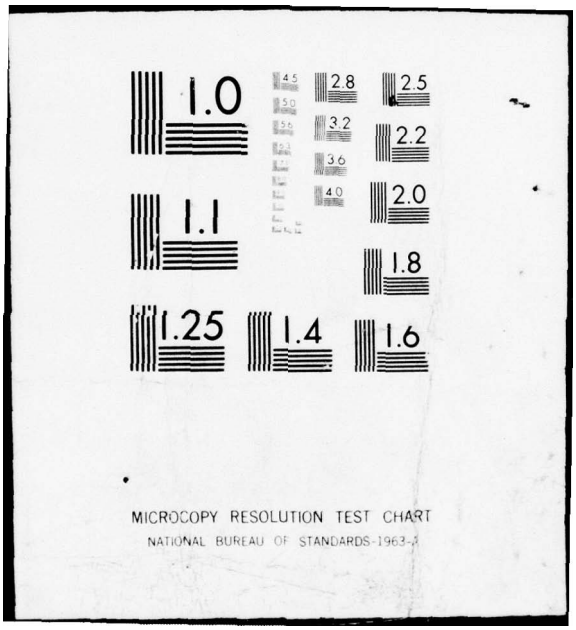
AD-A043 120 HUMAN ENGINEERING LAB ABERDEEN PROVING GROUND MD F/G 1/3
COMPUTING INTERNAL COCKPIT REFLECTIONS OF EXTERNAL POINT-LIGHT --ETC(U)
JUN 77 C SMYTH

UNCLASSIFIED HEL-TM-20-77 NL

| OF |
AD
A043120

END
DATE
FILMED
9-77
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD

Technical Memorandum 20-77

COMPUTING INTERNAL COCKPIT REFLECTIONS OF EXTERNAL POINT-LIGHT SOURCES FOR THE MODEL 209 AH-1S HELICOPTER FLAT-PLATE CANOPY DESIGN

Christopher Smyth

June 1977
AMCMS Code 624209.12.DC52

U. S. ARMY HUMAN ENGINEERING LABORATORY
Aberdeen Proving Ground, Maryland

Destroy this report when no longer needed.
Do not return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Use of trade names in this report does not constitute an official endorsement or approval of the use of such commercial products.

AMCMS Code 624209.12.DC52

Technical Memorandum 20-77

14 HEL-TM-20-77

6

COMPUTING INTERNAL COCKPIT REFLECTIONS OF EXTERNAL POINT-LIGHT SOURCES FOR THE MODEL 209 AH-1S HELICOPTER FLAT-PLATE CANOPY DESIGN

9 Final rpt.,

10 Christopher Smyth

11 June 1977

12 50 p.

APPROVED: John D. Weisz
JOHN D. WEISZ
Director
US Army Human Engineering Laboratory

US ARMY HUMAN ENGINEERING LABORATORY
Aberdeen Proving Ground, Maryland 21005

172 850

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Technical Memorandum 20-77	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* COMPUTING INTERNAL COCKPIT REFLECTIONS OF EXTERNAL POINT-LIGHT SOURCES FOR THE MODEL 209 AH-1S HELICOPTER FLAT-PLATE CANOPY DESIGN		5. TYPE OF REPORT & PERIOD COVERED Final 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Christopher Smyth		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS US Army Human Engineering Laboratory Aberdeen Proving Ground, Maryland 21005		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS AMCMS Code 624209.12.DC52
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE June 1977 13. NUMBER OF PAGES 50
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Helicopters
Light-Reflection Program
Flat-Plate Canopy
AH-1S Helicopter
Human Factors Engineering

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The US Army Human Engineering Laboratory (HEL) has developed a computer program for computing the internal cockpit reflections of external point light sources. Computations have been completed for the Model 209 AH-1S COBRA helicopter with the flat-plate canopy design. The results indicate that internal reflections are possible for a wide range of external source locations. A computer graphics output is included in the program to show the reflection points on a perspective drawing of the cockpit canopy as seen from the pilot's position. This report contains hard copies of such perspectives and describes the program and routines.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

CONTENTS

COMPUTING INTERNAL COCKPIT REFLECTIONS OF EXTERNAL POINT-LIGHT SOURCES FOR THE MODEL 209 AH-1S HELICOPTER FLAT-PLATE CANOPY DESIGN

INTRODUCTION

The US Army Human Engineering Laboratory (HEL) has developed a computer program for computing internal cockpit reflections of external point light sources. This work was accomplished at the request of the US Army Air Mobility Research and Development Laboratory for application to the flat-plate canopy design of the Model 209 AH-1S COBRA helicopter. The flat-plate canopy (FPC) design reduces the solar glint which occurs during daylight operations to a momentary flash at certain observer-aircraft-sun angles. Consequently, a moving aircraft no longer has the continual solar glint which occurred with the earlier compound-shaped canopy designs. The continuous presence of solar glint increased the range of visual detection by ground observers.

However, in certain situations during night operations the internal surfaces of the FPC act like mirrors for external sources of light as seen from the pilot's position. These virtual images of ground-level lights can cause disorientation. The pilot cannot discriminate between reflections and light sources on the ground and creates a safety hazard during flight.

The computer program (Appendix C) was part of a larger study effort by HEL to determine the full extent of the nighttime light reflection problem and review various proposals for solution (5). The conclusion reached by HEL is that the most realistic solution to the combined daytime glint/nighttime reflection problem is a modification of the FPC side panels to a simple curved shape. It is possible with little additional programming to use the inclosed program to assist in further canopy-design work.

METHOD

A three-dimensional ray-tracing program was written to trace straight-line rays backwards from the position (nominal) of the pilot's eye to visible points on the internal transparent surfaces of the canopy. Each such ray is continued backwards between internal canopy-surface points until a non-transparent surface is reached. Such surfaces are assumed to be diffusive without specular reflectance. At each reflection point, the reflectance and transmittance are computed along with the directional cosines of the corresponding incident external and reflected rays. In this way, a reflected ray is traced backwards from the pilot's eye to all possible external sources.

The method is applicable to any canopy design although application is limited to the FPC in this report. The canopy is specified by a covering net of flat surfaces. Each surface is described by the coordinates of its vertices in the three-dimensional space and the rotational order in which they are listed.

Tracing rays backwards, the program computes the intersection point of a straight-line ray given its directional cosines and origin point, with a plane containing each surface segment in turn. The reflection point for the ray is that intersection point which is contained within the edges of the corresponding segment. The angle of incidence between the surface normal and ray at this point is computed along with the corresponding reflectance and transmittance values and the directional cosines of the transmitted and reflected rays. Working backwards, the reflected ray becomes the incident ray for the next set of computations (Appendix A).

The program includes obstructing surfaces as well as the transparent surfaces of the canopy in the computations. These surfaces either obstruct the pilots' forward vision or block out incident rays from external sources. These surfaces are (a) the pilot's display panel and side armor, (b) the copilot's seat, side armor and gunner's sight, and (c) the sides and floor of the cockpit. They are specified in the same manner as are the canopy segments with however, appropriate coding added. The computation of a reflection point for a ray on an obstructing surface renders the computation complete since the backwards traced ray is considered to be absorbed.

This computation process is repeated for pilot-viewing directions which are indexed at equal increments over a quarter sector. The sector is bounded by vision directly to the front, to the side, and bottom. In this way a table is constructed which lists at discrete intervals all possible internal-reflection points and the corresponding external-light directions. This approach generates a large amount of data and a computer-graphics routine is included for output. The incident-ray entry points and the corresponding reflection points are shown on a perspective drawing of the cockpit as seen from the pilot's position for light-ray conditions of interest (see Appendixes B and C).

DISCUSSION

The results of this application to the FPC are shown in Figures 1 through 4. These figures are hard copies of the computer-graphics output. Figure 1 shows side, top, and front views of the flat-plate canopy. Internal obstructing surfaces are sketched in with broken lines. The pilot's nominal eye position is shown in each view. The aircraft is not included in the sketch.

Figures 2, 3, and 4 are perspective drawings of the cockpit as seen from the pilot's position. The pilot's nominal viewing direction is shown by the small cross in the center of the upper front canopy surface. The figure shows that the lower portions of the front and forward side canopy surfaces are blocked from view by the pilot's instrument panel.

The "dots" on Figure 3 indicate entry points of external rays which generate visible reflections on the right-hand side of the canopy. The figure shows that some entry points are on the lower front and side canopy surfaces which are outside of the pilot's field-of-view.

Figure 4 shows the corresponding reflection points which are generated on the right-hand side of the canopy. The reflection points are spaced at $2^{o} \times 2^{o}$ increments. The number shown at each reflection point is equal to the negative value of the logarithm (base 10) of the light reflectance. The numbers are truncated to the integer values by dropping the fractional part. The numerical "zero" corresponds to those reflectances which are greater than 0.1 in value. The numerical "one" corresponds to those values equal to 0.1 or less but greater than 0.01. A similar range exists for the numerical "two."

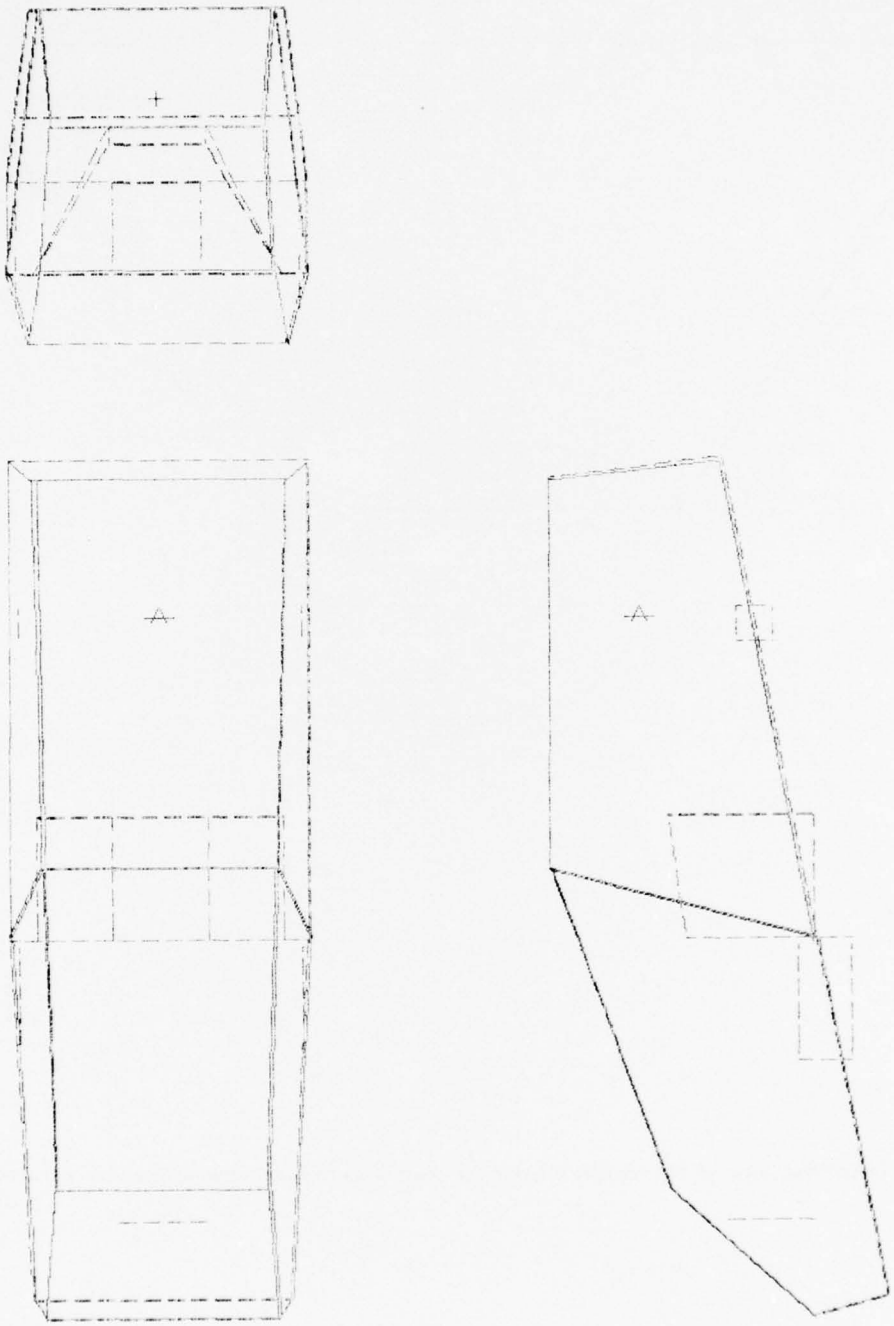


Figure 1. Top, side, and front views of the flat-plate canopy design without aircraft.

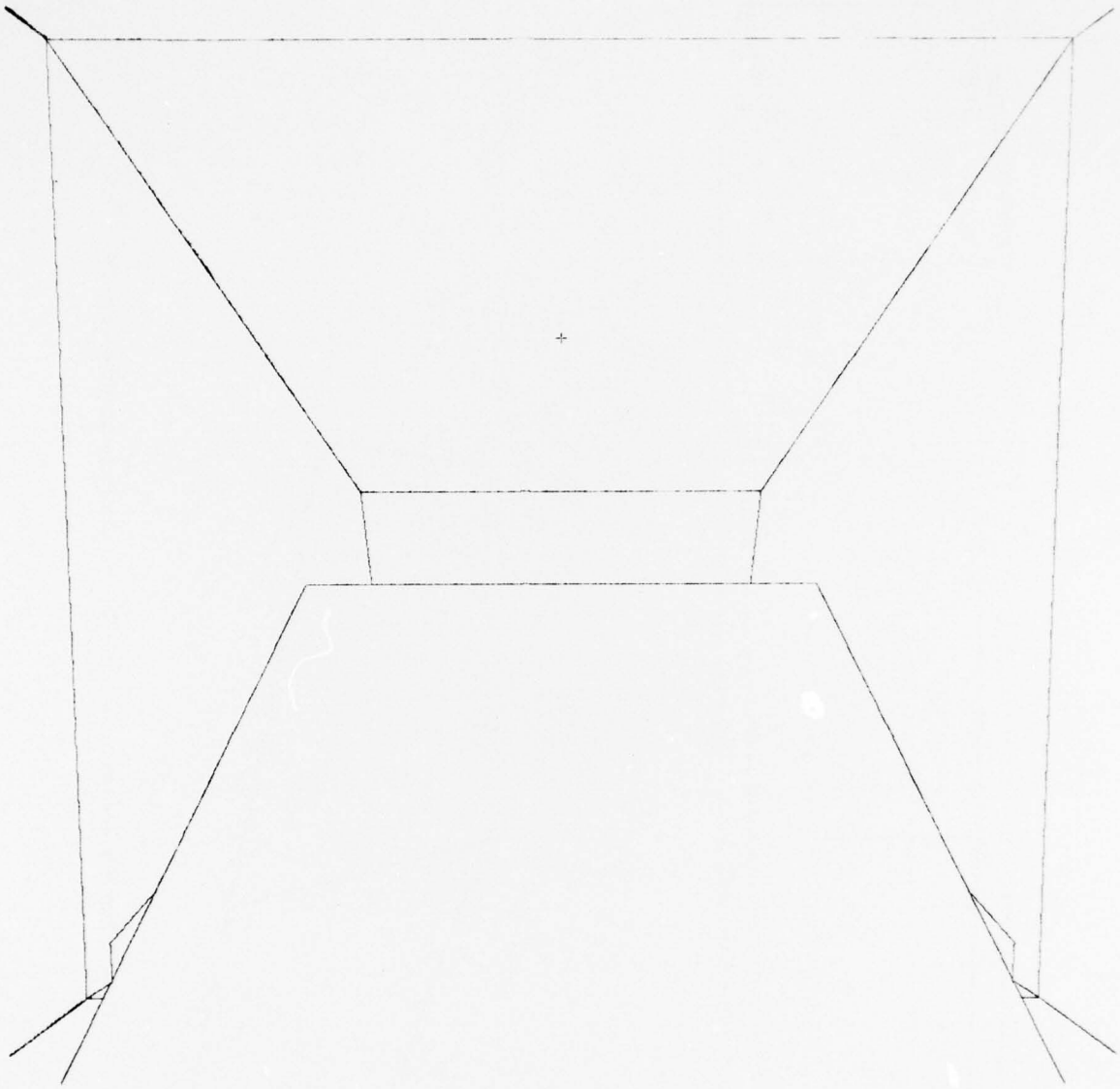


Figure 2. Perspective view of the cockpit interior for the pilot's nominal viewing position and direction.

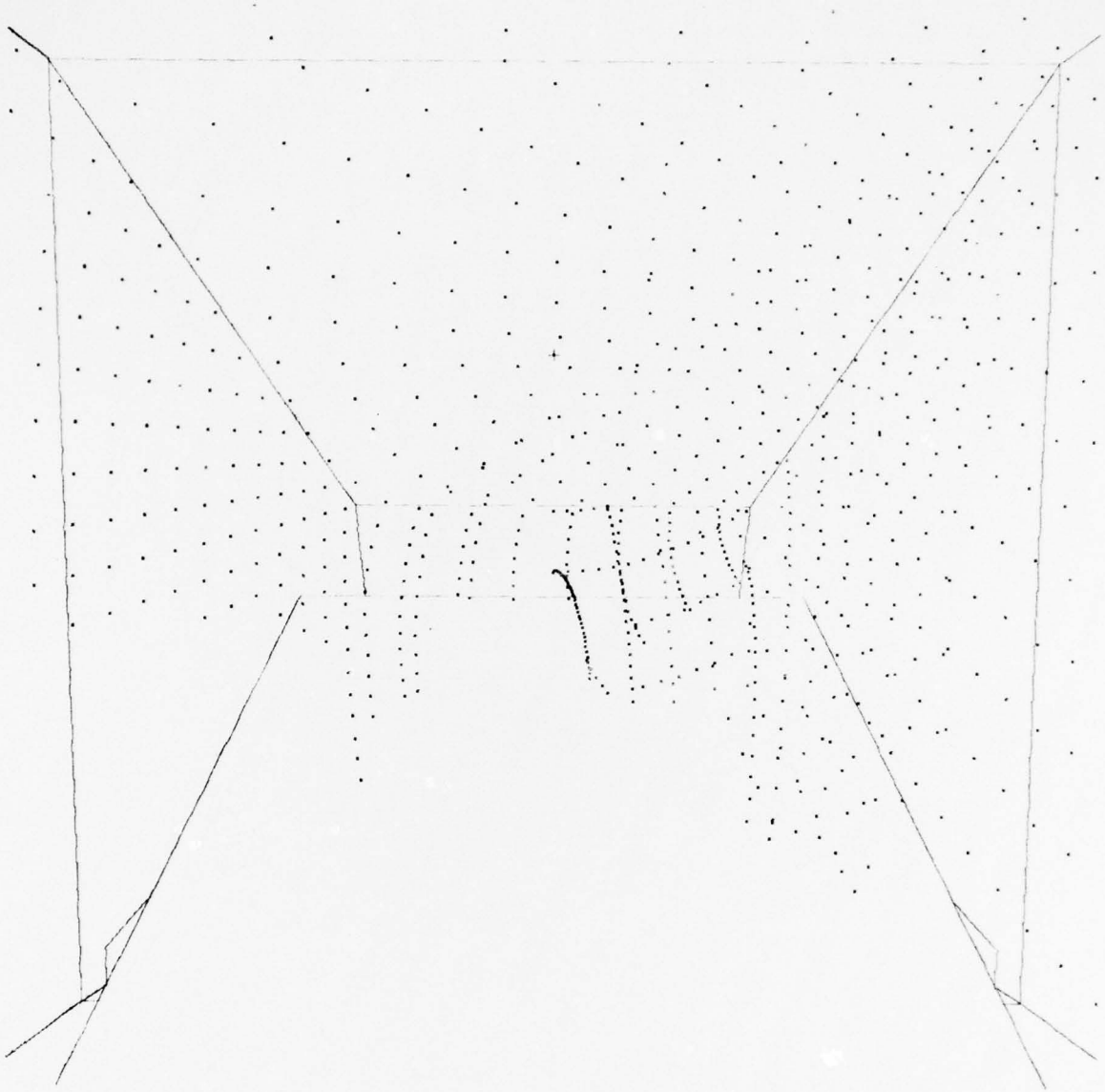


Figure 3. Entry ray positions generating primary reflections on the right-hand side of the canopy.

Figure 4. Primary reflection points on the right-hand side of the canopy and their associated reflectance values.

The points shown on the figures are for primary reflectances only. Secondary and higher order reflections are possible. Entry rays and reflections can occur at surface points which are further back on the canopy. These points are outside of the perspective view. (See reference 5 for a further breakdown of entry point and primary reflection point distributions by canopy surfaces.)

RECOMMENDATIONS FOR FURTHER RESEARCH

The following recommendations are made for further development:

1. Investigate the perspective space for ray tracing in place of the presently used object space.

2. Extend the program to realistic nighttime lighting situations and approach scenarios.

3. Include the visual function of the pilot as a measure of the magnitude of the reflection problem.

4. Extend the program to canopy designs other than the flat-plate canopy.

5. Incorporate similar developments for the daytime-glint problem.

6. Incorporate photometric measurements and subject testing at suitable stages in the development to confirm the validity of the computatory approach.

CONCLUSION

A computer program has been developed by HEL to show internal cockpit reflections of external point light sources. The program has been applied to the Model 209 AH-1S COBRA helicopter with the flat-plate canopy design. The results show that during nighttime operations reflections are possible over much of the canopy internal surfaces. Which of these reflections actually occur depends upon the particular lighting situation and flight scenario.

REFERENCES

1. Fadell, A. G. Vector calculus and differential equations. New York: Van Nostrand Reinhold Co., Inc., 1968.

2. Jenkins, F. A., & White, H. E. Fundamentals of optics. New York: McGraw-Hill, 1957.

3. Judd, D. B., & Wyszecki, G. Color in business, science, and industry. New York: John Wiley & Sons, Inc., 1975.

4. Newman, W. M., & Sproull, R. F. Principles of interactive computer graphics. New York: McGraw-Hill, 1973.

5. Stowell, H. R., & Smyth, C. C. Investigation of inside light reflection problem on the flat plate canopy (FPC) for Model 209 AH-1S helicopter. Technical Memorandum 13-77, US Army Human Engineering Laboratory, Aberdeen Proving Ground, MD, 1977.

6. User's manuals. Higher operating system. December 1974. Fortran graphics II. March 1974. Informations Displays, Inc., Mt. Kisco, NY.

APPENDIX A

INTERNAL REFLECTIONS

The internal reflections are computed using standard vector geometry techniques (1). Consider a standard cartesian coordinate system (x,y,z) with the y-axis taken along the longitudinal axis of the aircraft, the z-axis directed through the canopy top and the x-axis perpendicular to these two. The cockpit and canopy may be described by a set of flat planar surfaces. Each such surface consists of corners and straight edges. It is described by a sequence of adjacent corner vertices and their corresponding coordinates, with the consecutive order determining the direction of the surface normal.

(1) Computation of surface normal

Consider a flat surface with three adjacent vertices none being the same, P_1 (x_1,y_1,z_1), P_2 (x_2,y_2,z_2), and $P_3(x_3,y_3,z_3)$. The surface normal may be determined from the crossproduct and dot product of the vectors $\overrightarrow{P_1P_2}$ and $\overrightarrow{P_1P_3}$. These vectors may be expressed in terms of the unit vectors for the cartesian coordinate as follows:

$$\overrightarrow{P_1P_2} = \Delta x_{21}\vec{i}_X + \Delta Y_{21}\vec{i}_Y + \Delta Z_{21}\vec{i}_Z$$

where $\Delta X_{21} = X_2 - X_1$, $\Delta Y_{21} = Y_2 - Y_1$, and $\Delta Z_{21} = Z_2 - Z_1$. Let P_2 be the length of the vector $\overrightarrow{P_1P_2}$ and P_3 that of $\overrightarrow{P_1P_3}$, i.e.,

$$P_2 = \{\Delta X_{21}^2 + \Delta Y_{21}^2 + \Delta Z_{21}^2\}^{1/2}.$$

The dot product of the two vertice vectors is:

$$\overrightarrow{P_1P_2} \cdot \overrightarrow{P_1P_3} = P_2P_3 \cos\Theta_{23},$$

where Θ_{23} is the angle between the two vectors. This angle is measured in the plane found by the two vectors about the vertex P_1. In terms of the coordinate unit vectors,

$$\overrightarrow{P_1P_2} \cdot \overrightarrow{P_1P_3} = \Delta X_{21} \cdot \Delta X_{31} + \Delta Y_{21} \cdot \Delta Y_{21} + \Delta Z_{21} \cdot \Delta Z_{31}$$

$$\text{Therefore, } \Theta_{23} = \text{ARCCOSINE } \frac{\{\overrightarrow{P_1P_2} \cdot \overrightarrow{P_1P_3}\}}{P_2P_3}$$

The crossproduct of the two vectors is given by:

$$\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3} = \vec{\imath}_n P_2P_3 \sin\theta_{23}$$

where $\vec{\imath}_n$ is the unit vector normal to the surface. In terms of the coordinate unit vectors,

$$\overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3} = (\Delta Y_{21} \cdot \Delta Z_{31} - \Delta Z_{21} \cdot \Delta Y_{31})\,\vec{\imath}_X$$

$$- (\Delta X_{21} \cdot \Delta Z_{31} - \Delta X_{31} \cdot \Delta Z_{21})\,\vec{\imath}_Y$$

$$+ (\Delta X_{21} \cdot \Delta Y_{31} - \Delta X_{31} \cdot \Delta Y_{21})\,\vec{\imath}_Z,$$

The surface normal may be expressed in terms of the coordinate unit vectors as:

$\vec{\imath}_n = a_n \vec{\imath}_x + b_n \vec{\imath}_y + c_n \vec{\imath}_z$, where a_n,b_n, and c_n are the appropriate directional cosines. Combining expressions and using $R=1/P_2P_3\sin\theta_{23}$ we see that

$$a_n = (\Delta y_{21} \cdot \Delta x_{31} - \Delta z_{21} \cdot \Delta y_{31}) \cdot R,$$

$$b_n = -(\Delta x_{21} \cdot \Delta z_{31} - \Delta x_{31} \cdot \Delta z_{21}) \cdot R,$$

$$c_n = (\Delta x_{21} \cdot \Delta y_{31} - \Delta x_{31} \cdot \Delta y_{21}) \cdot R$$

(2) Specification of trace ray direction

A straight ray may be specified by an origin point and a unit vector directed along its length. Consider a spherical coordinate system with the origin at the pilot's nominal eye position and the y-axis parallel to that of the aircraft's cartesian coordinate system. Let α be the angular elevation of a ray from the origin measured from the z-axis. Let β be the azimuth angle of the ray measured from the x-z plane toward the y-axis. The direction of the ray is specified by the angles α, β. Since the axes of the two coordinate systems are parallel, the directional cosines of the ray direction are:

$$a_s = \cos(\beta) \cdot \sin(\alpha)$$

$$b_s = \sin(\beta)$$

$$c_s = \cos(\beta) \cdot \cos(\alpha)$$

The origin point is the pilot's nominal eye position

(x_o, y_o, z_o), i.e.,

$$x_s = x_o$$

$$y_s = y_o$$

$$z_s = z_o$$

(3) Intersection of ray with surface

The intersection point of a line with a planar surface may be computed given the origin and directional cosines of the line and a vertex and normal of the surface. Given the origin x_s, y_s, z_s and the directional cosines of the line a_s, b_s, c_s, the coordinates of the intersection point, P, are:

$$x = x_s + a_s \cdot R$$

$$y = y_s + b_s \cdot R$$

$$z = z_s + c_s \cdot R,$$

where R is the line length between the origin and the intersection point.

Since the intersection point, P, is in the planar surface, then the vector from P to any vertice (assuming they are not the same point), P_1, must be perpendicular to the surface normal. For this reason, the dot product of the two vectors equals zero, i.e., $\overrightarrow{P_1P} \cdot \vec{\imath}_n = 0$. Therefore,

$$a_n(x-x_1) + b_n(y-y_1) + c_n(z-z_1) = 0,$$

and using the above expressions for the intersection point coordinates, the line length is given by:

$$R = \frac{\{a_n \cdot (x_1-x_s) + b_n \cdot (y_1-y_s) + c_n \cdot (z_1-z_s)\}}{a_n \cdot a_s + b_n \cdot b_s + c_n \cdot c_s}$$

(4) Determine if ray intersects surface in outward direction.

The consecutive order of vertices assigned for the cockpit and canopy surfaces is selected so that the surface normals are directed into the cockpit interior. The computed intersection point will be outward directed when the angle between the ray vector and the surface normal exceeds 90^o. This occurs when the dot product between the two vectors:

$$Q = a_s \cdot a_n + b_s \cdot b_n + c_s \cdot c_n,$$

is negative, i.e., $Q \leq 0$. Note that Q is the denominator of the expression for R the line length. This implies that for an outward directed intersection, $R \geq 0$.

(5) Determining if the intersection point is on the canopy surface.

The intersection point lies on the canopy surface when it is contained by the surface edges. Consider one such edge defined by two adjacent vertices P_j, P_{j+1}, and the ray origin, P_s. These three points define a planar constraint surface. A unit normal $\vec{\imath}_c$, can be computed for this surface as above. The unit normal will be directed inward if adjacent points are selected in the same consecutuve order as selected for the canopy surface. The unit normal for the constraining surface is computed from the vectors $\overrightarrow{P_sP_j}$ and $\overrightarrow{P_sP_{j+1}}$. Noting that the angle between these two vectors is geven by:

$$\theta_{j,j+1} = \text{arccosine } \frac{\{\overrightarrow{P_sP_j} \cdot \overrightarrow{P_sP_{j+1}}\}}{P_j \cdot P_{j+1}} ,$$

then,

$$\vec{\imath}_c = \frac{\{\overrightarrow{P_sP_j} \otimes \overrightarrow{P_sP_{j+1}}\}}{P_jP_{j+1} \sin(\theta_{j,j+1})}$$

which may be decomposed into directional cosines a_c, b_c, c_c as in section (a) above.

The intersection point will be within the j^{th} edge if the angle between the line vector and the unit normal of the corresponding constraining surface is less than 90^o. This is true if the dot product between the two vectors,

$$\theta_j = a_s \cdot a_{cj} \cdot b_s \cdot b_{cj} + c_s \cdot c_{cj},$$

is greater than zero, $\Theta_j > 0$. Here j is the number of the first vertex defining the edge. Therefore, the intersection point will lie on a canopy surface with N-vertices if $\Theta_j > 0$ for all j=1, N.

(6) Reflected and transmitted rays

The canopy material is assumed to be dielectric. The incident ray is reflected from the surface and refracted into the material. The unit vector, $\vec{\imath}_s$, for the incident ray direction may be separated into surface components using vector notation. The vector component in the direction of the surface normal is equal to $(\vec{\imath}_s \cdot \vec{\imath}_n)\vec{\imath}_n$, while the component parallel to the surface equals $\vec{\imath}_s - (\vec{\imath}_s \cdot \vec{\imath}_n)\vec{\imath}_n$. These two components are orthogonal and completely define the vector.

The reflected ray lies in the plane defined by the surface normal and the incident ray. The angle between the reflected ray and the surface normal is equal in value but opposite in sign to that between the incident ray and the surface normal. Therefore, the unit vector for the reflected ray equals the surface component for the incident ray minus its normal component, i.e.,

$$\vec{\imath}_R = \vec{\imath}_s - 2(\vec{\imath}_s \cdot \vec{\imath}_n)\vec{\imath}_n$$

This may be expressed in terms of directional cosines for the unit vectors of the coordinate system,

$$a_R = a_s - 2Ra_n,$$

$$b_R = b_s - 2Rb_n,$$

$$c_R = c_s - 2Rc_n,$$

where $R=a_s \cdot a_n b_s \cdot b_n + c_s c_n$. Note that the angle of incidence equals Θ_o=arccosine $(-R)$.

Consideration of electromagnetic boundary conditions at the material surface results in Ferrel's law. Assuming non-polarized light the reflectance is equal to (2).

$$r = \tfrac{1}{2}(r_s + r_p), \tag{1}$$

where,

$$r_s = \left[\frac{\cos\theta_o - \sqrt{n^2 - \sin^2\theta_o}}{\cos\theta_o + \sqrt{n^2 + \sin^2\theta_o}}\right], \text{ and}$$

$$r_p = \left[\frac{n^2\cos\theta_o - \sqrt{n^2 - \sin^2\theta_o}}{n^2\cos\theta_o + \sqrt{n^2 + \sin\theta_o}}\right],$$

The refracted ray lies also in the plane defined by the incident ray and surface normal. The angle of refraction is determined by Snell's law, i.e., $\sin\Theta_o = n\sin\Theta$, where n is the material index of refraction. The surface transmittance is determined from energy considerations. The energy flux in the ray beam is equal to the product of the intensity (i.e., square of electric amplitude times dielectric constant) and the beam crosssectional area. The conservation of energy specifies that the sum of the energy fluxes in the reflected and refracted beams equals that in the incident beam. Assuming distant sources of light so that the wavefronts are closely planar, the surface reflectance and transmittance is related by:

$$r + t\,\frac{\cos\theta}{\cos\theta_o} = 1 \tag{2}$$

The refracted ray is multiply reflected between the top and bottom surfaces of the canopy material. This internally transmitted energy is partially absorbed as it transverses the medium between the surfaces. At each surface the energy remaining is partially refracted from the medium and the rest reflected toward the other side. The same reflection and refraction equations apply for this internal reflections as above except that the angles Θ_o and Θ are interchanged.

The total reflectance and transmittance can be determined by adding together the fractions of energy that leave the top and bottom surfaces. Assume that the top and bottom surfaces are parallel and that the material thickness is infinitesimal. All internally refracted rays leave at the same surface point as does the initially transmitted ray, and this point is the same as that at which the incident ray reaches the surface.

Let r_o and t_o be the surface reflectance and transmittance for the reflection and refraction of an externally incident ray (see equations 1 and 2). Similarly, let r_i and t_i be the internal surface reflectance and transmittance (equations 1 and 2 with Θ substituted for Θ_o). Letting T_i be the internal transmittance of the material, the total reflectance is given by (3).

$$R = r_o + t_o \cdot T_i \ (r_i \cdot T_i \ t_i + r_i^2 \cdot T_i^2 \ r_i \cdot t_i \cdot + \ldots.)$$

and the total transmittance by

$$T = t_o T_i \ (t_i \cdot + R_i^2 \cdot T_i^2 \cdot \ t_i \cdot + \ . \ . \ )$$

In closed form, these equations are

$$R = \frac{r_o + t_o t_i r_i T_i}{(1 - r_i^2 T_i^2)} \qquad (3)$$

and

$$T = \frac{t_o t_i T_i}{(1 - r_i^2 T_i^2)} \qquad (4)$$

The total transmitted ray will exit at the intersection point and have directional cosines equal to those of the incident ray, i.e., $a_T = a_S$, $b_T = b_S$, and $c_T = c_S$.

(7) Repeated computations and interpretation

The above computation process can now be repeated to determine a canopy intersection point for the reflected ray. Put the incident ray equal to the just computed reflected ray, i.e., $a_S=a_R, b_S=b_R, c_S=c_R$, and the ray origin to the intersection point, i.e., $X_S=X_I, Y_S=Y_I, Z_S=Z_I$. Continuing the computation process will determine the next intersection point, the corresponding reflected and transmitted rays, and the reflectance and transmitted values.

Assume that N such successive reflections have been determined starting with a ray from the pilot's nominal eye position. One interpretation of the results is that the transmitted ray of the $N^{\underline{th}}$ computation stage is the negative of an incident ray from an external light source. This ray is transmitted through the canopy at the $N^{\underline{th}}$ intersection point with transmission T(N). The ray is then reflected in the direction opposite to that computed until it reaches the pilot's eye position. The final reflectance value is equal to:

$$R=T(N)\cdot \prod_{i=1}^{N-1} R(i),$$

where R (i) is the reflectance computed at the $i^{\underline{th}}$ intersection point. Note that for $N \geq 1$, $R(0)=1$.

APPENDIX B

PERSPECTIVE DRAWINGS

Perspective drawings of the cockpit as seen from the pilots' nominal eye position have been prepared using a computer-graphics terminal. The pilots' viewing direction is assumed to be fixated along a line which is directed five degrees above the aircraft's longitudinal axis (negative y-axis). The fixation line lies in the y-z plane of the aircraft's coordinate system.

A perspective drawing shows a three-dimensioal view of objects on a two-dimensional scene. The drawing should be held by the viewer at a specified distance (say $\underline{a}$) from his eye position and in a plane normal to this fixation direction. The two-dimensional scene is normally bordered on four sides (each of length, say $2\underline{b}$). Not shown on the scene are objects which would appear outside of this border or behind the viewer. The scene may be prepared on a graphics terminal of dimension, say $2\underline{c}$. The drawings shown in this report fill a 10-inch by 10-inch scene and have a 10-inch viewing distance. The graphics terminal used in preparation has a span of 1024 rasters (a=10 inches, b=5 inches, and c=512).

The viewer's eye position may be considered as forming the apex for a four-sided viewing pyramid. The sides of the pyramid are determined by the apex and the borders of the scene. The objects within the viewing pyramid are projected as images onto the scene surface. A straight line connecting the apex with a vertex of an object will pass through the corresponding vertex of the object's image. Consequently, the image is described by its vertices in the scene and the edges connecting them.

Each object is described by its surfaces and the corresponding vertices and edges. Only the images of surfaces which are facing the viewer should be drawn, since the back surfaces are hidden from view. In some cases, a surface will be partially hidden from view by intermediate objects and only an image of the visible portion should be drawn. Still other objects may be completely hidden from view by intermediate objects. Other objects will be outside of the viewing pyramid and completely hidden from view. Finally, some remaining objects will be partially inside the viewing pyramid and partially outside.

The construction of perspective drawings follows a three-stage decision process: First, find those surfaces which face toward the viewer. Next, find which of these surfaces are partially or completely within the viewing pyramid. Finally, determine what portions, if any, of the surface edges are not hidden by the sides of the pyramid or intermediate objects.

The above decision process is implemented by a seven step analysis sequence (4).

1. Compare each surface normal to the viewer's direction and discard surfaces which do not face the viewer.

2. Convert the coordinates of the surface vertices from the aircraft object space to the viewer's space.

3. Trim off surface edges which extend behind the viewer.

4. Convert the vertex coordinates from the viewer's space to a scene display space in which the viewing pyramid has become a view box.

5. Discard surfaces which lie completely outside the view box.

6. Determine what portion of a surface edge lies within the viewing box.

7. Determine the apparent crossing points of the surface edge with the edges of every other surface. Determine whether the points are on constraining edges of another surface, whether they are in front of the test edge, and finally, what portion is blocked by the constraining surface.

This sequence determines the end points of the visible portion of each surface edge and the resulting image lines are drawn on the perspective drawing.

(1) Discarding surfaces not facing the viewer.

The consecutive ordering of surface vertices was chosen so that surface normals are are directed into the cockpit volume. Consider a line from the pilot's nominal eye position (x_o,y_oz_o) to a vertex on a surface P_1 (x_1,y_1,z_1). The directional cosines (a_s,b_s,c_s) of a unit vector directed along this line are computed from the differences in coordinate values and the line length, R_s, i.e.,

$$a_s = (x_1 - x_o) / R_s,$$

$$b_s = (y_1 - y_o) / R_s,$$

$$c_s = (z_1 - z_o) / R_s$$

Consider now the dot product between the unit vector along this line and the unit normal at the surface. The unit normal is described by its directional cosines (a_n,b_n,c_n) and the dot product is given by

$$\Theta = a_s \cdot a_n + b_s \cdot b_n + c_s \cdot c_n$$

This value is equal to the cosine of the angle betwen the two vectors. The surface normal is directed back toward the viewer when this angle is greater than 90^o. In this case the dot product value is less than zero, i.e., $\Theta < 0$.

(2) Converting the coordinates of surface vertices to the viewer's space.

The coordinates of the surface vertices are transformed from the aircraft's coordinate system (x,y,z) to the viewer's coordinate system (x^1,y^1,z^1) in two steps. The origin of the coordinate system is translated to the pilot's nominal eye position. The coordinate axes are then rotated about the origin so that the transformed y^1-axis is aligned along the pilot's nominal fixation direction. The x^1-axis remains in the x-y plane of the aircraft space.

The aircraft's coordinate system is a left-handed cartesian coordinate system with the y-axis directed backward along the aircraft's longitudinal axis. Consider now a translation of the coordinate origin to the pilot's nominal eye position, and let the y-axis be directed in the opposite direction to form a right-handed cartesian coordinate system. The pilot's fixation direction may be specified in this coordinate system by the spherical angles α and β (see Appendix 2A)). The angular elevation, α, of a ray from the origin is measured from the z-axis. The azimuth angle β is measured from the x-z plane in the x-y plane.

The coordinate system may be aligned with the fixation direction by: (1) a clockwise rotation about the z-axis equal to the complement of the angle, β, and (2) a counterclockwise rotation about the x-axis equal to the complement of the angle, α. The rotations are performed about the origin in the appropriate directions when viewed from the positive sides of the axes. The above translational and rotational transformations may be described by concatenated matrices.

Let (x,y,z) be a point in the aircraft's space and (x^1,y^1,z^1) the equivalent point in the pilots' viewing space, then in matrix notation –

$$(x^1,y^1,z^1,1) = (x,y,z,1) \cdot M_T \cdot M_Z \cdot M_X$$

The translational matrix is given by:

$$M_T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -Y_o & -Z_o & 1 \end{bmatrix}$$

and the two rotational matrices by:

$$M_Z = \begin{bmatrix} \sin\beta & -\cos\beta & 0 & 0 \\ \cos\beta & \sin\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

and

$$M_X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & -\cos\alpha & \sin\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The order of application of the transformations are to be preserved when the matrices are multiplied together. The resulting equations are:

$$X^1 = X \cdot \sin\beta - (y-y_o)\cos\beta$$

$$Y^1 = -x \cdot \cos\beta\sin\alpha - (y-y_o)\sin\beta\sin\alpha - (z-z_o)\cos\alpha$$

$$Z^1 = -x\cos\beta\cos\alpha - (y-y_o)\sin\beta\cos\alpha + (z-z_o)\sin\alpha$$

In the case considered in this report, $\beta = 90^o$ and $\alpha = 85^o$. The above equations reduce to:

$$X^1 = X$$

$$Y^1 = -(y-y_o)\cos(5^o) - (z-z_o)\sin(5^o)$$

$$Z^1 = -(y-y_o)\sin(5^o) + (z-z_o)\cos(5^o)$$

(3) Trimming surface edges lying behind the viewer —

Our objective is to place a two-dimensional scene on a plane which is normal to the fixation direction (i.e., the viewer's y-axis) and at a distance a from the viewer's eye. The scene is to be identical in appearance to the three-dimensional view contained within the scene borders. It will be to our advantage to trim surface-edges which extend behind the viewer, for reasons given later.

Consider a trimming plane normal to the y-axis and placed a minute distance, y_t, in front of the eye position. Surfaces behind the trimming plane are to be discarded, while surfaces penetrating the plane are to be trimmed to reduced forms. In a sense, we are considering a truncated viewing pyramid, since surfaces between the trimming plane and the eye position will be discarded.

Consider the starting and ending vertices of each edge on a surface. If the value of the y-coordinate of a vertex point is less than y_t, then the vertex must be behind the trimming plane. Let the j^{th} edge be defined by the j^{th} vertex, $P_j(X_j,Y_j,Z_j)$, for a starting point and the $j+1^{th}$ vertex, $P_{j+1}(X_{j+1},Y_{j+1},Z_{j+1})$, for an ending point. The following cases are possible: (1) if y_j and Y_{j+1} are both larger than y_t, then the j^{th} edge is inside the trimming edge, (2) if $Y_j \geq y_t$, but $y_{j+1} < y_t$, then the edge crosses out of the trimming plane, (3) if y_j and $y_{j+1} < y_t$, then the edge lies outside the plane, and (4) if $y_j < y_t$, but $y_{j+1} \geq y_t$, then the edge crosses inside the plane.

We may construct an equivalent but reduced surface by defining a set of vertices $\{P_{ek}\}$ as follows:

(1) If y_j and y_{j+1} are both larger than y_t, then set the coordinates of P_{ek} equal to those of P_j i.e., $x_{ek}=x_j, y_{ek}=y_j, z_{ek}=z_j$.

(2) If $y_j \geq y_t$, but $y_{j+1} < y_t$, then the coordinates of P_{ek} equal to those of P_j as above, but set those of P_{ek+1} equal to those of P_{Ij+1}. Here P_I is the intersection point of the edge with the trimming plane (see below).

(3) If y_j and $y_{j+1} < y_t$, then discard the edge.

(4) If $y_j < y_t$, but $y_{j+1} \geq y_t$, then set the coordinates of P_{ek} equal to those of P_I defined above.

Note that if all vertices are outside of the trimming plane, this construction will discard the corresponding surface.

The coordinates of the intersection point that an edge makes with the trimming plane may be computed using the technique of Appendix 3A. Note that the directional cosines of the normal to the trimming plane are (0,1,0) and that a convient vertex point for this plane is $(0,y_t,0)$. Consider the $j^{\underline{th}}$ edge with starting point, P_j and ending point, P_{j+1}. The directional cosines for a unit vector along the edge are given by:

$$a_s = (x_{j+1} - x_j)/R_e,$$

$$b_s = (y_{j+1} - y_j)/R_e,$$

$$c_s = (z_{j+1} - z_j)/R_e,$$

where the edge length is given by:

$$R_e = \{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2 + (z_{j+1} - z_j)^2\}^{1/2}.$$

The distance along the edge from the starting point to the intersection point is given by (see Appendix 3A).

$$R_I = (y_t - y_j)/b_s.$$

The coordinates of the intersection point are given by:

$$X_I = X_j + a_s R_I$$

$$Y_I = Y_j + b_s R_I$$

$$Z_I = z_j + c_s R_I$$

(4) Converting the coordinates of surface vertices to a scene display space.

The scene coordinates (X_D, Y_D) *for an image point of a surface vertex,* $P(x,y,z)$ *in the* viewer's space, are defined by the intersection of a straight line with the scene plane. The line runs from the viewer's eye position to the vertex point. This line along with the y-axis of the viewer's space forms two congruent triangles, one for the image point and the other for the vertex point. The ratio of the eye-scene distance, a, to the image-point distance normal to the y-axis, is equal to the ratio of the y-distance for the vertex point to the corresponding normal distance, i.e.

$$X_D = \frac{X}{Y} \cdot a,$$

$$Z_D = \frac{Z}{Y} \cdot a.$$

Consider the addition of a third coordinate to add depth thereby forming a display space,

$$Y_D = -1/Y.$$

This transformation converts the viewer's eye position to minus infinity. All lines from the eye position to surface vertex points are parallel to the Y_D-axis. The (truncated) viewing pyramid is converted into an infinitely deep viewing box. The four sides of the viewbox parallel the Y_D-axis. Assuming that the scene borders parallel the scene coordinate axes (X_D,Y_D), then the viewbox sides lie in the planes $X_D=\pm b$ and $Z_D=\pm b$.

Edges which pass through the trimming plane in the viewer's space are difficult to trace in the display space. The portion of the edge in front of the plane is transformed into a line running from the endpoint image to positive infinity. On the other hand, the portion behind the plane is transformed into a line running from negative infinity to the other endpoint image. This is not readily apparent when the two endpoints are considered alone. Note also that placing the trimming plane at a minute distance, Y_t, in front of the viewer precludes division by zero when transforming trimmed edges.

(5) Discarding surfaces completely outside of the display space viewbox.

A surface vertex will not be seen if the image coordinates exceed the scene bounds of length 2b. Consider the coordinate conversions:

$$X_D^1 = X_{D/b}, \text{ and } Z_D^1 = Z_{D/b}.$$

If either the absolute value of X_D^1 or Z_D^1, or both are larger than unity, then the vertex image should not be in the scene.

(6) The clipping of surface edges protruding beyond the viewing box –

The surface edge is tested against the viewbox in display space. A starting point (x_v,y_v,z_v), directional cosines (a_v,b_v,c_v) and edge length, R_v, are computed from the end vertices (P_j,P_{j+1}). Let R_o be the distance along the edge line from the starting point to the initial point of the edge portion within the viewbox and R_F the distance to the final point. Initially set RO=0 and $R_F=R_v$ when entering the testing procedure. Note that if R_O is larger than R_F, then no portion of the edge is within the viewbox.

The unit vector normal to the surface of a viewbox side is assumed to be directed into the viewbox interior. Letting the directional cosines of the surface normal be denoted by a_n,b_n,c_n, a convenient vertex (x_n,y_n,z_n) within the surface side is determined by $x_n=-b\cdot a_n$, $y_n=b\cdot b_n$, and $z_n=-b\cdot c_n$.

Consider the dot product of the unit vector taken along the edge line and the surface normal of the viewbox side, i.e.,

$$Q = a_v \cdot a_n + b_v \cdot b_n + c_v \cdot c_n$$

If the vectors are orthogonal, then the angle between them equals 90^o and Q equals zero. In this case, the edge line parallels the side surface, and the starting point (x_v,y_v,z_v) must be tested to determine whether or not it is within the viewbox (see section 5, above).

If the dot product, Q, is not equal to zero then the edge line intersects the side surface. The distance along the edge line from the starting point to the intersection point is given by (see Appendix 3A).

$$R_I = ((X_I-X_v) \cdot a_n + (y_n-y_v) \cdot b_n + (z_n-z_v) \cdot c_n)/Q.$$

If the edge line crosses into the side surface from outside the viewbox, then the angle between the two vectors is less than 90^o and the value of Q is greater than zero. In this case given that the distance R_I is greater than R_o, then the initial point of the contained portion must be moved forward to the intersection point, i.e., $R_o=R_I$.

If, however, the edge line is directed outward from inside the viewbox, then the angle between the two vectors is greater than 90^o and Q is less than zero. In this case given that the distance R_I is less than R_F then the end point of the contained portion must be moved back to the intersection point, i.e., $R_F=R_I$.

(7) Removal of hidden lines on surface edges and display of visible segments.

The clipping routine determines which segment of a surface edge is contained within the viewbox. The test edge is specified by: (1) a starting point (x_v,y_v,z_v), (2) directional cosines (a_v,b_v,c_v) and (3) a length, R_v. The contained segment is determined by its initial and final endpoints on the edge. The endpoints can be described by their distances along the edge line from the starting point. Let R_o denote the distance of the initial point and R_F that of the final point. Note that if R_o equals zero and R_F equals R_v, than the complete edge is contained within the viewbox. If R_o is greater than zero or R_F is less than R_v or both, then only a portion of the edge is contained. However, if R_o exceeds R_F in value, than no segment can be contained.

The contained segment of the edge may be either partially or completely hidden from view by intermediate surfaces. Assume all surfaces to be convex. That is, any point defined as the linear sum of edge points each times a coefficient the sum of which equals unity, is on the surface contained by the edges. In this case, a line running along the test edge will appear to the viewer to intersect a constraining surface at two points. Each point is the apparent crossover of the test line with an edge of the constraining syrface. For this reason, an intermediate surface can mask no more than a single line segment from view.

Consider the apparent crossover point between the test edge and a constraining edge. A starting point (x_e,y_e,z_e), directional cosines (a_e,b_e,c_e) and line length R_e, may be computed for the constraining edge given the coordinates of the starting and ending vertices (P_j, P_{j+1}). The image lines of the two edges in the display scene will not appear parallel if the y-component of the crossproduct is unequal to zero, i.e., the value of $Q=a_e \cdot c_v - a_v \cdot c_e$, does not equal zero. In this case there is an apparent crossover point. It is determined in the display space by the intersections of the two edges with a straight line parallel to the y-axis and from the viewer's position. The intersection points have the same display coordinates (x_I,y_I), but different dept values (y_{Ie},y_{Iv}). Let R_{Iv} denote the distance from the starting point of the test edge to the intersection point with the viewing line. Let R_{Ie} denote the equivalent distance for the constraining edge. The display coordinates for the intersection points are given for the constraining edge by —

$$X_I = x_e + a_e \cdot R_{Ie},$$

$$Z_I = y_e + c_e \cdot R_{Ie},$$

and for the test edge by:

$$x_I = x_v + a_v \cdot R_{Iv},$$

$$z_I = y_v + c_v \cdot R_{Iv}.$$

Solving the above equations for the intersection distance on the test edge results in:

$$R_{Iv} = (c_e \cdot (x_e - x_v) - a_e \cdot (z_e - z_v))/Q.$$

Similarly, solving for the intersection distance on the constraining edge results in:

$$R_{Ie} = (z_v - z_e + c_v \cdot R_{Iv})/c_e,$$

or equivalently,

$$R_{Ie} = (x_v - x_e + a_v \cdot R_{Iv})/a_e.$$

The constraining edge will be part of a masking surface if two conditions exist. First, the intersection point must be within the edge endpoints. That is, the distance R_{Ie} must be greater than zero, i.e., $R_{Ie} > 0$, and less than the edge length, i.e., $R_{Ie} < R_e$. Furthermore, the constraining edge must be between the viewer and the test edge. The distance along the viewing direction (i.e., the y-axis) to the intersecting point on the test edge is given by:

$$y_{Iv} = y_v + b_v \cdot R_{Iv},$$

and the distance to the intersection point on the constraining edge is given by:

$$y_{Ie} = y_e + b_e \cdot R_{Ie}.$$

The constraining edge will appear in front of the test edge if the distance y_{Ie} is larger than y_{Iv}, i.e., $y_{Ie} > y_{Iv}$.

A masking surface will intersect the test edge line at two points. Let the distance from the test edge starting point to the closest intersection point be denoted by R_1, while that to the farthest intersection point be R_2. Note that R_2 is larger than R_1, i.e. $R_2 > R_1$. Four masking cases are possible:

(1) The surface may mask a segment of the test edge line which is outside of the edge limits. In this case R_1 is larger than the edge length, R_v or R_2 is less than zero. The constraining surface does not mask the test edge and may be ignored.

(2) The surface may mask the entire edge. In this case R_1 is less than zero and R_2 is greater than R_v. No segment of the edge is visible.

(3) The surface may mask a central portion of the test edge. In this case R_1 is greater than zero and R_2 is smaller than R_v. The masked segment extends from R_1 to R_2.

(4) The surface masks one end point of the edge but not the other. Here the above three conditions do not apply, and (a) R_1 is greater than zero or (b) R_2 is less than R_v. In the former case the masked segment extends from the starting point to R_1. In the latter case it extends from R_2 to the end point.

The test edge may be masked by more than one constraining surface. Consider a file of N visible segments for the test edge, $\{RO(i),RF(i), i=1,N\}$. Here RO(i) is the distance from the edge starting point to the initial point of the i^{th} segment and RF(i) is the corresponding distance for the end point. Initially, this file may be generated using the end point distances for the edge segment contained within the viewbox, i.e., N=1, $RO(1)=R_o$ and $RF(1)=R_F$ (see section 6 above). The test edge may then be compared against each of the other surfaces in turn and the change in the display file computed as each masking surface is located.

Suppose that a masking surface is in one of the above cases. The corresponding change to the display file is as follows:

(1) The constraining surface does not mask the test edge. No change in visible segments occurs.

(2) The constraining surface masks the entire edge. The number of visible segments wihtin the file is set equal to zero.

(3) The constraining surface masks a central portion of the test edge which extends from R_1 to R_2. Considering each of the visible segments in turn we note that for the i^{th} segment:

(a) If RF(i) is greater than R_1 but less than R_2 the segment protrudes into the masked region and RF(i) must be set equal to R_1.

(b) If RO(i) is greater than R_1 and RF(i) is less than R_2, the segment is masked and it must be removed from the file and the file number reduced by one.

(c) If RF(i) is greater than R_2 and RO(i) is less than R_1, the segment straddles the masked section. The segment is separated into two segments by setting $RO(k) = R_2$,RF(k)=RF(i) and $RF(i)=R_1$. The file number is increased by one to include the segment, k.

(d) If RO(i) is greater than R_1 but less than R_2 and RF(i) is greater than R_2, the segment protrudes out of the masked region. In this case $RO(i)=R_1$.

(4) The constraining surface masks one end of the test edge. If the starting end is masked let $R_{vo}=R_2$ and $R_{vf}=R_v$. If the terminal end is masked let $R_{vo}=0$ and $R_{vf}=R_1$. Then considering each of the visible segments in turn, we note that for the i^{th} segment:

(a) If $RF(i)$ is less than R_{vo} the segment is masked; it must be removed from the file, and the file number reduced by one.

(b) If $RO(i)$ is less than R_{vo}, the segment protrudes out of the masked region and $RO(i)=R_{vo}$.

(c) If $RF(i)$ is greater than R_{vf}, the segment protrudes into the masked region and $RF(i)=R_{vf}$.

(d) If $RO(i)$ is greater than R_{vf} the segment is masked; it must be removed from the file number reduced by one.

The result is a file of N visible segments for each surface edge. The file contains an initial and final distance for each visible segment. The scene coordinates for the image points of each of the segment vertices are for the initial vertex,

$$x_I = x_v + a_v \cdot RO(i),$$

$$z_1 = z_v + c_v \cdot RO(i),$$

and for the final vertex,

$$x_F = x_v + a_v \cdot RF(i)$$

$$z_F = z_v + b_v \cdot RF(i).$$

The scene coordinates are converted to the display screen coordinates (x_s,z_s) using $x_s=(x_D+1)\cdot C$ and $z_s=(z_D+1$. Here $2\underline{c}$ is the screen dimension (C=512). A display line element is then drawn between the starting and ending vertices for each of the N visible segments.

APPENDIX C

COMPUTER PROGRAM

The computer program is programmed for a disk-based batch operating system in Fortran IV language (6). A listing of the subroutines follows. The program is attached.

1. CONTL - controls canopy data read in, computation of reflection points, ray indexing, file writing, and data output.

2. READV - reads in from cards the number of vertices, the number of canopy surfaces, the coordinates of each vertex, the number of vertices for each surface, and the assignment of vertices to each surface in rotational order. Reads in also coordinates of pilot's nominal eye position. Called by CONTL.

3. NORML - uses coordinates of vertices and assignment of vertices to canopy surfaces to determine coordinates of surface vertices. Uses these coordinates and order of vertice assignment to compute surface normals. Writes computed data into disk file. Called by CONTL.

4. READF - reads in previously stored canopy data from desk file to common area. Called by CONTL.

5. TELTY - prints out canopy data from common area. Called by CONTL.

6. INTEC - computes intersection point of straight line with planar surface given origin and directional cosines of line and coordinates of surface vertex and directional cosines of surface normal. Determines if intersection point is contained by edges defined by surface vertices. Called by CONTL.

7. COMP - computes incident angle between ray and surface normal at intersection point, and the corresponding reflectance and transmittance. Called by CONTL as a function of INTEC return.

ASIN

ACOS - computes arcsine and arccosine of function value. Called by COMP and other routines.

9. DRWCN - generates side, top and front views of canopy as a check on data input. Initiates display, sets up display entities in display file using canopy data in common area, and starts display. Uses Fortran graphics commands. Called by CONTL.

10. EMARK

ETIC - Loads display file with line elements for display entity showing nominal eye position in canopy views. Called by DRWCN. ETIC called also by PERP.

11. LINS

LINF

LINT - loads display file with line elements for canopy surface entities. Called by DRWCN.

12. PERP - generates perspective drawing of cockpit as seen from pilot's nominal eye position and shows entry and reflection points. Initiates display, loads display file using canopy data in common, reads disk file for computed entry and reflection data, selects data for showing according to parameters passed by calling routine, and starts display. Called by CONTL.

13. EFIX - uses INTEC to determine coordinates of fixation point upon canopy surface for pilot's nominal eye position and viewing position. Called by PERP.

14. PLAC - converts coordinates of vertex in object space to their equivalent values in perspective space. The origin of this space is the pilot's nominal eye position and the y-axis is along his nominal viewing direction. Called by PERP and TRSDC.

15. TRSDC - converts coordinates of surface vertices from object space to equivalent values in display space. Called by PERP.

16. INTEP - removes portions of figure edges which lie behind viewer. Called by TRSDC.

17. TEST - determines whether surface vertex is outside of display space viewbox. Called by TRSDC.

18. CLIPL - computes apparent intersection points of test edge line with masking surfaces and generates visible portion on display. Called by PERP.

19. CHKS - computes portion of test edge contained within display space viewbox. Called by CLIPL.

20. ORDLN - orders apparent intersection points of test edge with edges of masking surface. Called by CLIPL.

21. CKLIN - determines visible portions of test edge remaining after surface masking. Called by CLIPL.

21. GIN, GBEG, GPUT, GEON and GSTART are Fortran graphics routines. SEEK, GETDEV and GEVT are system file control routines. See reference 6 for details.

```
/JOB(MCAN)
/CREATE(FINKSH)
/FORT
C MAINLINE, RAY TRACING FOR PILOT
C 3 DIMENSIONAL FLAT PLATE CANOPY WITH OBSTRUCTIONS
      SUBROUTINE CONTL
      COMMON/FORG/LFG
      COMMON/GAREA/IDFIL(10000)
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/LINE/AS,BS,CS,XS,YS,ZS
      COMMON/PILOT/XO,YO,ZO
      DIMENSION AN(2)
      DATA AN/2HYS,2HNO/
      DATA A11,A12/2.,2./
      DATA PI/3.14159/
      CALL GETDEV(LUN,'FINKSH',1C)
      CALL GETDEV(LFG,'FORGRA',-1)
      WRITE(1,998)
  998 FORMAT(2X,'CANOPY DATA ON FILE$ (YS OR NO)')
      READ(1,1001)A
      IF(A.EQ.AN(1)) GO TO 1
      CALL READV
      CALL NORML
      GO TO 2
    1 CONTINUE
      CALL READF
    2 CONTINUE
      WRITE(1,1000)
 1000 FORMAT(2X,'PRINTOUT CANOPY DATA$ (YS OR NO)')
      READ(1,1001)A
 1001 FORMAT(1A2)
      IF(A.EQ.AN(1)) CALL TELTY
      WRITE(1,1002)
 1002 FORMAT(2X,'DISPLAY CANOPY CONFIGURATION$ (YS OR NO)')
      READ(1,1001)A
      IF(A.EQ.AN(2)) GO TO 3
      CALL DRWCN
      WRITE(1,1003)
 1003 FORMAT(2X,'HARD COPY$ (YS OR NO)')
      READ(1,1001)A
      IF(A.EQ.AN(2)) GO TO 3
      WRITE(1,1004)
 1004 FORMAT(2X,'PRINTER, TURN ON')
      PAUSE
      CALL GS31(1)
    3 CONTINUE
      WRITE(1,1005)
 1005 FORMAT(2X,'DISPLAY PERPECTIVE$ (YS OR NO)')
      READ(1,1001)A
      IF(A.EQ.AN(2)) GO TO 4
      CALL PERP(C)
      WRITE(1,1003)
      READ(1,1001)A
      IF(A.EQ.AN(1)) CALL GS31(1)
    4 CONTINUE
      CALL SEEK(LUN,0)
      READ(LUN)NVW
      WRITE(1,999)
  999 FORMAT(2X,'COMPUTE PRIMARY REFLECTION POINTS$ (YS OR NO)')
```

```
      READ(1,100)A
      IF(A.EQ.AN(2)) GO TO 300
      IPC=0
      WRITE(1,996)
  996 FORMAT(2X,'PRINTOUT COMPUTATIONS$ (YS OR NO)')
      READ(1,100)A
      IF(A.EQ.AN(1)) IPC=1
      NVLL=0
      CALL SEEK(LUN,NVW)
      WRITE(LUN)NVLL
C INDEX PILOT VIEWING DIRECTION
C A1 ELEVATION, ANGLE FROM Z AXIS TOWARD X AXIS IN X Z PLANE
C A2 AZIMUTH, ANGLE FROM X Z PLANE TOWARD Y AXIS
C Z AXIS TOWARD UPWARD, X AXIS TOWARD LEFT FACING FRONT OF CANOPY AND Y AXIS
C TOWARD BACK OF CANOPY ALONG LONGITUDINAL AXIS OF AIRCRAFT
      A1=0.
   10 A2=AI2
   11 A2=A2-AI2
      IF(A2.GE.-90.) GO TO 15
      A1=A1+AI1
      IF(A1.GT.180.) GO TO 299
      WRITE(1,995)A1
  995 FORMAT(2X,F10.4)
      GO TO 10
   15 CONTINUE
      A1M=A1*PI/180.
      A2M=A2*PI/180.
C DETERMINE DIRECTIONAL COSINES OF VIEWING DIRECTION
      AS=COS(A2M)*SIN(A1M)
      BS=SIN(A2M)
      CS=COS(A2M)*COS(A1M)
      R=1.
      XS=XO
      YS=YO
      ZS=ZO
      IPUN=0
C DETERMINE SURFACE INTERCEPT
   17 CONTINUE
      DO 20 IS=1,NP
      ISK=0
      CALL INTEC(ISK,IS,XR,YR,ZR)
      IF(ISK.GT.0) GO TO 25
   20 CONTINUE
      GO TO 11
   25 CONTINUE
      IF(IS.GT.NB) GO TO 26
      GO TO 20
   26 CONTINUE
      IF(IS.LE.NC) GO TO 11
      CALL COMP(ANG,IS,RT,TT)
      T=TT*R
      R=RT*R
      IF(R.LT..0001) GO TO 11
      IPUN=IPUN+1
      AI=-AS
      BI=-BS
      CI=-CS
      RX=AS*AXN(IS)+BS*AYN(IS)+CS*AZN(IS)
      AR=-AS+2.*RX*AXN(IS)
      BR=-BS+2.*RX*AYN(IS)
```

```
      CR=-CS+2.*RX*AZN(IS)
      IF(IPLN.EQ.2) GO TO 30
      IPP=IS
      ASP=AS
      BSP=BS
      CSP=CS
      XP=XR
      YP=YR
      ZP=ZR
      AIP=AI
      BIP=BI
      CIP=CI
      ARP=AR
      BRP=BR
      CRP=CR
      AS=-AR
      BS=-BR
      CS=-CR
      XS=XR
      YS=YR
      ZS=ZR
      GO TO 17
   30 CONTINUE
      NVLL=NVLL+1
      WRITE(LUN)A1,A2,ASP,BSP,CSP,IPP,XP,YP,ZP,AIP,BIP,CIP,ARP,BRP,CRP,
     CR,IS,XR,YR,ZR,AI,BI,CI,AR,BR,CR,T
      IF(IPC.EQ.0) GO TO 11
      WRITE(3,8CC)A1,A2,ASP,BSP,CSP
      WRITE(3,8C1)IPP,XP,YP,ZP,AIP,BIP,CIP,ARP,BRP,CRP,R
      WRITE(3,8C1)IS,XR,YR,ZR,AI,BI,CI,AR,BR,CR,T
  8CC FORMAT(2X,2(F7.2,2X),3(F7.4,2X))
  8C1 FORMAT(2X,I3,3(F6.2,2X),7(F7.4,2X))
      GO TO 11
  299 CONTINUE
      CALL SEEK(LUN,NVW)
      WRITE(LUN)NVLL
  3CC CONTINUE
      WRITE(1,1CC6)
 1CC6 FORMAT(2X,'PERPECTIVE'/2X,'RAY ENTRENCE POINTS'/2X,'RIGHT HAND SID
     CE COCKPIT (YS OR NO)')
      READ(1,1CC1)A
      IF(A.EQ.AN(2)) GO TO 40
      CALL PERP(1)
      WRITE(1,1CC3)
      READ(1,1CC1)A
      IF(A.EQ.AN(1)) CALL GS31(1)
   40 CONTINUE
      WRITE(1,1CC7)
 1CC7 FORMAT(2X,'PERPECTIVE'/2X,'PRIMARY REFLECTIONS$')
      READ(1,1CC1)A
      IF(A.EQ.AN(2)) GO TO 41
      CALL PERP(2)
      WRITE(1,1CC3)
      READ(1,1CC1)A
      IF(A.EQ.AN(1)) CALL GS31(1)
   41 CONTINUE
      CALL DEVT(LUN,C,0)
      RETURN
      END
/FOPT
```

```
C
C
C READ IN SURFACE VERTICES
      SUBROUTINE READV
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/VERT/XV(100),YV(100),ZV(100),NVR(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/PILOT/XC,YC,ZC
      READ(2,1000)
 1000 FORMAT()
      READ(2,1001)NT,NP,NC,NB
 1001 FORMAT(/4(2X,I2))
      READ(2,1000)
      READ(2,1002)(NV(I),I=1,NP)
 1002 FORMAT(18(2X,I2))
      READ(2,1000)
      DO 10 J=1,5
      READ(2,1002)(NVR(I,J),I=1,NP)
   10 CONTINUE
      READ(2,1000)
      READ(2,1003)(XV(I),I=1,NT)
 1003 FORMAT(8(2X,F7.4))
      READ(2,1000)
      READ(2,1003)(YV(I),I=1,NT)
      READ(2,1000)
      READ(2,1003)(ZV(I),I=1,NT)
      READ(2,1000)
      READ(2,1003)XC,YC,ZC
      RETURN
      END
/FORT
C
C
C
C ESTABLISH SURFACE NORMAL FOR EACH PLATE SURFACE
C SURFACE NORMALS DIRECTED TOWARD COCKPIT INTERIOR
      SUBROUTINE NORML
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/VERT/XV(100),YV(100),ZV(100),NVR(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/PILOT/XC,YC,ZC
      NVW=0
      CALL GETDEV(LUN,'FINKSH',10)
      CALL SEEK(LUN,0)
      WRITE(LUN)NVW
      WRITE(LUN)NP,NC,NB
      DO 10 I=1,NP
      NK=NV(I)
      WRITE(LUN)NK
      DO 5 K=1,NK
      KV=NVR(I,K)
      PXV(I,K)=XV(KV)
      PYV(I,K)=YV(KV)
      PZV(I,K)=ZV(KV)
      WRITE(LUN)PXV(I,K),PYV(I,K),PZV(I,K)
    5 CONTINUE
      K=2
    7 A1=PXV(I,K)-PXV(I,1)
      A2=PXV(I,K+1)-PXV(I,1)
      B1=PYV(I,K)-PYV(I,1)
```

```
      P2=PYV(I,K+1)-PYV(I,1)
      C1=PZV(I,K)-PZV(I,1)
      C2=PZV(I,K+1)-PZV(I,1)
      P1=SQRT(A1**2+B1**2+C1**2)
      P2=SQRT(A2**2+B2**2+C2**2)
      A=(A1*A2+B1*B2+C1*C2)/(P1*P2)
      IF(ABS(A).LT.1.) GO TO 9
      K=K+1
      IF(K.EQ.NK) GO TO 10
      GO TO 7
    9 AN=ACOS(A)
      R=1./(P1*P2*SIN(AN))
      AXN(I)=-(B1*C2-C1*B2)*R
      AYN(I)=+(A1*C2-C1*A2)*R
      AZN(I)=-(A1*B2-A2*B1)*R
      WRITE(LUN)AXN(I),AYN(I),AZN(I)
   10 CONTINUE
      WRITE(LUN)XC,YC,ZC
      NVW=NEXREC(LUN)
      CALL SEEK(LUN,0)
      WRITE(LUN)NVW
      CALL DEVT(LUN,0,0)
      RETURN
      END
/FORT
C
C
C READ FILE FOR CANOPY DATA
      SUBROUTINE READF
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/PILOT/XC,YC,ZC
      CALL GETDEV(LUN,'FINKSH',10)
      CALL SEEK(LUN,0)
      READ(LUN)NVW
      READ(LUN)NP,NC,NB
      DO 10 I=1,NP
      READ(LUN)NK
      NV(I)=NK
      DO 5 K=1,NK
      READ(LUN)PXV(I,K),PYV(I,K),PZV(I,K)
    5 CONTINUE
      READ(LUN)AXN(I),AYN(I),AZN(I)
   10 CONTINUE
      READ(LUN)XC,YC,ZC
      CALL DEVT(LUN,0,0)
      RETURN
      END
/FORT
C
C
C PRINTOUT OF CANOPY DATA FOR REVIEW
      SUBROUTINE TELTY
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/PILOT/XC,YC,ZC
      WRITE(1,998)
  998 FORMAT(2X,'PRINTER, TURN ON')
      PAUSE
      WRITE(3,1000)NT,NP,NC,NB
```

```
 1000 FORMAT(4(2X,I2))
      DO 10 I=1,NP
      KN=NV(I)
      DO 5 K=1,KN
      WRITE(3,1001)PXV(I,K),PYV(I,K),PZV(I,K)
 1001 FORMAT(2X,3(F6.2,2X))
    5 CONTINUE
      WRITE(3,1002)AXN(I),AYN(I),AZN(I)
 1002 FORMAT(3(2X,F10.4))
   10 CONTINUE
      WRITE(3,1002)XC,YC,ZC
      RETURN
      END
/FORT
C
C
C DETERMINES IF RAY STRIKES CONVEX SURFACE
      SUBROUTINE INTEC(ISK,IS,XR,YR,ZR)
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/LINE/AS,BS,CS,XS,YS,ZS
      CK=AXN(IS)*AS+AYN(IS)*BS+AZN(IS)*CS
      IF(CK.GE.0.) RETURN
C RAY STRIKES SURFACE IN OUTWARD DIRECTION
      I=1
      IF(XS.EQ.PXV(IS,I).AND.YS.EQ.PYV(IS,I).AND.ZS.EQ.PZV(IS,I))I=I+1
      S=(AXN(IS)*(PXV(IS,I)-XS)+AYN(IS)*(PYV(IS,I)-YS)+AZN(IS)*(PZV(IS,I
     C)-ZS))/CK
      XR=AS*S+XS
      YR=BS*S+YS
      ZR=CS*S+ZS
      IN=NV(IS)
      DO 10 I=1,IN
      IC=I+1
      IF(I.EQ.IN) IC=1
      A1=PXV(IS,I)-XS
      B1=PYV(IS,I)-YS
      C1=PZV(IS,I)-ZS
      A2=PXV(IS,IC)-XS
      B2=PYV(IS,IC)-YS
      C2=PZV(IS,IC)-ZS
      P1=XR-XS
      P2=YR-YS
      P3=ZR-ZS
      Q=P1*(B1*C2-P2*C1)-P2*(A1*C2-C1*A2)+P3*(A1*P2-B1*A2)
      IF(Q.LT.0.) RETURN
C RAY STRIKES SURFACE ON ENCLOSED SIDE OF SURFACE EDGE
   10 CONTINUE
C RAY STRIKES ENCLOSED SURFACE
      ISK=1
      RETURN
      END
/FORT
C
C
C COMPUTES INCIDENCE ANGLE, REFLECTANCE, AND TRANSMITTANCE
C NATURAL LIGHT, ADDITION OF POLARIZATION COMPONENTS IGNORED
      SUBROUTINE COMP(ANG,IS,RT,TT)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/LINE/AS,BS,CS,XS,YS,ZS
```

```
C XN, INDEX OF REFRACTION, TX, INTERNAL TRANSMITTANCE
      DATA XN,TX/1.5,.92/
      A=-AS*AXN(IS)-BS*AYN(IS)-CS*AZN(IS)
      IF(ABS(A).GT.1.) A=1.
      ANG=ACOS(A)
      ANGP= ASIN(SIN(ANG)/XN)
      CA=COS(ANG)
      SA=SIN(ANG)
      S1=SQRT(XN**2-SA**2)
      RO=(((CA-S1)/(CA+S1))**2+((CA*(XN**2)-S1)/(CA*(XN**2)+S1))**2)/2.
      TO=(1.-RO)*CA/COS(ANGP)
      CA=COS(ANGP)
      SA=SIN(ANGP)
      S1=SQRT(XN**2-SA**2)
      RI=(((CA-S1)/(CA+S1))**2+((CA*(XN**2)-S1)/(CA*(XN**2)+S1))**2)/2.
      TI=(1.-RI)*CA/COS(ANG)
      TT=TO*TI*TX/(1.-(RI*TX)**2)
      RT=RO+RI*TT
      RETURN
      END
/FORT
C
C
C COMPUTES ARCSIN
      FUNCTION ASIN(X)
      Y=X
      AX=ABS(Y)
      IF(AX.GE.1.0) GO TO 4
      AC=ATAN(Y/SQRT(1.-Y*Y))
      ASIN=AC
      RETURN
    4 IF(AX.GE.1.0001) GO TO 10
      ASIN=1.5707
      IF(Y.LT.0.)ASIN=-ASIN
      RETURN
   10 WRITE(1,11)X
      STOP
   11 FORMAT(8HERROR * ,23HARCSIN ARGUMENT .GT.1. ,6HARG = ,E16.8)
      END
/FORT
C
C
C BRL ARCCOS ROUTINE
C REF. GLENN BECK
      FUNCTION ACOS(X)
      Y=X
      AX=ABS(Y)
      IF(AX.GE.1.0)GOTO 4
      IF(Y.EQ.0.)GOTO 3
      AC=ATAN(SQRT(1.0-Y*Y)/Y)
    1 IF(Y.LT.0.)GOTO 2
      ACOS=AC
      RETURN
    2 ACOS=AC+3.1415926
      RETURN
    3 ACOS=1.5707963
      RETURN
    4 IF(AX.GE.1.00001)GOTO 10
      AC=0.
      GOTO 1
```

```
   10 WRITE(1,11)X
      STOP
11    FORMAT(8HERROR * ,23HARCCOS ARGUMENT .GT.1. ,6HARG = ,E16.8)
      END
/FORT
C
C
C DRAWS GRAPHIC PICTURE OF CANOPY IN 3 FOLD LAYOUT
      SUBROUTINE DRWCN
      COMMON/FORG/LFG
      COMMON/CAREA/ICFIL(1CCCO)
      COMMON/CAN/NT,NP,NC,NB,NV(1CO),PXV(1CO,5),PYV(1CC,5),PZV(100,5)
      COMMON/NORM/AXN(1CC),AYN(1CC),AZN(1CO)
      COMMON/PILOT/XO,YC,ZO
      DATA SX/6.0235/
      WRITE(1,998)
  998 FORMAT(2X,'CONSOLE NO 1, TURN ON')
      PAUSE
      CALL GIN(6CCC)
      IX=(YC-5C.)*SX
      IY=(ZO-6C.)*SX
      CALL GBEG(1,IX,IY)
      CALL EMARK
      IY=(XO+9C.)*SX
      CALL GCPY(2,1,IX,IY)
      IX=(ZC+5C.)*SX
      CALL GBEG(3,IX,IY)
      CALL ETIC
      NE=3
      NS=NB+1
      DO 10 I=NS,NP
C ENTITY IS CANOPY SURFACE--FENCE OR TRANSPARANT
      CS=-AXN(I)
      CF=-AYN(I)
      CT=AZN(I)
      IX=(PYV(I,1)-5C.)*SX
      IF((I.LE.NC.AND.CS.LT.0.).OR.(I.GT.NC.AND.CS.GT.C.)) GO TO 2
C SURFACE FACES VIEWER FROM SIDE VIEW
      IY=(PZV(I,1)-6C.)*SX
      NE=NE+1
      CALL GBEG(NE,IX,IY)
      IF(I.GT.NB.AND.I.LE.NC)CALL GPUT(3,130,1,2)
      CALL LINS(I)
    2 CONTINUE
      IY=(PXV(I,1)+9C.)*SX
      IF((I.LE.NC.AND.CT.LT.0.).OR.(I.GT.NC.AND.CT.GT.C.)) GO TO 3
C SURFACE FACES VIEWER FROM TOP VIEW
      NE=NE+1
      CALL GBEG(NE,IX,IY)
      IF(I.GT.NB.AND.I.LE.NC)CALL GPUT(3,130,1,2)
      CALL LINT(I)
    3 CONTINUE
      IF((I.LE.NC.AND.CF.LT.0.).OR.(I.GT.NC.AND.CF.GT.C.)) GO TO 10
C SURFACE FACES VIEWER FROM FRONT VIEW
      IX=(PZV(I,1)+5C.)*SX
      NE=NE+1
      CALL GBEG(NE,IX,IY)
      IF(I.GT.NB.AND.I.LE.NC)CALL GPUT(3,130,1,2)
      CALL LINF(I)
   10 CONTINUE
```

```
      DO 20 I=1,NE
      CALL GEON(I)
   20 CONTINUE
      CALL GSTART
      RETURN
      END
/FORT
C
C
C MARK EYE POSITION IN CANOPY
      SUBROUTINE EMARK
      CALL GPUT(6,50,-10,-5)
      CALL GPUT(7,70,0,10)
      CALL GPUT(8,50,10,-5)
      CALL GPUT(9,70,-7,-10)
      CALL GPUT(10,50,0,20)
      RETURN
      END
/FORT
C
C
      SUBROUTINE ETIC
      CALL GPUT(6,70,0,-5)
      CALL GPUT(7,50,0,10)
      CALL GPUT(8,70,-5,-5)
      CALL GPUT(9,50,10,0)
      RETURN
      END
/FORT
C
C
      SUBROUTINE LINS(I)
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      DATA SX/6.0235/
      CALL GPUT(3,130,2,0)
      NK=NV(I)
      DO 10 K=1,NK
      K1=K+1
      IF(K.EQ.NK) K1=1
      IX=(PYV(I,K1)-PYV(I,K))*SX
      IY=(PZV(I,K1)-PZV(I,K))*SX
      CALL GPUT(K+5,53,IX,IY)
   10 CONTINUE
      RETURN
      END
/FORT
C
C
      SUBROUTINE LINF(I)
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      DATA SX/6.0235/
      CALL GPUT(3,130,2,0)
      NK=NV(I)
      DO 10 K=1,NK
      K1=K+1
      IF(K.EQ.NK) K1=1
      IX=(PZV(I,K1)-PZV(I,K))*SX
      IY=(PXV(I,K1)-PXV(I,K))*SX
      CALL GPUT(K+5,53,IX,IY)
   10 CONTINUE
```

```
      RETURN
      END
/FORT
C
C
      SUBROUTINE LINT(I)
      COMMON/CAN/NT,NP,NC,NR,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      DATA SX/6.0235/
      CALL GPLT(3,130,2,0)
      NK=NV(I)
      DO 10 K=1,NK
      K1=K+1
      IF(K.EQ.NK) K1=1
      IX=(PYV(I,K1)-PYV(I,K))*SX
      IY=(PXV(I,K1)-PXV(I,K))*SX
      CALL GPLT(K+5,53,IX,IY)
   10 CONTINUE
      RETURN
      END
/FORT
C
C
C DRAWS PERPECTIVE OF CANOPY AND ENTRY AND REFLECTION POINTS PILOT EYE
C POSITION
      SUBROUTINE PERP(IES)
      COMMON/FORG/LFG
      COMMON/GAREA/IDFIL(10000)
      COMMON/PER/ND,NS(100),PXS(100,10),PYS(100,10),PZS(100,10)
      DATA AX,PX,CX/10.,5.,512./
      CALL TRSDC
      CALL GETDEV(LUN,'FINKSH',10)
      CALL GIN(10000)
      CALL EFIX(XX,YY,ZZ)
      CALL PLAC(YY,ZZ,PY,PZ)
      IXE=XX*AX*CX/(PY*PX)+CX
      IYE=PZ*AX*CX/(PY*PX)+CX
      CALL GBEG(1,IXE,IYE)
      CALL ETIC
      CALL GECN(1)
      NE=2
      IK=10
      DO 10 I=1,ND
      NK=NS(I)
      IF(IK.GT.6) CALL GBEG(NE,IXE,IYE)
      IK=6
      DO 5 K=1,NK
      K1=K+1
      IF(K.EQ.NK) K1=1
C CHECK EDGES FOR HIDDEN LINES
      CALL CLIPL(IK,I,K,K1)
    5 CONTINUE
      IF(IK.EQ.6) GO TO 10
      CALL GECN(NE)
      NE=NE+1
   10 CONTINUE
      IF(IES.EQ.0) GO TO 21
      CALL GBEG(NE+1,IXE,IYE)
      CALL GPLT(5,1760,0,.FALSE.)
      IF(IES.EQ.2) CALL GPLT(5,1750,0,0)
      CALL SEEK(LUN,0)
```

```
      READ(LUN)NVWC
      CALL SEEK(LUN,NVWC)
      READ(LUN)NVW
      IK=6
      DO 20 I=1,NVW
      READ(LUN)A1V,A2V,ASP,BSP,CSP,IP,XP,YP,ZP,AIP,BIP,CIP,ARP,BRP,CRP,
     CR,IS,XR,YR,ZR,AI,BI,CI,AR,BR,CR,T
      IF(IES.EQ.1) GO TO 30
      GO TO 40
   20 CONTINUE
      CALL GEON(NE+1)
   21 CALL GSTART
      CALL DEVT(LUN,C,0)
      RETURN
C ENTRENCE POINTS FOR EXTERNAL RAYS
   30 CONTINUE
      CALL PLAC(YR,ZR,PY,PZ)
      IF(PY.LE.C.) GO TO 20
      XX=XR*AX/(PY*BX)
      ZZ=PZ*AX/(PY*BX)
      IF(ABS(XX).GT.1..OR.ABS(ZZ).GT.1.) GO TO 20
      IX=XX*CX+CX
      IY=ZZ*CX+CX
      CALL GPUT(IK,43,IX,IY)
      IK=IK+1
      GO TO 20
C PRIMARY REFLECTION POINTS
   40 CONTINUE
      CALL PLAC(YP,ZP,PY,PZ)
      IF(PY.LE.C.) GO TO 20
      XX=XP*AX/(PY*BX)
      ZZ=PZ*AX/(PY*BX)
      IF(ABS(XX).GT.1..OR.ABS(ZZ).GT.1.) GO TO 20
      IX=XX*CX+CX
      IY=ZZ*CX+CX
      CALL GPUT(IK,100,IX-4,0)
      CALL GPUT(IK+1,110,IY-4,0)
      IC=-ALOG10(T)
      IC=IC+176
      CALL GPUT(IK+2,90,IC,0)
      IK=IK+3
      GO TO 20
      END
/FORT
C
C
C COMPUTES EYE FIXATION CENTER FOR PERPECTIVE DRAWING
      SUBROUTINE EFIX(XR,YR,ZR)
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/PILOT/XC,YC,ZC
      COMMON/LINE/AS,BS,CS,XS,YS,ZS
      DATA AN,PI/5.,3.14159/
      ANN=AN*PI/180.
      XS=XC
      YS=YC
      ZS=ZC
      AS=0.
      BS=-COS(ANN)
      CS=SIN(ANN)
      DO 20 IS=1,NP
```

```
      ISK=0
      CALL INTEC(ISK,IS,XR,YR,ZR)
      IF(ISK.GT.0) RETURN
   20 CONTINUE
      RETURN
      END
/FORT
C
C
C CONVERTS OBJECT SPACE COORDINATES INTO EYE SPACE COORDINATES
C PILOT EYE DIRECTED IN Y-Z PLANE OF AIRCRAFT COORDINATES AND 5-DEGREES
C ABOVE NEGATIVE Y-AXIS
      SUBROUTINE PLAC(PYV,PZV,PY,PZ)
      COMMON/PILOT/XC,YC,ZC
      DATA AN/5./
      DATA PI/3.14159/
      AN1=AN*PI/180.
      PY=-(PYV-YC)*COS(AN1)-(PZV-ZC)*SIN(AN1)
      PZ=-(PYV-YC)*SIN(AN1)+(PZV-ZC)*COS(AN1)
      RETURN
      END
/FORT
C
C
C CONVERTS OBJECT COORDINATES TO DISPLAY COORDINATES , REMOVES LINES BEHIND
C VIEWER, SURFACES OUTSIDE VIEWING BOX, SURFACES NOT FACING VIEWER
      SUBROUTINE TRSOC
      COMMON/CAN/NT,NP,NC,NB,NV(100),PXV(100,5),PYV(100,5),PZV(100,5)
      COMMON/NORM/AXN(100),AYN(100),AZN(100)
      COMMON/PILOT/XC,YC,ZC
      COMMON/LINE/X1,Y1,Z1,X2,Y2,Z2
      COMMON/PER/ND,NS(100),PXS(100,10),PYS(100,10),PZS(100,10)
      DIMENSION XS(100,5),YS(100,5),ZS(100,5)
      DATA A,B/10.,5./
      NST=NB+1
      ND=1
      DO 10 I=NST,NP
      C=AXN(I)*(PXV(I,1)-XC)+AYN(I)*(PYV(I,1)-YC)+AZN(I)*(PZV(I,1)-ZC)
      IF(C.GT.0.) GO TO 10
C SURFACE FACES VIEWER
      KN=NV(I)
      DO 2 K=1,KN
      XS(ND,K)=PXV(I,K)
      CALL PLAC(PYV(I,K),PZV(I,K),YS(ND,K),ZS(ND,K))
C CONVERTED TO VIEWER COORDINATES
    2 CONTINUE
      KC=0
      IXP=0
      IXN=0
      IZP=0
      IZN=0
      DO 5 K=1,KN
      K1=K+1
      IF(K.EQ.KN) K1=1
      IF(YS(ND,K).LT.0..AND.YS(ND,K1).LT.0.) GO TO 5
      IF(YS(ND,K).LT.0..OR.YS(ND,K1).LT.0.) GO TO 3
      X1=XS(ND,K)
      Y1=YS(ND,K)
      Z1=ZS(ND,K)
      C=1
```

```
      GO TO 4
    3 CONTINUE
      CALL INTEP(XS(ND,K),YS(ND,K),ZS(ND,K),XS(ND,K1),YS(ND,K1),ZS(ND,K1
     C),C)
C SURFACE EDGE EXTENDING BEHIND VIEWER TRIMMED
    4 KD=KD+1
      PXS(ND,KD)=X1*A/(Y1*B)
      PYS(ND,KD)=-1./Y1
      PZS(ND,KD)=Z1*A/(Y1*B)
      CALL TEST(PXS(ND,KD),PZS(ND,KD),IXP,IXN,IZP,IZN)
C POSITIVE POINT OR CROSSING LINE FROM BEHIND
      IF(C.GT.0.) GO TO 5
      KD=KD+1
      PXS(ND,KD)=X2*A/(Y2*B)
      PYS(ND,KD)=-1./Y2
      PZS(ND,KD)=Z2*A/(Y2*B)
C LEAD POINT FOR LINE CROSSING FROM IN FRONT
      CALL TEST(PXS(ND,KD),PZS(ND,KD),IXP,IXN,IZP,IZN)
C VERTEX CONVERTED TO DISPLAY VIEWBOX COORDINATES
    5 CONTINUE
      IF(KD.EQ.0) GO TO 10
C SURFACE NOT BEHIND VIEWER
      IF(IXP.GE.KD.OR.IXN.GE.KD.OR.IZP.GE.KD.OR.IZN.GE.KD) GO TO 10
C SURFACE PARTIALLY OR COMPLETELY WITHIN VIEW BOX
      NS(ND)=KD
      ND=ND+1
   10 CONTINUE
      ND=ND-1
      RETURN
      END
/FORT
C
C
C TEST SURFACE VERTEX FOR POSITION OUTSIDE VIEWBOX
      SUBROUTINE TEST(X,Z,IXP,IXN,IZP,IZN)
      IF(X.GT.1.) IXP=IXP+1
      IF(X.LT.-1.)IXN=IXN+1
      IF(Z.GT.1.) IZP=IZP+1
      IF(Z.LT.-1.)IZN=IZN+1
      RETURN
      END
/FORT
C
C
C CALCULATES INTERSECTION POINT FOR LINE WITH PLANE NORMAL TO Y-AXIS
      SUBROUTINE INTEP(X1,Y1,Z1,X2,Y2,Z2,R)
      COMMON/LINE/XS,YS,ZS,XF,YF,ZF
      DATA YO/1./
      R=SQRT((X1-X2)**2+(Y1-Y2)**2+(Z1-Z2)**2)
      A=(X2-X1)/R
      B=(Y2-Y1)/R
      C=(Z2-Z1)/R
      IF(B.EQ.0.) RETURN
C EDGE NOT PARALLEL TO PLANE
      YI=YO
      RI=(YI-Y1)/B
      XI=X1+A*RI
      ZI=Z1+C*RI
      IF(B.LT.0.) GO TO 3
C LINE ORIGINATES BEHIND VIEWER
```

```
      XS=X1
      YS=Y1
      ZS=Z1
      XF=X2
      YF=Y2
      ZF=Z2
      RETURN
    3 CONTINUE
C LINE ORIGINATES IN FRONT OF VIEWER
      XS=X1
      YS=Y1
      ZS=Z1
      XF=X1
      YF=Y1
      ZF=Z1
      RETURN
      END
/FORT
C
C
C DETERMINES VISIBLE PORTION OF EDGE BY REMOVING HIDDEN LINE
C ALL CONSTRAINT SURFACES CONVEX
      SUBROUTINE CLIPL(IK,IV,KV,KV1)
      COMMON/LINE/AV,BV,CV,XV,YV,ZV
      COMMON/PER/ND,NS(100),PXS(100,10),PYS(100,10),PZS(100,10)
      COMMON/DRAW/NSC,RV,ROV(10),RFV(10)
      DATA CX/512./
      XV=PXS(IV,KV)
      YV=PYS(IV,KV)
      ZV=PZS(IV,KV)
      RV=SQRT((PXS(IV,KV1)-XV)**2+(PYS(IV,KV1)-YV)**2+(PZS(IV,KV1)-ZV)**
     C2)
      AV=(PXS(IV,KV1)-XV)/RV
      BV=(PYS(IV,KV1)-YV)/RV
      CV=(PZS(IV,KV1)-ZV)/RV
      RO=0.
      RF=RV
      CALL CHKS(-1.,0.,0.,RO,RF)
      CALL CHKS(1.,0.,0.,RO,RF)
      CALL CHKS(0.,0.,1.,RO,RF)
      CALL CHKS(0.,0.,-1.,RO,RF)
      IF(RF.LT.RO) RETURN
C EDGE CHECKED AGAINST VIEWBOX
      NSC=1
      ROV(1)=RO
      RFV(1)=RF
      DO 10 I=1,ND
      IF(IV.EQ.I) GO TO 10
C FIND APPARANT INTERSECTION POINT OF EDGES
      ISC=0
      KN=NS(I)
      DO 5 K=1,KN
      K1=K+1
      IF(K.EQ.KN) K1=1
      XE=PXS(I,K)
      YE=PYS(I,K)
      ZE=PZS(I,K)
      RE=SQRT((PXS(I,K1)-XE)**2+(PYS(I,K1)-YE)**2+(PZS(I,K1)-ZE)**2)
      AE=(PXS(I,K1)-XE)/RE
      BE=(PYS(I,K1)-YE)/RE
```

```
      CE=(PZS(I,K1)-ZE)/RE
      Q=AE*CV-AV*CE
      IF(Q.EQ.0.) GO TO 5
C EDGES NOT PARALLEL
      RIV=(CE*(XV-XE)-AE*(ZV-ZE))/Q
      IF(AE.EQ.0.) GO TO 2
      RIE=(XV-XE+AV*RIV)/AE
      GO TO 3
    2 CONTINUE
      RIE=(ZV-ZE+CV*RIV)/CE
    3 CONTINUE
      YIE=YE+BE*RIE
      YIV=YV+BV*RIV
      IF(YIE.GT.(YIV-.001)) GO TO 5
C CONSTRAINT EDGE IN FRONT OF TEST EDGE
      IF(RIE.GT.0..AND.RIE.LT.RE) GO TO 7
    5 CONTINUE
      IF(ISC.LT.2) GO TO 10
      IF(RO.LT.0..AND.RF.GT.RV) RETURN
C TEST EDGE NOT HIDDEN BEHIND CONSTRAINT SURFACE
      IF(RO.GT.RV.OR.RF.LT.0.) GO TO 10
C TEST EDGE PARTIALLY BLOCKED BY CONSTRAINT SURFACE
      CALL CKLIN(RO,RF)
      IF(NSC.EQ.0) RETURN
      GO TO 10
    7 CONTINUE
C CONSTRAINT EDGE LOCATED
      ISC=ISC+1
      IF(ISC.EQ.1) R1=RIV
      IF(ISC.EQ.2) CALL CRDLN(R1,RIV,RO,RF)
      GO TO 5
   10 CONTINUE
C VISIBLE PORTION OF EDGE REMAINS
      DO 20 I=1,NSC
      RO=ROV(I)
      RF=RFV(I)
      XS=XV+RO*AV
      ZS=ZV+RO*CV
      XF=XV+RF*AV
      ZF=ZV+RF*CV
      IXS=(XS+1.)*CX
      IZS=(ZS+1.)*CX
      IXF=(XF+1.)*CX
      IZF=(ZF+1.)*CX
      CALL GPUT(IK,100,IXS,0)
      CALL GPUT(IK+1,110,IZS,0)
      IDX=IXF-IXS
      IDZ=IZF-IZS
      CALL GPUT(IK+2,53,IDX,IDZ)
      IK=IK+3
   20 CONTINUE
      RETURN
      END
/FORT
C
C
C CHECKS EDGE AGAINST VIEWING BOX SIDE
      SUBROUTINE CHKS(AN,BN,CN,RO,RF)
      COMMON/LINE/AV,BV,CV,XV,YV,ZV
      XN=-AN
```

```
      YN=-BN
      ZN=-CN
      C=AV*AN+BV*BN+CV*CN
      IF(C.EQ.C.) GO TO 5
      RI=((XN-XV)*AN+(YN-YV)*BN+(ZN-ZV)*CN)/Q
      IF(C.LE.C.) GO TO 3
C EDGE DIRECTED INTO VIEWBOX FROM OUTSIDE
      IF(RI.GT.RO) RO=RI
      RETURN
    3 CONTINUE
C EDGE DIRECTED OUT OF VIEWBOX FROM INSIDE
      IF(RI.LT.RF) RF=RI
      RETURN
    5 CONTINUE
C EDGE PARALLEL TO SIDE
      IF(ABS(XV).GT.1..OR.ABS(ZV).GT.1.) RF=RO-1.
      RETURN
      END
/FORT
C
C
C ORDER EDGE INTERSECTION POINTS ACCORDING TO LOW AND HIGH VALUES
      SUBROUTINE ORDLN(R1,R2,RO,RF)
      RO=R1
      RF=R2
      IF(R1.LT.R2) RETURN
      RO=R2
      RF=R1
      RETURN
      END
/FORT
C
C
C CHECKED PARTIALLY BLOCKED LINE FOR VISIBLE SEGMENTS
      SUBROUTINE CKLIN(R1,R2)
      COMMON/DRAW/NSC,RV,RCV(10),RFV(10)
      DIMENSION RC(10),RF(10)
      IF(R1.GT.C..AND.R2.LT.RV) GO TO 20
C VERTEX OF TEST EDGE BEHIND CONSTRAINT SURFACE
      IF(R2.GT.RV) GO TO 5
C LOW END OF EDGE HIDDEN
      RVC=R2
      RVF=RV
      GO TO 10
C HIGH END OF EDGE HIDDEN
    5 CONTINUE
      RVC=0.
      RVF=R1
   10 CONTINUE
      DO 12 I=1,NSC
      RC(I)=RCV(I)
      RF(I)=RFV(I)
      IF(RFV(I).LE.RVO) RF(I)=-1.
      IF(RCV(I).LT.RVC) RC(I)=RVC
      IF(RFV(I).GT.RVF) RF(I)=RVF
      IF(RCV(I).GT.RVF) RF(I)=-1.
   12 CONTINUE
      GO TO 30
C CONSTRAINT SURFACE SEPARATES EDGE INTO TWO VISIBLE ENDS
   20 CONTINUE
```

```
      RVC=R1
      RVF=R2
      K=C
      DO 22 I=1,NSC
      K=K+1
      RO(K)=RCV(I)
      RF(K)=RFV(I)
      IF(RFV(I).GT.RVO.AND.RFV(I).LT.RVF) RF(K)=RVO
      IF(RCV(I).GT.RVC.AND.RFV(I).LT.RVF) RF(K)=-1.
      IF(RFV(I).GT.RVF.AND.ROV(I).LT.RVO) GO TO 27
      IF(RFV(I).GT.RVF.AND.ROV(I).GT.RVO.AND.ROV(I).LT.RVF) RO(K)=RVF
   22 CONTINUE
      NSC=K
      GO TO 30
   27 CONTINUE
      RF(K)=RVC
      K=K+1
      RO(K)=RVF
      RF(K)=RFV(I)
      GO TO 22
C ARRANGE REMAINING VISIBLE SEGMENTS
   30 CONTINUE
      NK=C
      DO 32 I=1,NSC
      IF(RF(I).LE.C.) GO TO 32
      NK=NK+1
      ROV(NK)=RO(I)
      RFV(NK)=RF(I)
   32 CONTINUE
      NSC=NK
      RETURN
      END


C FLAT PLATE CANOPY DATA
C NO. VERTICES, AND SURFACES--BODY, CONSTRAINT, AND TRANSPARENT
  5C  30  23   3
C NO. VERTICES PER SURFACE
   4   3   3   4   4   4   4   4   4   4   4   4   4   4   4   4   4   4
   4   4   4   4   4   4   4   4   5   5   5   5
C  SURFACE VERTICES IN ROTATIONAL ORDER-ALL 1ST VERTICES, ALL 2ND, ETC.
   1   1   2  1C   7  13  14  17  18  22  22  24  23  25  29  30  33  34
   5  37  39  41  43  49  47  45  38  39  40  39
   2   5   4  12   8  14  13  18  17  24  21  23  21  26  30  29  34  33
   6  38  4C  42  44  5C  48  46   6  47  48  41
   3   3   6  11  10  15  16  19  2C  25  23  26  28  28  31  32  35  36
  38  40  42  44  46   6  5C  48  5C  49  46  43
   5   0   C   9   9  16  15  2C  19  27  24  25  26  27  32  31  36  35
  37  39  41  43  45   5  49  47  48   5  44  45
   0   0   C   C   0   C   C   C   C   0   0   0   0   0   0   0   0   0
   C   0   C   C   C   C   C   C  4C  37  42  47
C X-POSITION ALL VERTICES
    7.23    -7.23    14.47   -14.47    13.C79  -13.079     5.       -5.
     5.      -5.       5.      -5.      16.      16.      16.       16.
  -16.     -16.     -16.     -16.     -14.      14.      -5.5       5.5
    5.5     -5.5     14.     -14.      16.      16.      16.       16.
  -16.     -16.     -16.     -16.      14.47   -14.47    16.944  -16.944
   16.944  -16.944   16.039  -16.C39   14.366  -14.366   13.273  -13.273
   12.C06  -12.C06     .        .         .       .         .        .
C Y-POSITION ALL VERTICES
```

```
   25.06     25.06     61.25     61.25     56.769    56.769    67.769    67.769
   67.769    67.769    67.769    67.769    85.928    85.928    99.928    99.928
   85.925    85.925    99.928    99.928    99.928    99.928    99.928    99.928
  113.928   113.928   113.928   113.928   133.7     133.7     137.7     137.7
  133.7     133.7     137.7     137.7      59.06     59.06     99.928    99.928
  154.602   154.602   153.609   153.609   152.257   152.257   108.032   108.032
   71.5      71.5        .         .         .         .         .         .
C Z-POSITION ALL VERTICES
   63.829    63.829    60.       60.       76.018    76.018    76.018    76.018
   86.018    86.018    76.018    76.018    71.717    77.717    77.717    71.717
   71.717    77.717    77.717    71.717    75.717    75.717    90.467    90.467
   92.467    92.467    75.717    75.717    80.3      84.6      84.6      80.3
   80.3      84.6      84.6      80.3      67.829    67.829    75.717    75.717
   86.271    86.271    93.726    93.726   105.96    105.96    105.96    105.96
   92.4      92.4        .         .         .         .         .         .
C PILOT-EYE POSITION
    0.      137.7      95.67       .         .         .         .         .
```