL BK CONFIG BK

// FOR SUBROUTINE ROT3

*LIST ALL

*ONE WORD INTEGERS

**3-AXIS COORDINATE TRANSFORMATION SUBROUTINE

'ROT3'

F70-09

SUBROUTINE ROT3(VECTR, ANGLE)

```

C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THIS SUBROUTINE PERFORMS AN ORTHONORMAL TRANSFORMATION OF THE
C VECTOR DENOTED BY 'VECTR' ABOUT THE 3-AXIS THROUGH AN ANGLE
C DENOTED BY 'ANGLE'.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C

```

```

DIMENSION VECTR(3), A(3,3), DUMMY(3)
DATA A / 8*0.0, 1.0 /

```

```

C = COS(ANGLE)
S = SIN(ANGLE)

```

```

C DEFINE TRANSFORMATION MATRIX ELEMENTS
C

```

```

A(1,1) = C
A(1,2) = S
A(2,1) = -S
A(2,2) = C

```

```

C PERFORM MATRIX MULTIPLICATION
C

```

```

DO 100 J=1,3
  DUMMY(J) = 0.0
DO 100 K=1,3
  DUMMY(J) = DUMMY(J) + A(J,K)*VECTR(K)
CONTINUE

```

```

C SET 'VECTR' = 'DUMMY'
C

```

```

DO 110 J=1,3
  VECTR(J) = DUMMY(J)
CONTINUE
RETURN
END

```

VARIABLE ALLOCATIONS

```

AIR )=0010-0000 DUMMY(R )=0016-0012 C(R )=0018 S(R )=001A J(I )=001C K(I )=001D

```

STATEMENT ALLOCATIONS

```

100 =0082 110 =00A1

```

```

FEATURES SUPPORTED
ONE WORD INTEGERS

```

CALLED SUBPROGRAMS

```

FCOS FSIN FADDX FMPYX FLD FLDX FSTO FSTOX SUBSC SNR SUBIN

```

```

REAL CONSTANTS
.000000E 00=0022

```

```

INTEGER CONSTANTS
1=0024 3=0025

```

PAGE 3 29OCT70 3-AXIS COORDINATE TRANSFORMATION SUBROUTINE 'ROT3'

CORE REQUIREMENTS FOR ROT3
COMMON 0 VARIABLES 34 PROGRAM 138

RELATIVE ENTRY POINT ADDRESS IS 0026 (HEX)

END OF COMPILATION

// DUP

*DELETE ROT3
CART ID OCEB DB ADDR 460D DB CNT 000C

*STORE WS UA ROT3
CART ID OCEB DB ADDR 5077 DB CNT 000C

PAGE 1 29OCT70
 // JOB 29OCT70
 LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
 0000 0CEB 0CEB 0000
 V2 M08 ACTUAL 8K CONFIG 8K
 // FOR SUBROUTINE XNORM LBDO ANALYSIS PACKAGE
 **SUBROUTINE 'XNORM' VECTOR NORMALIZER.
 *LIST ALL
 *ONE WORD INTEGERS

VECTOR NORMALIZER.

SUBROUTINE XNORM(VECTR, LENGTH)

[illegible]

REAL LNTH
DIMENSION VECTR(3)

```

      LENGTH = 0.0
      DO 100 J=1,3
        LENGTH = LENGTH + VECTR(J)*VECTR(J)
      CONTINUE
      LENGTH = SQRT(LENGTH)
      DO 110 J=1,3
        VECTR(J) = VECTR(J)/LENGTH
      CONTINUE
    END SUBROUTINE
  END PROGRAM

```

**CONTINUE
RETURN**

END
VARIABLE ALLOCATIONS
J(I)=0000

STATEMENT ALLOCATIONS
100 #002A 110 =0047

FEATURES SUPPORTED
ONE WORD INTEGERS

CALLLED SUBPROGRAMS

```
FSQRT  FADD  FM
REAL CONSTANTS
.0000000E 00=00002
```

INTEGER CONSTANTS 3=0005
 1=0004

CORE REQUIREMENTS FOR XNORM
COMMON 0 VARIABLES 2 PROGRAM 84

RELATIVE ENTRY POINT ADDRESS IS 0006 (HEX)

END OF COMPILATION

11/09/09

#DELETE	CART ID	OCB	DB ADDR	460D	DB CNT	0007
			XNORM			

F70-09

PAGE 3 29OCT70
*STORE WS UA XNORM
CART ID OCEB DB ADDR 5073 DB CNT 0007

B-56

F70-09

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR FUNCTION INNER

**FUNCTION SUBPROGRAM 'INNER'

*ONE WORD INTEGERS

*LIST ALL

'LBDO' ANALYSIS PACKAGE

```

REAL FUNCTION INNER(A, B)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THIS FUNCTION FORMS THE INNER PRODUCT OF THE VECTORS A AND B. C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C DIMENSION A(3), B(3)
C
C INNER = 0.0
C DO 100 J=1,3
C   INNER = INNER + A(J)*B(J)
C 100 CONTINUE
C   RETURN
C   END

```

```

VARIABLE ALLOCATIONS
INNER(R )=0000 J(I )=0002

```

```

STATEMENT ALLOCATIONS
100 =0024

```

FEATURES SUPPORTED
ONE WORD INTEGERS

```

CALLED SUBPROGRAMS
FADD FMPYX FLD FLDX FSTO SUBSC SUBIN

```

```

REAL CONSTANTS
.000000E 00=0004

```

```

INTEGER CONSTANTS
1=0006 3=0007

```

```

CORE REQUIREMENTS FOR INNER
COMMON 0 VARIABLES 4 PROGRAM 46

```

RELATIVE ENTRY POINT ADDRESS IS 0008 (HEX)

END OF COMPILATION

// DUP

```

*DELETE
CART ID 0CEB DB ADDR 460D DB CNT 0005
C
*STORE
CART ID 0CEB DB ADDR 507A DB CNT 0004

```

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR SUBROUTINE MESAG LBDO ANALYSIS PACKAGE
**SUBROUTINE 'MESAG' 'LBDO' ANALYSIS PACKAGE
*ONE WORD INTEGERS
*LIST ALL

**E
L
E
L
E
E
E
E
E
E
E
E
E
E
E
E**

*DELETE		MESAG		
CART ID	OCEB	DB ADDR	460D	DB CNT 0008
*STORE	WS	UA MESAG		
CART ID	OCEB	DB ADDR	506E	DB CNT 0008

F70-09

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR SUBROUTINE LBDOZ

*LIST ALL

*ONE WORD INTEGERS

** ROUTING WORD INPUT AND ANALYSIS SUBROUTINE FOR 'LBDO'

B-62

SUBROUTINE LBDOZ(NUMBR)

THIS SUBROUTINE READS AN EIGHT-CHARACTER ROUTING WORD FROM
EITHER THE CONSOLE KEYBOARD OR THE CARD READER AND COMPARES IT
TO A LIST CONTAINING THE LITERALS 'KEYBOARD', 'CARD', AND 'QUIT'.
IF A MATCH IS FOUND, THE VARIABLE 'NUMBER' IS SET EQUAL TO 1, 2,
OR 3 RESPECTIVELY. IF NO MATCH IS FOUND, THE ROUTING WORD IS
RE-ENTERED OR 'NUMBER' IS SET EQUAL TO 4, DEPENDING ON WHICH
INPUT DEVICE IS IN CONTROL.

CHARLES E. ROSE
BALL BROTHERS RESEARCH CORPORATION
BOULDER, COLORADO 80302
28 AUGUST 1970

CHARLES E. ROSE
BALL BROTHERS RESEARCH CORPORATION
BOULDER, COLORADO 80302
28 AUGUST 1970

```

INTEGER MODE(4), WORD(4,3), DEVIC(2)
DATA WORD, DEVIC /KE, YB, OA, RD,
1      CA, RD, I, I, I,
2      QU, IT, I, I, I,
3      6, 2 /

```

COMMON NMODE

```

100 GO TO(100,110), NMODE
110 WRITE(1,1000)
      IO = DEVIC(NMODE)
      READ(IO, 1010) (MODE(J), J=1,4)
      DO 130 NUMBER = 1,3
      DO 120 J=1,4
      IF(MODE(J) - WORD(J,NUMBER)) 130,120,130

```

```

120 CONTINUE
    RETURN
130 CONTINUE
    GO TO(140,150), NMODE
140 WRITE(1,1020)
    GO TO 110
150 NUMBR = 4
    RETURN

```

```
1000 FORMAT('ENTER ROUTING WORD.')
1010 FORMAT(4A2)
1020 FORMAT('UNDEFINED WORD. ENTER
1')
```

END

VARIABLE ALLOCATIONS

```

MODE(I) = 0003-0000      WORD(I) = 000F-0004  DEVIC(I) = 0011-0010
NMODE(IC)=7FFF           IC(I) = 0012

```

STATEMENT ALLOCATIONS

INTERMENT ALLOCATIONS

1000 =0019	1010 =0025	1020 =0028	100 =0053	110 =0057	120 =0094	130 =009F	140 =00AE	150 =0084
------------	------------	------------	-----------	-----------	-----------	-----------	-----------	-----------

PAGE 3 29OCT70 ROUTING WORD INPUT AND ANALYSIS SUBROUTINE FOR 'LBDO'

FEATURES SUPPORTED
ONE WORD INTEGERS

CALLED SUBPROGRAMS
SRED SWRT SCOMP SIOIX SUBSC SUBIN

INTEGER CONSTANTS
1=0016 4=0017 3=0018

CORE REQUIREMENTS FOR LBDOZ
COMMON 2 VARIABLES 22 PROGRAM 164

RELATIVE ENTRY POINT ADDRESS IS 0044 (HEX)

END OF COMPILATION

// DUP

*DELETE LBDOZ
CART ID OCEB DB ADDR 4880 DB CNT 000E

*STORE WS UA LBDOZ
CART ID OCEB DB ADDR 5066 DB CNT 000E

F70-09

PAGE 1 29OCT70 29OCT70
// JOB
LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0000
V2 M08 ACTUAL 8K CONFIG 8K
// FOR SUBROUTINE CRDMP ... ALBEDO ANALYSIS SYSTEM
*LIST ALL
*ONE WORD INTEGERS
** SUBROUTINE 'CRDMP' ... ALBEDO ANALYSIS SYSTEM.

B-65

INTEGER CONSTANTS 5=0003 7=0004
6=0002

PAGE 3 29OCT70 SUBROUTINE 'CRDMP' ... ALBEDO ANALYSIS SYSTEM.

CORE REQUIREMENTS FOR CRDMP
COMMON 136 VARIABLES 2 PROGRAM 54

RELATIVE ENTRY POINT ADDRESS IS 0005 (HEX)

END OF COMPILATION

// DUP

*DELETE				CRDMP			
CART ID	OCEB	DB	ADDR	4AF0	DB	CNT	0004
*STORE		WS	UA	CRDMP			
CART ID	OCEB	DB	ADDR	5074	DB	CNT	0004

F70-09

PAGE 1 29OCT70

// JOB

29OCT70

LOG DRIVE CART SPEC CART AVAIL PHY DRIVE
0000 0CEB 0CEB 0000

V2 M08 ACTUAL 8K CONFIG 8K

// FOR SUBROUTINE GRAF ... AXIS DRAWER AND SCALER FOR LBDOD
*LIST ALL
*ONE WORD INTEGERS
** SUBROUTINE 'GRAF', AXIS DRAWER AND SCALER FOR PROGRAM 'LBDOD'

B-68

SUBROUTINE GRAF(MAX, TITLE)

INTEGER TITLE(40)

DO 150 -1,5

150 CONTINUE

DO 160 J=1,4

150 CONTINUE

END
VARIABLE ALLOCATIONS

N(I) = 0007

FEATURES SUPPORTED
ONE WORD INTEGERS

REAL CONSTANTS

•100000E 01=000A
•100000E 02=0016
•500000E 01=0022
•200000E -01=0024
•500000E -01=0018
•000000E 00=000C

FSTO	FL0AT	SWRT	SCOMP	SIOIX	SIOF	SIOI
•250000E 00=0010		•200000E 02=0012		•450000E 02=0014		
•314159E 01=001C		•200000E 01=001E		•980000E 00=0020		

PAGE 3 29OCT70 SUBROUTINE 'GRAF'. AXIS DRAWER AND SCALER FOR PROGRAM 'LBDOD'
 INTEGER CONSTANTS
 1=0026 5=0027 10=0028 2=0029 7=002A 40=002B 3=002C 0=002D 9=002E 4=002F
 15=0030

CORE REQUIREMENTS FOR GRAF
 COMMON 0 VARIABLES 10 PROGRAM 276
 RELATIVE ENTRY POINT ADDRESS IS 0038 (HEX)

END OF COMPILATION

// DUP

*DELETE GRAF
 CART ID OCEB DB ADDR 4AFF DB CNT 0014
 *STORE WS UA GRAF
 CART ID OCEB DB ADDR 5058 DB CNT 0014

// *

// *

// * THUS, THE LOADING OF THE ALBEDO ANALYSIS SYSTEM CONCLUDETH.



F70-09

Appendix C
TYPICAL OUTPUT SHEETS

ATM ATMOSPHERIC REFRACTION AND ATTENUATION STUDY.

INTEGRATION INTERVAL = 0.10 KILOMETERS
INDEX-OF-REFRACTION SUBSCRIPT = 3
PRINT EVERY 106-TH CALCULATION

H KILOMETERS	THEIA DEGREES	D(THETA) DEGREES	R(1) KILOMETERS	R(2) KM/RADIAN	ZENITH DEGREES	PATH LENGTH KILOMETERS	REFRACT. ANGLE ARC-MINUTES
2.000	0.000	0.895-03	6374.000	0.000	90.000	0.00	
3.006	0.095	0.895-03	6374.003	8.634	89.922	10.60	1.1
3.028	0.190	0.895-03	6374.023	17.274	89.844	21.20	2.1
2.064	0.285	0.895-03	6374.064	25.820	89.767	31.80	3.1
3.114	0.381	0.895-03	6374.114	34.577	89.689	42.40	4.2
3.178	0.476	0.895-03	6374.178	43.247	89.611	53.00	5.2
3.257	0.571	0.895-03	6374.257	51.934	89.533	63.60	6.2
3.350	0.666	0.895-03	6374.350	60.644	89.454	74.20	7.3
3.458	0.762	0.895-03	6374.458	69.377	89.376	84.90	8.3
3.581	0.857	0.895-03	6374.581	78.137	89.297	95.40	9.3
3.718	0.952	0.895-03	6374.718	86.928	89.218	106.00	10.2
3.869	1.044	0.895-03	6374.869	95.752	89.139	116.50	11.2
4.036	1.143	0.895-03	6375.036	104.612	89.059	127.20	12.1
4.217	1.235	0.895-03	6375.217	113.510	88.979	137.80	13.1
4.413	1.333	0.895-03	6375.413	122.448	88.899	148.40	14.0
4.624	1.429	0.895-03	6375.624	131.429	88.819	159.00	14.9
4.849	1.524	0.895-03	6375.849	140.456	88.738	169.60	15.7
5.090	1.619	0.895-03	6376.090	149.528	88.656	180.20	16.5
5.346	1.714	0.895-03	6376.346	158.648	88.574	190.80	17.3
5.617	1.809	0.895-03	6376.617	167.818	88.492	201.40	18.1
5.903	1.905	0.895-03	6376.903	177.038	88.409	212.00	18.8
6.205	2.000	0.895-03	6377.205	186.309	88.326	222.60	19.6
6.522	2.095	0.895-03	6377.522	195.634	88.242	233.20	20.3
6.855	2.190	0.895-03	6377.855	205.011	88.158	243.80	20.9
7.203	2.285	0.895-03	6378.203	214.441	88.074	254.40	21.6
7.567	2.381	0.895-03	6378.567	223.924	87.989	265.00	22.2
7.946	2.476	0.895-03	6378.946	233.461	87.903	275.50	22.8
8.342	2.571	0.895-03	6379.342	243.051	87.818	286.20	23.3
8.753	2.666	0.895-03	6379.753	252.693	87.731	296.80	23.8
9.180	2.761	0.895-03	6380.180	262.387	87.645	307.40	24.3
9.624	2.856	0.895-03	6380.624	272.133	87.557	318.00	24.8
10.083	2.951	0.895-03	6381.083	281.930	87.470	328.60	25.3
10.559	3.046	0.895-03	6381.559	291.775	87.382	339.20	25.7
11.051	3.141	0.895-03	6382.051	301.670	87.293	349.80	26.1
11.560	3.235	0.895-03	6382.560	311.611	87.204	360.40	26.5
12.085	3.331	0.895-03	6383.085	321.599	87.115	371.00	26.8
12.626	3.427	0.895-03	6383.626	331.631	87.026	381.50	27.1
13.184	3.522	0.895-03	6384.184	341.706	86.936	392.20	27.4
13.759	3.617	0.895-03	6384.759	351.823	86.845	402.80	27.7
14.350	3.711	0.895-03	6385.350	361.980	86.755	413.40	28.0
14.958	3.806	0.895-03	6385.958	372.175	86.664	424.00	28.2
15.583	3.901	0.895-03	6386.583	382.407	86.573	434.60	28.5

15.225	3.996	0.89F-03	5387.725	392.675	86.481	445.20	28.7
16.384	4.051	0.89F-03	6387.884	402.976	86.390	455.80	28.9
17.559	4.186	0.89F-03	6388.559	413.310	86.298	466.40	29.1
18.252	4.281	0.89F-03	6389.252	423.673	86.206	477.00	29.2
19.962	4.375	0.89F-03	6389.962	434.066	86.113	487.60	29.4
19.638	4.471	0.89F-03	6390.638	444.456	86.021	498.20	29.5
20.432	4.565	0.89F-03	6391.432	454.932	85.928	508.80	29.6
21.193	4.660	0.89F-03	6392.193	465.403	85.835	519.40	29.7
21.971	4.755	0.89F-03	6392.971	475.897	85.742	530.00	29.8
22.767	4.850	0.89F-03	6393.767	486.413	85.649	540.60	29.9
23.579	4.944	0.89F-03	6394.579	496.949	85.556	551.20	30.0
24.409	5.039	0.89F-03	6395.409	507.505	85.462	561.80	30.1
25.254	5.134	0.89F-03	6396.256	518.079	85.369	572.40	30.2
26.170	5.228	0.89F-03	6397.120	528.670	85.275	583.00	30.3
27.001	5.323	0.89F-03	6398.001	539.278	85.182	593.50	30.3
27.900	5.417	0.89F-03	6398.900	549.900	85.088	604.20	30.3
28.816	5.512	0.89F-03	6399.816	560.537	84.994	614.80	30.4
29.749	5.607	0.89F-03	6400.749	571.188	84.900	625.40	30.4
30.700	5.701	0.89F-03	6401.700	581.851	84.806	636.00	30.4
31.668	5.795	0.89F-03	6402.668	592.527	84.712	646.60	30.5
32.653	5.890	0.89F-03	6403.653	603.214	84.618	657.20	30.5
33.656	5.984	0.89F-03	6404.656	613.912	84.524	667.80	30.5
34.676	6.079	0.89F-03	6405.676	624.620	84.430	678.40	30.5
35.713	6.173	0.89F-03	6406.713	635.338	84.336	689.00	30.6
36.767	6.267	0.89F-03	6407.767	646.066	84.242	699.60	30.6
37.839	6.362	0.89F-03	6408.839	656.803	84.148	710.20	30.6
38.928	6.456	0.89F-03	6409.928	667.548	84.054	720.80	30.6
40.035	6.550	0.89F-03	6411.035	678.302	83.960	731.40	30.6
41.158	6.644	0.89F-03	6412.158	689.064	83.866	742.00	30.6
42.300	6.738	0.89F-03	6413.300	699.834	83.772	752.60	30.6
43.459	6.833	0.89F-03	6414.458	710.612	83.678	763.20	30.6
44.633	6.927	0.89F-03	6415.633	721.398	83.584	773.80	30.6
45.826	7.021	0.89F-03	6416.826	732.191	83.490	784.40	30.7
47.036	7.115	0.89F-03	6418.036	742.991	83.396	795.00	30.7
48.264	7.209	0.89F-03	6419.264	753.799	83.302	805.60	30.7
49.509	7.303	0.89F-03	6420.509	764.613	83.208	816.20	30.7
50.758	7.396	0.89F-03	6421.758	775.333	83.115	826.70	30.7

F70-09

ATMOSPHERIC REFRACTION AND ATTENUATION STUDY.

WAVELENGTH = 0.80 MICRONS
 INTEGRATION INTERVAL = 0.10 KILOMETERS
 INDEX-OF-REFRACTION FUNCTION SUBSCRIPT = 3
 PRINT EVERY 106-TH CALCULATION

H	SUV	TRANSMISSION	PATH LENGTH
3.000	0.00000E 00	0.88040E 00	10.50
3.006	0.12691E 00	0.77554E 00	21.10
3.027	0.25414E 00	0.68405E 00	31.70
3.063	0.37971E 00	0.60474E 00	42.30
3.113	0.50261E 00	0.53686E 00	52.90
3.177	0.62200E 00	0.47854E 00	63.50
3.253	0.73599E 00	0.42829E 00	74.10
3.349	0.84570E 00	0.38664E 00	84.70
3.456	0.95023E 00	0.35194E 00	95.30
3.578	0.10467E 01	0.32135E 00	105.90
3.715	0.11352E 01	0.29675E 00	116.50
3.866	0.12143E 01	0.27669E 00	127.10
4.033	0.12844E 01	0.25973E 00	137.70
4.213	0.13480E 01	0.24515E 00	148.30
4.409	0.14058E 01	0.23274E 00	158.90
4.620	0.14578E 01	0.22235E 00	169.50
4.845	0.15034E 01	0.21381E 00	180.10
5.085	0.15426E 01	0.20655E 00	190.70
5.341	0.15771E 01	0.20034E 00	201.30
5.612	0.16070E 01	0.19522E 00	211.90
5.898	0.16334E 01	0.19037E 00	222.50
6.199	0.16555E 01	0.18584E 00	233.10
6.514	0.16732E 01	0.18154E 00	243.70
6.844	0.16863E 01	0.17750E 00	254.30
7.186	0.16934E 01	0.17375E 00	264.90
7.550	0.16935E 01	0.17034E 00	275.50
7.939	0.16859E 01	0.16724E 00	286.10
8.334	0.16700E 01	0.16440E 00	296.70
8.745	0.16467E 01	0.16189E 00	307.30
9.172	0.16157E 01	0.15955E 00	317.90
9.615	0.15782E 01	0.15746E 00	328.50
10.074	0.15354E 01	0.15562E 00	339.10
10.550	0.14882E 01	0.15405E 00	349.70
11.042	0.14369E 01	0.15274E 00	360.30
11.550	0.13819E 01	0.15161E 00	370.90
12.075	0.13244E 01	0.15062E 00	381.50
12.616	0.12647E 01	0.14975E 00	392.10
13.173	0.12030E 01	0.14898E 00	402.70
13.744	0.11405E 01	0.14834E 00	413.30
14.339	0.10774E 01	0.14780E 00	423.90
14.947	0.10139E 01	0.14735E 00	434.50
15.571	0.09501E 01	0.14699E 00	

F70-09

16.213
16.871
17.546
18.239
18.948
19.675
20.418
21.179
21.956
22.751
23.564
24.393
25.240
26.103
26.985
27.883
28.799
29.732
30.682
31.650
32.635
33.637
34.656
35.693
36.747
37.819
38.908
40.014
41.137
42.278
43.436
44.611
45.804
47.013
48.241
49.485
50.746
50.758

0.18713E 01
0.18762E 01
0.18811E 01
0.18856E 01
0.18901E 01
0.18944E 01
0.18987E 01
0.19029E 01
0.19070E 01
0.19109E 01
0.19147E 01
0.19181E 01
0.19212E 01
0.19240E 01
0.19265E 01
0.19297E 01
0.19306E 01
0.19322E 01
0.19337E 01
0.19348E 01
0.19358E 01
0.19367E 01
0.19374E 01
0.19380E 01
0.19386E 01
0.19390E 01
0.19394E 01
0.19397E 01
0.19399E 01
0.19401E 01
0.19402E 01
0.19404E 01
0.19405E 01
0.19406E 01
0.19407E 01
0.19407E 01
0.19407E 01
0.19407E 01

0.15391E 00
0.15314E 00
0.15242E 00
0.15172E 00
0.15105E 00
0.15039E 00
0.14975E 00
0.14913E 00
0.14852E 00
0.14793E 00
0.14737E 00
0.14687E 00
0.14641E 00
0.14601E 00
0.14565E 00
0.14533E 00
0.14505E 00
0.14481E 00
0.14460E 00
0.14444E 00
0.14429E 00
0.14417E 00
0.14406E 00
0.14397E 00
0.14390E 00
0.14384E 00
0.14378E 00
0.14374E 00
0.14370E 00
0.14367E 00
0.14365E 00
0.14363E 00
0.14362E 00
0.14360E 00
0.14360E 00
0.14359E 00
0.14359E 00
0.14359E 00
0.14359E 00

445.10
455.70
466.30
476.90
487.50
498.10
509.70
519.30
529.90
540.50
551.10
561.70
572.30
582.90
593.50
604.10
614.70
625.30
635.90
646.50
657.10
667.70
678.30
688.90
699.50
710.10
720.70
731.30
741.90
752.50
763.10
773.70
784.30
794.90
805.50
816.10
826.70
826.80

C-7/C-8

F70-09

AFT TRANSMISSION ALONG PATH = 14.35 PERCENT.

T² = 2.06%



F70-09

Appendix D
TWO-DIMENSIONAL INTERPOLATION



F70-09

In both the "LBDO" and atmospheric-attenuation programs, there arose the problem of interpolating over a two-dimensional field of tabulated data. This appendix develops one technique for doing this.

Assume we have data tabulated over a uniform two-dimensional rectangular grid pattern. That is, we have sets of points

$$(x_i, y_i, z_{ij}) \quad i, j = 1, 2 \quad (D1)$$

What is $z(x, y)$ where $x_1 < x < x_2$, $y_1 < y < y_2$?

There are many ways of performing such an interpolation; the one chosen for this application is to fit the highest-order polynomial of two independent variable to the four points. Since four pieces of information are available, the interpolating polynomial can contain at most four unknown constants.

First, we make the transformation

$$\begin{aligned} u &= x - x_1 \\ v &= y - y_1 \\ \Delta x &= x_2 - x_1 \\ \Delta y &= y_2 - y_1 \end{aligned} \quad (D2)$$

then

$$z = z(u, v, \Delta x, \Delta y) \quad (D3)$$



F70-09

and has the general form

$$z = a_{10}u + a_{11}uv + a_{01}v + a_{00} \quad (D4)$$

where the a's are to be determined to force $z(x_i, y_i) = z_{ij}$.

This can be accomplished by writing

$$\begin{bmatrix} 0 & 0 & 0 & 1 \\ \Delta x & 0 & 0 & 1 \\ 0 & 0 & \Delta y & 1 \\ \Delta x & \Delta x \Delta y & \Delta y & 1 \end{bmatrix} \begin{bmatrix} a_{10} \\ a_{11} \\ a_{01} \\ a_{00} \end{bmatrix} = \begin{bmatrix} z_{11} \\ z_{21} \\ z_{12} \\ z_{22} \end{bmatrix} \quad (D5)$$

Equation (D5) is simply the matrix representation of four simultaneous (D4) equations evaluated at the four known points.

Equation (D5) is equivalent to the matrix equation

$$DA = Z \quad (D6)$$

and the coefficients are obtained by solving the equation

$$A = D^{-1}Z \quad (D7)$$

The actual inversion of D is a tedious exercise in algebra and not very interesting, so it is omitted. Once D is inverted and the $D^{-1}Z$ product performed, the {A} coefficients are obtained to give

$$z = \left(\frac{z_{21} - z_{11}}{\Delta x} \right) u + \left(\frac{z_{11} - z_{21} - z_{12} + z_{22}}{\Delta x \Delta y} \right) uv + \left(\frac{z_{12} - z_{11}}{\Delta y} \right) v + z_{11} \quad (D8)$$



F70-09

Thus Eq. (D8) is the interpolating polynomial which can be used for estimating z -values within the four known points for which the polynomial is defined.

This function (D8) has the interesting property of describing a warped, compound surface which can be constructed entirely of straight lines. One such surface is shown in Fig. D-1. This is demonstrated by the fact that both $\partial z / \partial u$ and $\partial z / \partial v$ are constant but in general, the intersection of the $z(u,v)$ surface with a plane which is parallel to the z axis produces a parabola. In fact, the surface is a hyperbolic paraboloid aligned symmetric to the line $u = v$.

The property of $\partial z / \partial u$ and $\partial z / \partial v$ being constant has some interesting consequences in the field of architecture, since curved roofs of this shape can be constructed using straight two-by-fours; indeed, such houses are to be found.



F70-09

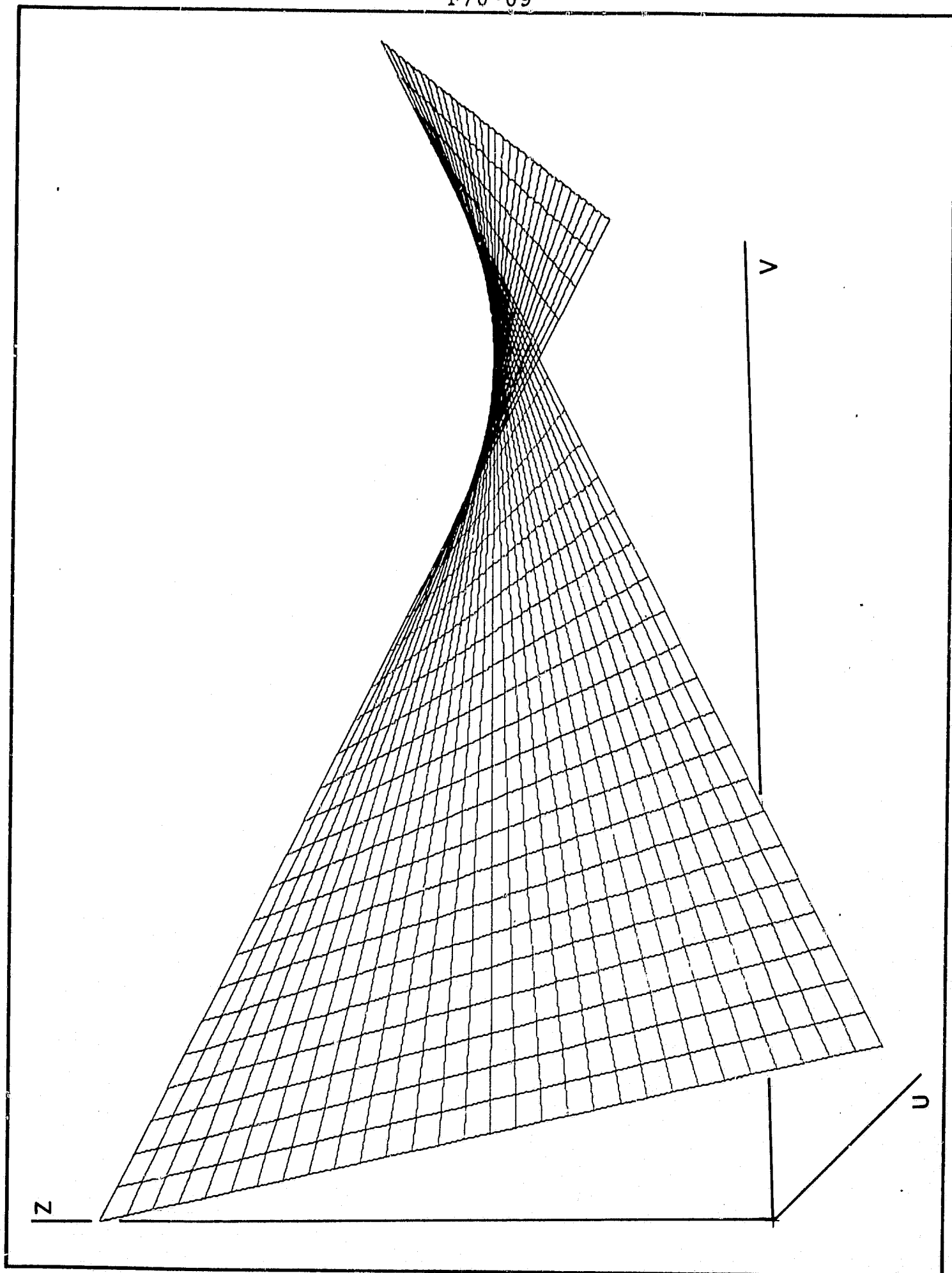


Fig. D-1 Two-Dimensional Interpolating Surface



F70-09

Appendix E

A NOTE ON LAMBERTIAN RADIATORS



F70-09

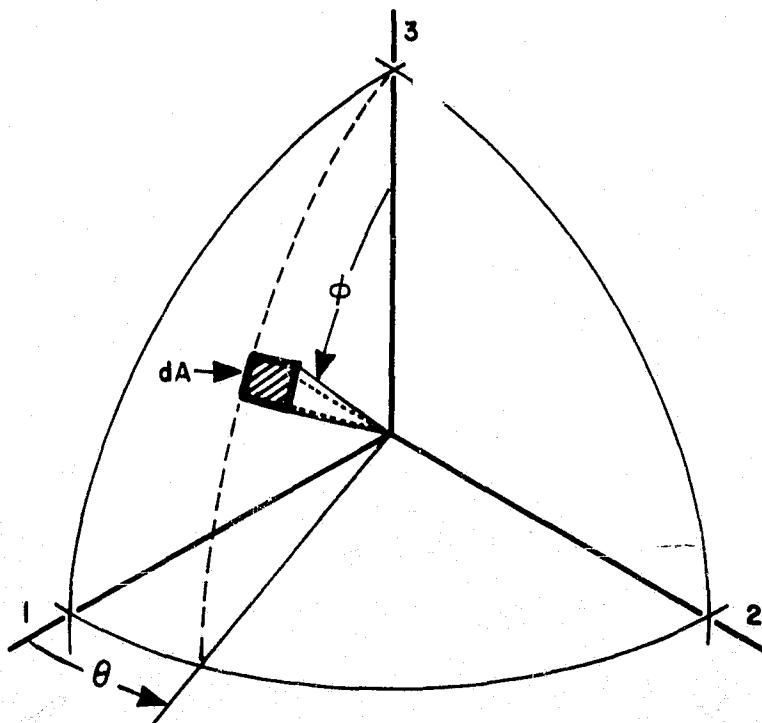
The albedo equation [ref. Eq. (4), Section 2] has included within it a factor $1/\pi$. One might suspect a certain magic in its presence, but this is not so; there is a good reason for its being there. It is a result of assuming that the atmosphere is a Lambertian radiator.

One way of demonstrating the basis for the $1/\pi$ number is to use the "correction factor" approach: What proportionality factor must we use to make the answer come out right?

Assume we have a very small area located at the origin of a reference system and perpendicular to the 3 axis, and that out of this area is radiated a power P^* . If the Lambert radiation law is assumed, then the differential power radiated into an element of area dA located on a hemisphere of radius r centered on the coordinate system is

$$dP = k \frac{P^* \cos \phi}{r^2} dA$$

where k is a correction factor (see figure below).





F70-09

Since Lambert's law states that no power is radiated below the (1,2) plane, we can compute the total energy incident upon the hemisphere as follows:

$$P = P^*k \int_0^{2\pi} \int_0^{\pi/2} \frac{\cos \phi}{r^2} \sin \phi \, d\phi \, d\theta = P^*k\pi$$

Since the power absorbed must equal the power radiated, we have

$$P^*k\pi \equiv P^*$$

so

$$k = 1/\pi$$