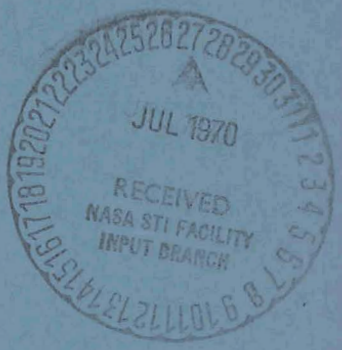
N 70-32669

NASA TECHNICAL MEMORANDUM

NASA TM X-2036

NASA TM X-2036



CASE FILE
COPY

FORTRAN SUBROUTINE FOR ROTATION OF THREE-DIMENSIONAL LINE FIGURES

by Kenneth Paciulan

Electronics Research Center
Cambridge, Mass. 02139

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • JULY 1970

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM X-2036		

4. Title and Subtitle	5. Report Date
FORTRAN Subroutine for Rotation of Three-Dimensional Line Figures	July 1970
	6. Performing Organization Code
7. Author(s)	8. Performing Organization Report No.
Kenneth Paciulan	C-122
9. Performing Organization Name and Address	10. Work Unit No.
Electronics Research Center Cambridge, Mass.	11. Contract or Grant No. 125-21-06-14-25
	13. Type of Report and Period Covered Technical Memorandum
12. Sponsoring Agency Name and Address	
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

In projecting a three-dimensional line figure onto a plane, it is sometimes necessary to choose more than one projection of the figure in order to obtain a clear idea of what the projection represents. This can be done by rotating the object and projecting it onto the original plane. The computations involved are explained, and a FORTRAN subroutine to carry out these computations is described in detail.

17. Key Words	18. Distribution Statement
Three-Dimensional Line Figures Plane FORTRAN Subroutine Computer Graphics	Unclassified-Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	13	$3.00

*For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151

FORTRAN SUBROUTINE FOR ROTATION OF THREE-DIMENSIONAL LINE FIGURES

By Kenneth Paciulan

SUMMARY

In projecting a three-dimensional line figure onto a plane, it is sometimes necessary to choose more than one projection of the figure in order to obtain a clear idea of what the projection represents. This can be done by rotating the object and projecting it onto the original plane. The computations involved are explained, and a FORTRAN subroutine to carry out these computations is described in detail.

INTRODUCTION

With the use of computer graphics becoming more and more common in the scientific community, certain problems have been encountered. The problem which is dealt with below concerns the use of two-dimensional projections of three-dimensional line figures. The projection of a line figure in space onto a plane can cause two lines to be superimposed if the plane determined by these two lines contains the viewing point, i.e., the point where the eye of an observer would be when viewing the object. This will result in the two lines being plotted by the computer as one line. This will lead to confusion in determining the exact location of each line. The solution to this problem is an easy one. All that is necessary is to view the figure from a point not in the plane of the two lines. This can be accomplished by rotating the entire line figure until a suitable vantage point is attained.

PROJECTION

With continued improvements in computer graphics equipment, more and more people are using projection of three-dimensional objects onto a plane for displaying figures. The projection of three-dimensional objects onto a plane has been a useful tool for scientists in a variety of fields. Having at hand a computer-drawn picture of some unconstructed hardware, one can adjust the parts and specifications without having to build prototypes which may turn out to be useless.

As the figures become more and more intricate, it sometimes becomes difficult to distinguish lines or parts. A line may be directly behind another line and thus disappear in a projection. The distance to that hidden line would be uncertain. In order to determine correctly the length of a line and its distance from a reference point, one would have to view the object from a different vantage point.

This ROTATE subroutine gives the viewer the option of choosing any vantage point from which to view the figure. This is done by rotating the figure around fixed axes. A continuous series of rotations can be performed at small angle increments to generate a movie in which the object is viewed from a constantly changing viewpoint. The figure would appear to be rotating in space while being viewed from a fixed point. The angle increments can be arbitrarily chosen to control the speed at which one revolution is made.

The rotation can occur around any one, two, or all three of the orthogonal coordinate axes. Picking appropriate angle increments for the three axes can give the object the appearance of spinning. The order of successive rotations, when more than one axis is used, will be discussed later.

This subroutine uses figures composed of straight line segments. There can be any number of segments in the figure. Curved lines can be formed by using a large number of small straight line segments. The rotation of these small lines will preserve the shape of the curve.

The coordinate system used in this subroutine is shown in Figure 1. The three angles required for this program give the rotations around the x-axis, the y-axis, and the z-axis

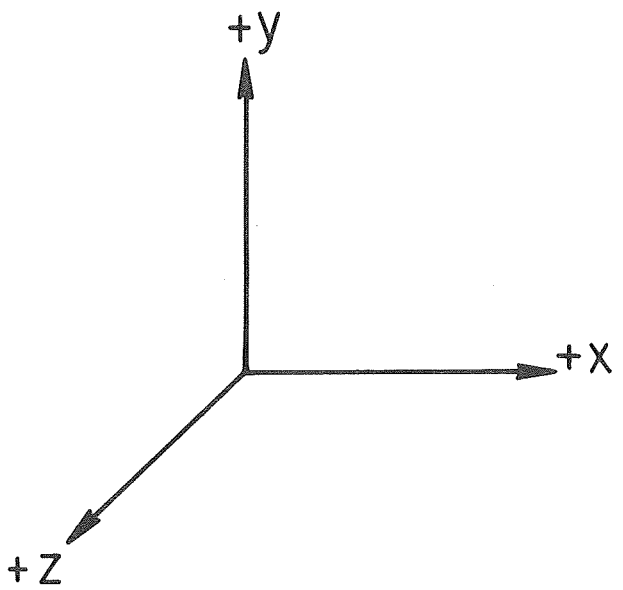


Figure 1.- Coordinate system used in this subroutine

respectively. The rotation around the x-axis is in the direction from the +y-axis to the +z-axis; for the y-axis, it is from the +z-axis to the +x-axis; for the z-axis, it is from the +x-axis to the +y-axis. These rotations are shown in Figure 2.

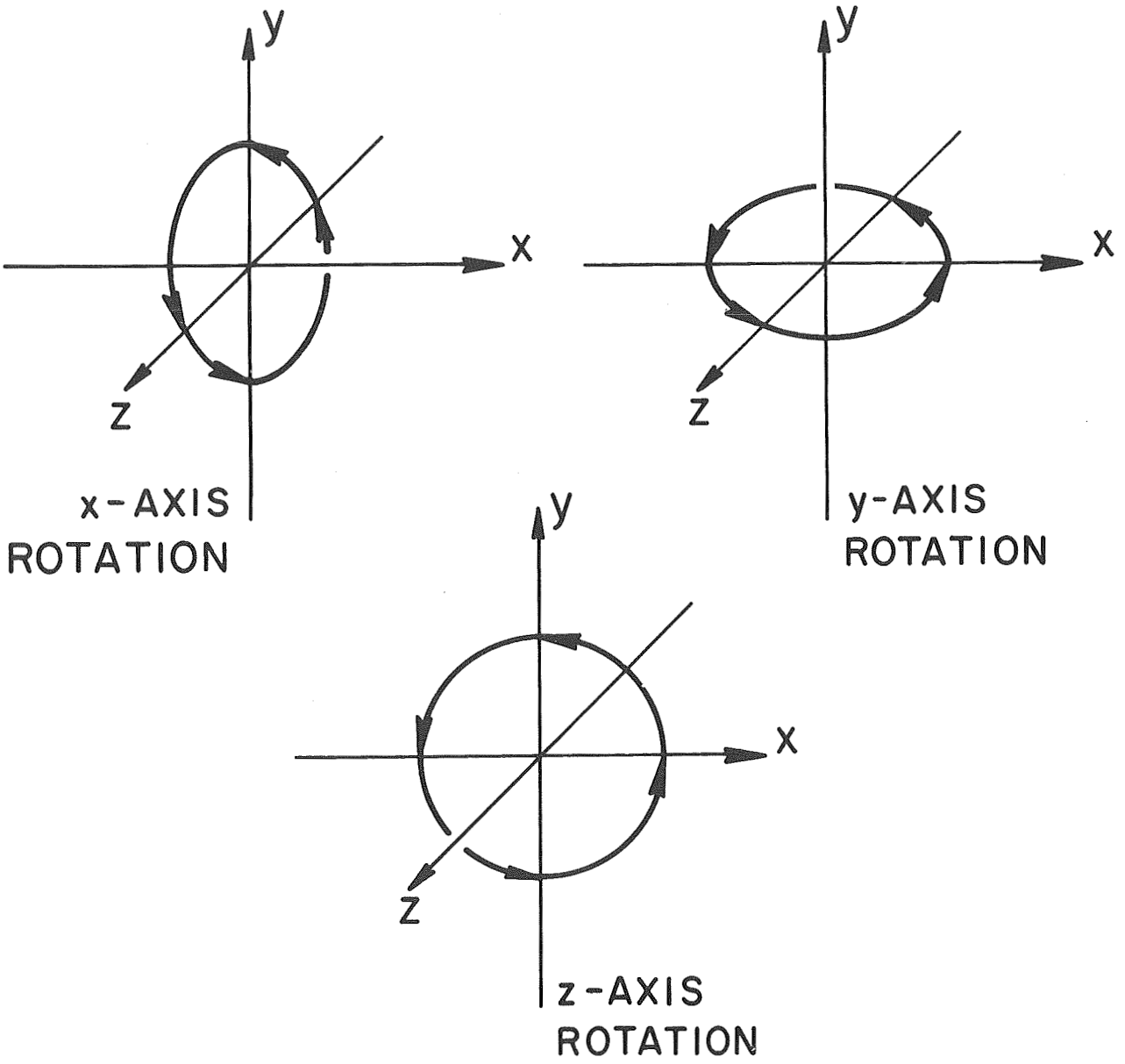


Figure 2.- Directions of rotation around the individual axes

This subroutine will rotate an object around the x-axis first, then the y-axis, and finally, the z-axis. This must be taken into consideration when rotating an object since, for example, an x-axis rotation followed by a y-axis rotation will generally give a different orientation from the one obtained by applying the y-axis rotation first. This can be shown by a simple example. Given the line segment from the origin, (0, 0, 0), to the point (1, 1, 1), a rotation of 90^{o} around the x-axis will yield the line segment from (0, 0, 0) to (1, -1, 1). Now a rotation of 90^{o} around the y-axis will give the line segment from (0, 0, 0) to (1, -1, -1). If one had applied the y-axis rotation first, the original line would have been first rotated to correspond to the line from (0, 0, 0) to (1, 1, -1). To then apply the x-axis rotation would have returned the line to its original position. Thus, care must be taken in choosing the appropriate angles. This can be avoided to some extent by making two separate calls to the subroutine if a different sequence of rotations is required.

The method by which the rotation takes place is based on simple trigonometric functions. The process is described for one representative rotation, that of z-axis rotation, and can be applied to the other two by appropriate renaming of the axes. The rotation of a line is performed by calculating the new positions of the endpoints from the old ones.

Since the rotation around the z-axis is from the +x-axis to the +y-axis, the names abscissa and ordinate will be applied to the axes such that the rotation will always be from the +-abscissa to the +-ordinate. In this case, the x-coordinates are the abscissas, and the y-coordinates are the ordinates. A point rotating around the axis will describe a circle lying in a plane which is perpendicular to the axis. Thus the coordinate along the axis will remain the same.

The endpoints of a line can be projected onto any plane perpendicular to the rotation axis without changing the values of the abscissas and the ordinates. In this case, the plane containing the x- and y-axes has been chosen for simplicity. A representative line will be taken so that one endpoint is at (0, 0, 0); then only one endpoint will be moved by any rotation, since (0, 0, 0) is a fixed point in space. If the line does not end at the origin, then each endpoint is rotated separately, and the new points are connected to form that line in its new position.

There are three possibilities for the location of the endpoint when it is projected onto the plane. The first is that the point will lie on the abscissa axis; the second is that it will be on the ordinate axis; the last case is that the point will lie between the two axes.

The first case is the one in which the ordinate equals zero (see Figure 3). From the diagram, it is easily seen that, since the distance from the point to the origin remains constant, the abscissa after rotation, x, is equal to the abscissa before rotation, x_o, times the cosine of the rotation angle, θ, i.e.,

$$x = x_o \cos \theta.$$

Again by simple trigonometry, the ordinate is merely the old abscissa times the sine of the rotation angle, i.e.,

$$y = x_o \sin \theta.$$

Now the point has its new rotated coordinates and can be joined to (0, 0, 0) as the rotated line. Thus the point at (x_o, y_o, z_o) is now at (x, y, z_o).

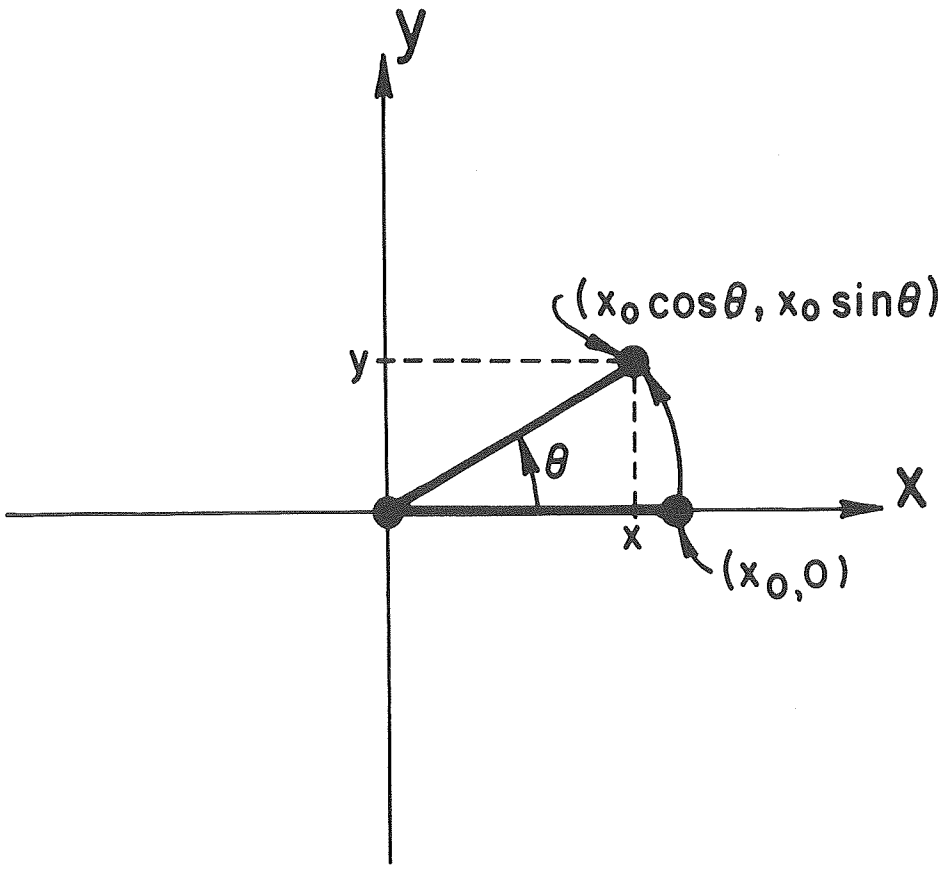


Figure 3.- Case one - the ordinate equaling zero

The second case has the abscissa equal to zero (see Figure 4). The process involved in this case is very much the same as that for the first case. Now

$$y = y_o \cos \theta,$$

and

$$x = -y_o \sin \theta.$$

These two cases take care of endpoints which have one of the x- or y-coordinates equal to zero.

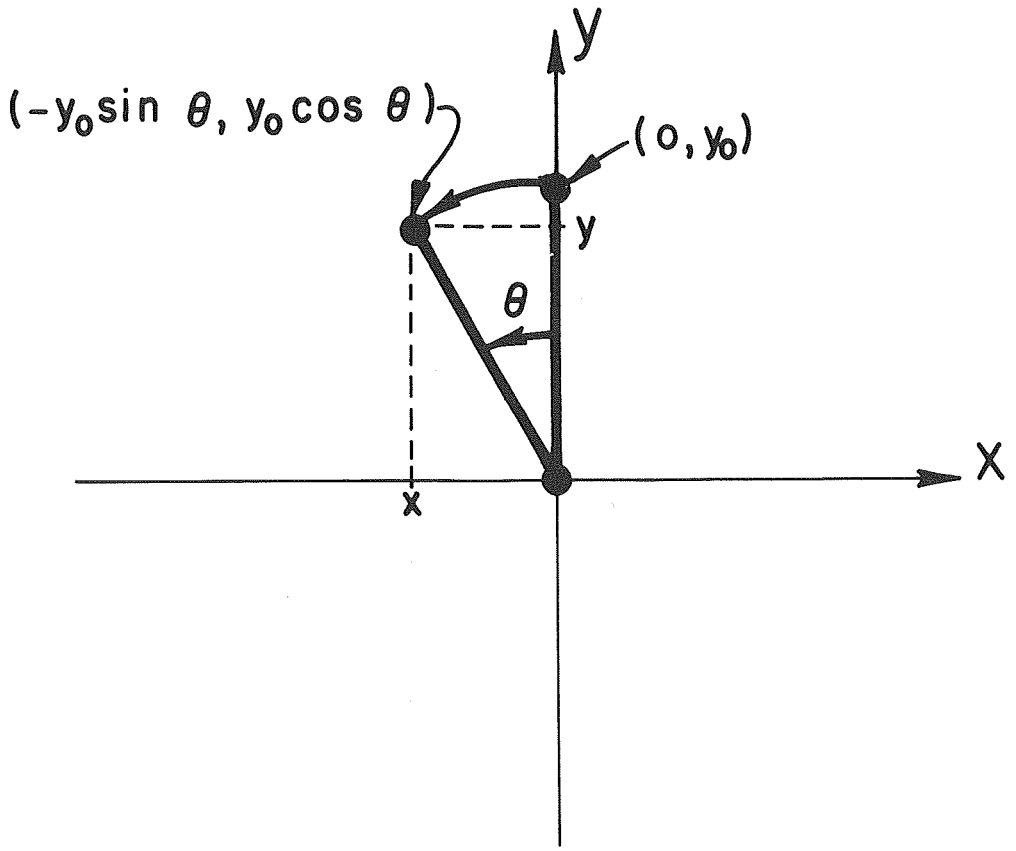


Figure 4.- Case two - the abscissa equaling zero

In the third case, Figure 5, the polar angle, α, i.e., the angle between the x-axis and the line from the point to the origin, must be calculated. The sum angle, $\alpha + \theta$, of the polar angle and the rotation angle is also used. The distance, r, to

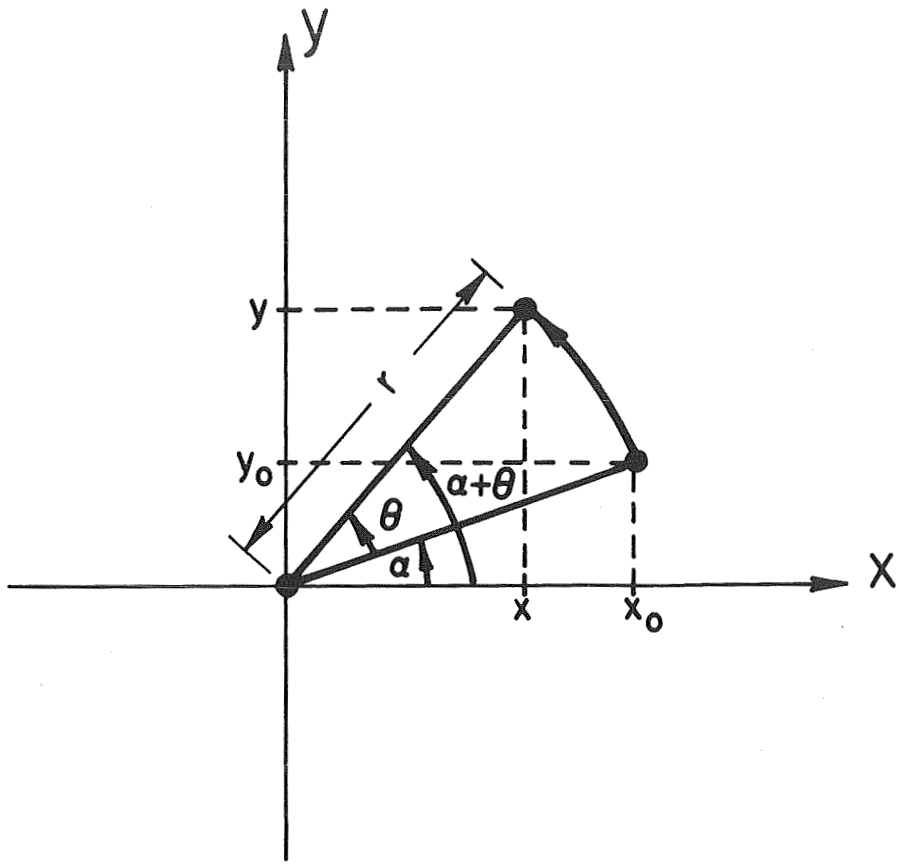


Figure 5.- Case three - the endpoint lying between the coordinate axes

the point is equal to the abscissa divided by the cosine of the polar angle:

$$r = x_o/\cos\ \alpha.$$

Since the distance to the origin remains the same, the new abscissa equals the distance r times the cosine of the sum angle:

$$x = r\ \cos\ (\alpha + \theta).$$

Combining these two equations gives

$$x = x_o\ \cos\ (\alpha + \theta)/\cos\ \alpha.$$

Similarly, using the sine functions for the ordinate:

$$y = y_o\ \sin\ (\alpha + \theta)/\sin\alpha.$$

Every point in the plane can now be rotated through any angle. This tells how the new coordinates of the endpoints are calculated. Next, the program which actually rotates the lines will be described.

This program, Appendix A, is a FORTRAN subroutine. Its name is ROTATE, and the arguments are FIG, LINNUM, XANGLE, YANGLE, and ZANGLE. FIG, which will be described later, is an array containing the line segments of the figure; LINNUM is the number of straight lines to be displayed; XANGLE, YANGLE, and ZANGLE are the angles of rotation around the x-axis, the y-axis, and the z-axis, respectively.

The array FIG is a three-dimensioned array of the form FIG (A, B, C). The first dimension, A, can vary from 1 to 3; A = 1 means an x-coordinate is being dealt with, A = 2 means a y-coordinate, and A = 3 means a z-coordinate. The second dimension, B, can be either 1 or 2. This determines the two endpoints of a line; B = 1 is the first endpoint, B = 2 is the other. These are used in the plotting routines to specify a line to be drawn from the first endpoint to the second. The third dimension, C, can vary between 1 and the number of line segments in the figure. Thus each line is distinguished from all others by the number in C.

An example of the proper use of the FIG array is given in Figure 6. A cube is given with two of its twelve edges arbitrarily numbered 4 and 7; their endpoints are a, b, and c. Edge 7 goes from a to b; edge 4 goes from b to c. Thus FIG (1, 1, 7) equals the x-coordinate (A = 1) of the first endpoint (B = 1) of line 7 (C = 7), or the value of the x-coordinate at point a. FIG (3,2,4) equals the z-coordinate (A = 3) of the second endpoint (B = 2) of line 4 (C = 4), or the value of the z-coordinate at point c. FIG (1,2,7) = FIG (1,1,4), FIG (2,2,7) = FIG (2,1,4), and FIG (3,2,7) = FIG (3,1,4) since the point b is both the second endpoint of line 7 and the first endpoint of line 4. Thus FIG (A,B,C) means the A coordinate of the B endpoint of line C.

The angles XANGLE, YANGLE, and ZANGLE are given in degrees to any decimal accuracy required. The first thing the program does is to convert these angles into radians for the computer to use. XANG, YANG, and ZANG store the corresponding converted angles.

To avoid unnecessary calculations and thus save on computer time, before each rotation a test is made to determine if the rotation angle is zero; if so, the rotation around that particular axis is passed over, and the next rotation is considered.

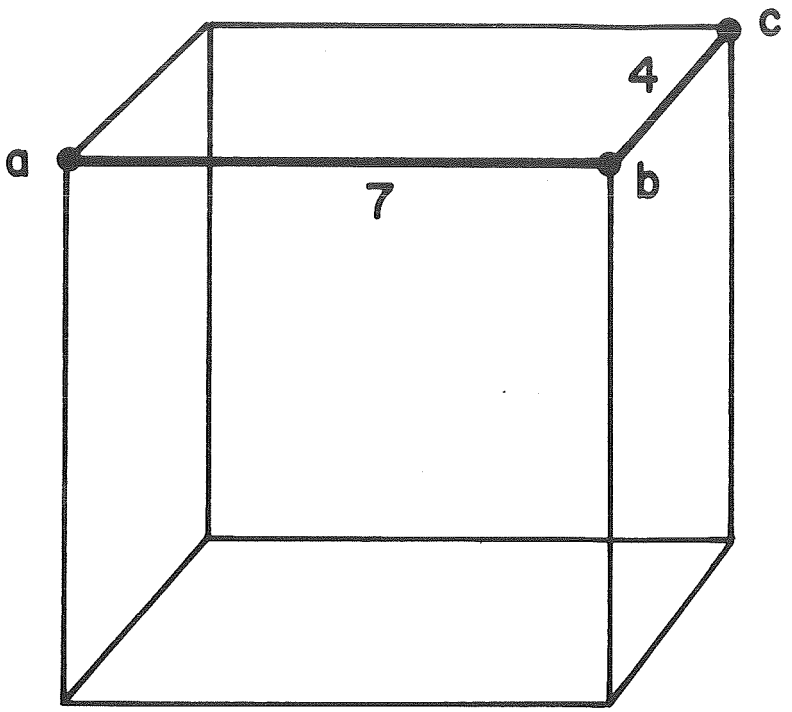


Figure 6.- Example for the proper use of the FIG array

Since the basic forms of the three axis rotations are similar, only the first rotation, the x-axis rotation, will be discussed. The other two can be arrived at by appropriately switching the names of the x-, y-, and z-coordinates. In the x-axis case, the abscissas are the y values and the ordinates are the z values. The direction of rotation corresponds to that which takes a point on the +y-axis and rotates it in a circle around the x-axis to a point on the +z-axis.

The rotation of the entire figure is accomplished by the rotation of each endpoint in the figure. This is done in two nested DO loops which vary from 1 to the number of lines and from 1 to 2 to rotate both endpoints of every line. The complete set of rotation equations is contained in these DO loops. First, the z-coordinate is tested to see if it is zero. If it is, this is case one, and the proper rotation equations are applied in terms of the FIG array and the angle XANG. If the z-coordinate is not zero, then the y-coordinate is tested to see if it is zero. If it is, this is case two, and the equations for this case are applied. If neither coordinate is zero, then it is case three, and additional angles must be calculated. The two extra angles are represented by *ANG1 and *ANG2. The asterisk is replaced by X, Y, or Z, depending on which axis rotation is being used. In this case, the angles would be XANG1 and XANG2. XANG1 is the polar angle and is computed by using the arctangent

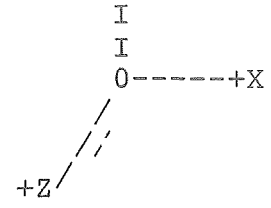
function, ATAN. The value of the tangent is simply the z-coordinate divided by the y-coordinate. The arctangent is easily calculated and is stored in XANG1. XANG2 is just the sum of XANG1 and the rotation angle, XANG. Now, using these two angles, the rotation of the point is performed. The DO loops will then take the next point and rotate it. This will continue until all the points are rotated and will then go on to the y- and z-axis rotations and compute them in a similar way.

Thus a figure in space can easily be rotated by simple trigonometric functions. As a degenerate case of three-dimensional rotation, a two-dimensional figure can be rotated by using only the rotation around the z-axis. This can easily be accommodated; but the main purpose of this subroutine is to give the viewer the option of picking any vantage point from which to view the three-dimensional figure under question.

APPENDIX A

FORTRAN SUBROUTINE PROGRAM

```
      SUBROUTINE ROTATE (FIG,LINNUM,XANGLE,YANGLE,ZANGLE)
C-----
C-----FIG    = THE ARRAY CONTAINING THE COORDINATES OF
C-----         THE ENDPOINTS OF THE LINES IN THE FIGURE
C-----
C-----LINNUM = THE NUMBER OF LINES IN THE FIGURE
C-----
C-----XANGLE = THE ANGLE OF ROTATION AROUND
C-----         THE X-AXIS FROM +Y TO +Z
C-----
C-----YANGLE = THE ANGLE OF ROTATION AROUND
C-----         THE Y-AXIS FROM +Z TO +X
C-----
C-----ZANGLE = THE ANGLE OF ROTATION AROUND
C-----         THE Z-AXIS FROM +X TO +Y
C-----
      DIMENSION FIG(3,2,LINNUM)
C-----
C-----ANGLES IN DEGREES ARE CONVERTED TO RADIANS
C-----FOR USE IN TRIGONOMETRIC FUNCTIONS
C-----
      XANG=XANGLE*3.14159/180.
      YANG=YANGLE*3.14159/180.
      ZANG=ZANGLE*3.14159/180.
C-----
C-----ROTATION IS MEASURED FROM ABSCISSA TO ORDINATE
C-----
C-----FOR EACH ROTATION THERE ARE THREE POSSIBILITIES -
C-----(1)ORDINATE=0, (2)ABSCISSA=0, (3)NEITHER=0
C-----
C-----CASE (1) - ORDINATE = 0
C-----NEW ORDINATE=(ABSCISSA) X SIN(ROTATION ANGLE)
C-----NEW ABSCISSA=(ABSCISSA) X COS(ROTATION ANGLE)
C-----
C-----CASE (2) - ABSCISSA = 0
C-----NEW ABSCISSA=-(ORDINATE) X SIN(ROTATION ANGLE)
C-----NEW ORDINATE= (ORDINATE) X COS(ROTATION ANGLE)
C-----
C-----CASE (3) - NEITHER = 0
C-----THE POLAR ANGLE IS FIRST CALCULATED = *ANG1
C-----THE SUM ANGLE OF THE POLAR ANGLE PLUS THE ROTATION ANGLE IS
C-----THEN CALCULATED = *ANG2
C-----NEW ABSCISSA=(ABSCISSA) X COS(SUM ANGLE) / COS(POLAR ANGLE)
C-----NEW ORDINATE=(ORDINATE) X SIN(SUM ANGLE) / SIN(POLAR ANGLE)
C-----
      IF(XANG) 10,11,10
```



```
C-----
C-----IF XANGLE=0, IT IS UNNECESSARY TO COMPUTE ANY ROTATION
C-----
   10 CONTINUE
      DO 1 LIN=1, LINNUM
      DO 1 I=1,2
      IF(FIG(3,I,LIN)) 20,21,20
   21 CONTINUE
C-----
C-----CASE (1) - X-AXIS ROTATION
C-----
      FIG(3,I,LIN)=FIG(2,I,LIN)*SIN(XANG)
      FIG(2,I,LIN)=FIG(2,I,LIN)*COS(XANG)
      GO TO 1
   20 CONTINUE
      IF(FIG(2,I,LIN)) 22,23,22
   23 CONTINUE
C-----
C-----CASE (2) - X-AXIS ROTATION
C-----
      FIG(2,I,LIN)=-FIG(3,I,LIN)*SIN(XANG)
      FIG(3,I,LIN)= FIG(3,I,LIN)*COS(XANG)
      GO TO 1
   22 CONTINUE
C-----
C-----CASE (3) - X-AXIS ROTATION
C-----
      XANG1=ATAN(FIG(3,I,LIN)/FIG(2,I,LIN))
      XANG2=XANG1+XANG
      FIG(2,I,LIN)=FIG(2,I,LIN)*COS(XANG2)/COS(XANG1)
      FIG(3,I,LIN)=FIG(3,I,LIN)*SIN(XANG2)/SIN(XANG1)
    1 CONTINUE
   11 CONTINUE
      IF(YANG) 12,13,12
C-----
C-----IF YANGLE=0, IT IS UNNECESSARY TO COMPUTE ANY ROTATION
C-----
   12 CONTINUE
      DO 2 LIN=1,LINNUM
      DO 2 I=1,2
      IF(FIG(1,I,LIN)) 30,31,30
   31 CONTINUE
C-----
C-----CASE (1) - Y-AXIS ROTATION
C-----
      FIG(1,I,LIN)=FIG(3,I,LIN)*SIN(YANG)
      FIG(3,I,LIN)=FIG(3,I,LIN)*COS(YANG)
      GO TO 2
   30 CONTINUE
      IF(FIG(3,I,LIN)) 32,33,32
   33 CONTINUE
```

```
C-----
C-----CASE (2) - Y-AXIS ROTATION
C-----
      FIG(3,I,LIN)=-FIG(1,I,LIN)*SIN(YANG)
      FIG(1,I,LIN)= FIG(1,I,LIN)*COS(YANG)
      GO TO 2
   32 CONTINUE
C-----
C-----CASE (3) - Y-AXIS ROTATION
C-----
      YANG1=ATAN(FIG(1,I,LIN)/FIG(3,I,LIN))
      YANG2=YANG1+YANG
      FIG(3,I,LIN)=FIG(3,I,LIN)*COS(YANG2)/COS(YANG1)
      FIG(1,I,LIN)=FIG(1,I,LIN)*SIN(YANG2)/SIN(YANG1)
    2 CONTINUE
   13 CONTINUE
      IF(ZANG) 14,15,14
C-----
C-----IF ZANGLE=0, IT IS UNNECESSARY TO COMPUTE ANY ROTATION
C-----
   14 CONTINUE
      DO 3 LIN=1,LINNUM
      DO 3 I=1,2
      IF(FIG(2,I,LIN)) 40,41,40
   41 CONTINUE
C-----
C-----CASE (1) - Z-AXIS ROTATION
C-----
      FIG(2,I,LIN)=FIG(1,I,LIN)*SIN(ZANG)
      FIG(1,I,LIN)=FIG(1,I,LIN)*COS(ZANG)
      GO TO 3
   40 CONTINUE
      IF(FIG(1,I,LIN)) 42,43,42
   43 CONTINUE
C-----
C-----CASE (2) - Z-AXIS ROTATION
C-----
      FIG(1,I,LIN)=-FIG(2,I,LIN)*SIN(ZANG)
      FIG(2,I,LIN)= FIG(2,I,LIN)*COS(ZANG
      GO TO 3
   42 CONTINUE
C-----
C-----CASE (3) - Z-AXIS ROTATION
C-----
      ZANG1=ATAN(FIG(2,I,LIN)/FIG(1,I,LIN))
      ZANG2=ZANG1+ZANG
      FIG(1,I,LIN)=FIG(1,I,LIN)*COS(ZANG2)/COS(ZANG1)
      FIG(2,I,LIN)=FIG(2,I,LIN)*SIN(ZANG2)/SIN(ZANG1)
    3 CONTINUE
   15 CONTINUE
      RETURN
      END
```

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
WASHINGTON, D. C. 20546

OFFICIAL BUSINESS

FIRST CLASS MAIL

POSTAGE AND FEES PAID
NATIONAL AERONAUTICS AND
SPACE ADMINISTRATION

POSTMASTER: If Undeliverable (Section 158 Postal Manual) Do Not Return

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546