THE UNIVERSITY OF MICHIGAN
Technical Report 30

DRAWL 70: A COMPUTER GRAPHICS LANGUAGE

B. Herzog
Fred Shadko

CONCOMP: Research in Conversational Use of Computers
F. H. Westervelt, Director
ORA Project 07449

Supported by:

ADVANCED RESEARCH PROJECTS AGENCY
Department of Defense
Washington, D.C.

Contract No. DA-49-083 OSA-3050
ARPA Order No. 716

Administered through:

OFFICE OF RESEARCH ADMINISTRATION
Ann Arbor

August 1970

ABSTRACT

The DRAWL language provides a simple means of defining a graphical composition and specifying operations on it. A catalog of parts is kept; any defined item may be re-used any number of times. Changes in viewing angle, scale, absolute location, and projection are easily affected in three dimensions via homogeneous coordinate projective geometry. Graphical output is available on cathode-ray tube-displays, digital-incremental plotters, and on-line computer line-printers and remote printing terminals.

TABLE OF CONTENTS

LIST OF FIGURES

PREFACE

The DRAWL language was conceived in conjunction with a computer graphics course offered during the Fall term, 1966, by the Industrial Engineering Department, the University of Michigan. It was actually implemented.

The language is still in a state of flux as a result of continuing efforts to improve its functional capabilities and overcome some of its original limitations and deficiencies.

The main intent of the DRAWL language was to create a teaching device that would provide for a relatively simple means of defining a graphical composition and specifying subsequent operations without requiring the user to be trained as a programmer. It was envisioned that DRAWL could be a particularly useful tool, since once an object or an assembly was defined, changes in the viewing angle (perspective), scale, or projection could be obtained easily. The same task performed manually would ordinarily require many hours at a drawing board and probably involve tedious and repetitious operations.

In essence, DRAWL provides a mechanism for presenting an idea to a person, defining an object completely before actually drawing it, transmitting precise information about the object under consideration, and making meaningful transformations on the object. The graphical output is particularly meaningful when working with a complex object, because as an object's complexity increases the individual's ability to visualize and construct the object is impaired markedly.

Another important factor that led to the development of DRAWL was the need to provide for a cataloging of parts. Such a "catalog of parts" simplifies many graphics problems. This feature is not always provided in some language structures designed for controlling drafting or plotting machines. In DRAWL not only can a "catalog of parts" be easily specified, but a set of translations and rotations can be recalled from memory on command. This gives some insight into the form, depth, and flexibility of the language.

PROGRAMMER'S NOTE

DRAWL is the "all-obvious" acronym for DRAWLanguage. DRAWL was originally written in the MAD language and implemented on an IBM 7090 computer at the University of Michigan. With the arrival of a new system, DRAWL has been rewritten in FORTRAN IV and is now running under the Michigan Terminal System (MTS) on an IBM 360/67 computer and its associated plotting system, the CALCOMP 780/763 Digital Incremental Plotter.

The sequence of operations involved in utilizing DRAWL as a computer graphics tool is depicted in Figure 1.

All DRAWL instructions are FORTRAN subroutine or function "calls." This type of organization enables the language to retain simplicity without compromising functional utility. The more experienced user can best utilize the language by appropriately combining DRAWL instructions with other FORTRAN statements.

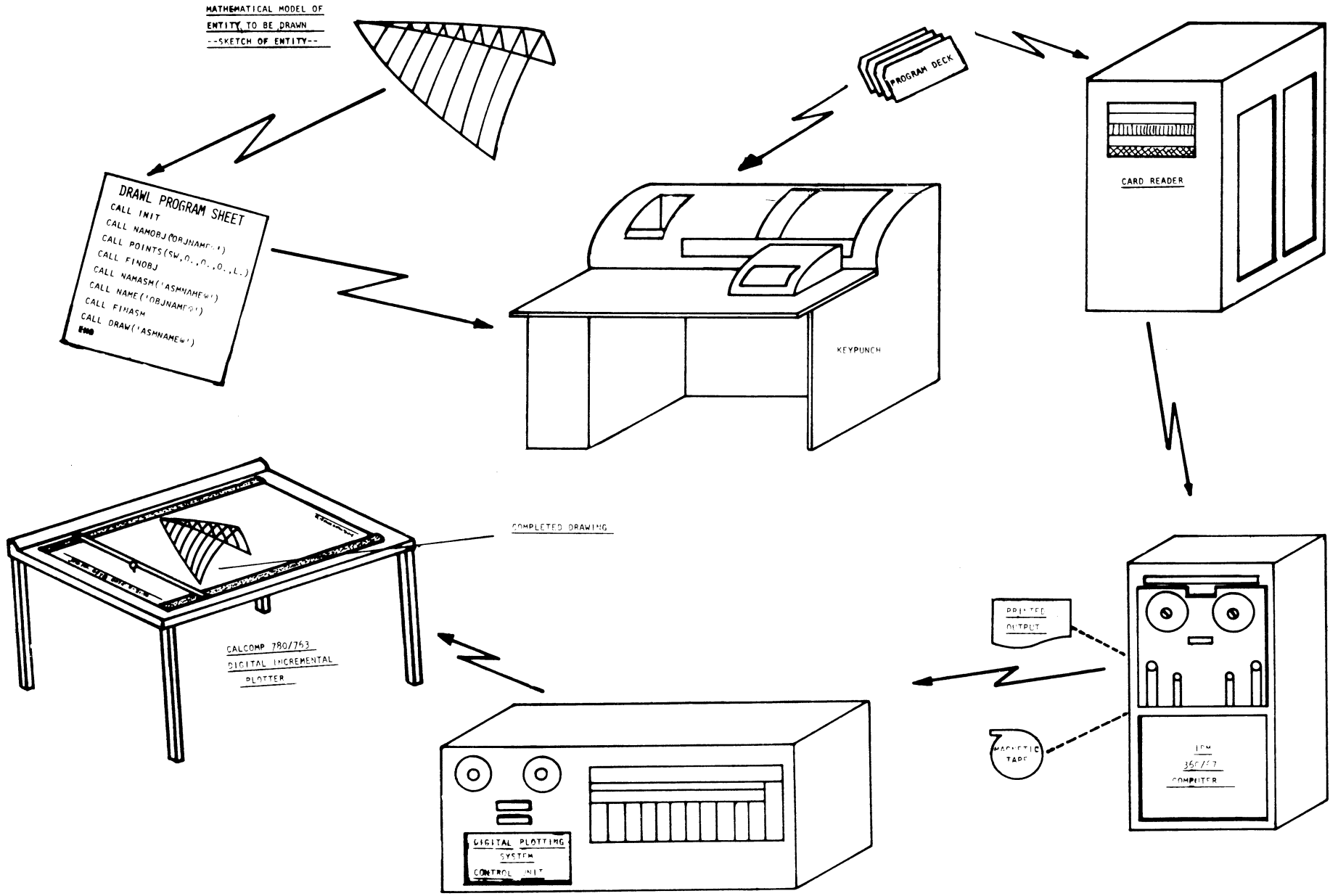

MATHEMATICAL MODEL OF ENTITY TO BE DRAWN
--SKETCH OF ENTITY--
PROGRAM DECK
CARD READER
DRAWL PROGRAM SHEET
CALL INIT
CALL NAMOBJ(OBJNAME1')
CALL POINTS(SW,0.,0.,0.,L.)
CALL FINOBJ
CALL NAMASM('ASMNAME1')
CALL NAME('OBJNAME1')
CALL FINASM
CALL DRAW('ASMNAME1')
END
KEYPUNCH
COMPLETED DRAWING
CALCOMP 780/763
DIGITAL INCREMENTAL
PLOTTER
PRINTED
OUTPUT
MAGNETIC
TAPE
IBM
360/67
COMPUTER
DIGITAL PLOTTING
SYSTEM
CONTROL UNIT

Figure 1

INTRODUCTION

Drafting is a method of representing projections of three-dimensional objects in two dimensions. The DRAWL language is a drafting language that takes the pencil out of the draftsman's hands and gives it to the computer. The resulting drawings are easily changed, and more easily stored.

The DRAWL language relieves the user of tedious, noncreative drafting chores and thus provides him with more time for creative work.

The basis of this drafting system is that all objects are to be described by straight lines connecting specified points in space. Further, it is assumed that a given object is a set of lines connected in one logical element. Thus, generally, invisible lines connecting line segments are required, as illustrated by the rectangular parallelopiped (Figure 2), where the dotted lines represent the invisible lines. All objects are wire-frame figures; no shading or solidity is considered.

These line segments are defined by sequentially specifying the starting point, all intermediate points, and the end point. Information specifying the visibility of each segment accompanies these definitions.

Once a number of objects are so defined they may be associated to become an assembly. Several assemblies and objects may, in turn, be collected into another assembly. Conceptually, assemblies may be nested within assemblies to any depth. Thus objects are defined by points, and assemblies are defined as a collection of objects, or as objects and previously defined assemblies. This hierarchy is illustrated in Figure 3.

To name and construct, i,e., define, objects and assemblies, the following key words and commands are needed:

NAMOBJ	NAMASM
POINTS	NAME
INCRPT	FINASM
FINOBJ	

Since each of these instructions is a call on a subroutine, each must be preceded by the word CALL. Their use and purpose are described in the sequel. Means for manipulating geometric elements are described; transformations are defined and carried out by the key word instructions

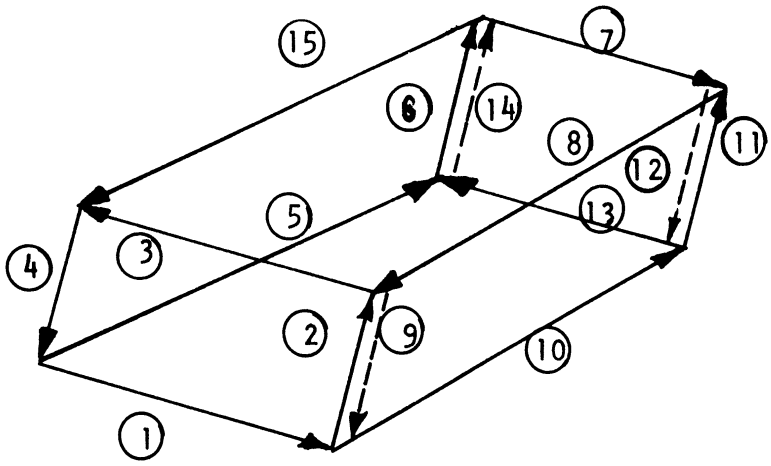


Figure 2 Rectangular Parallelopiped
(DRAWL Specification)

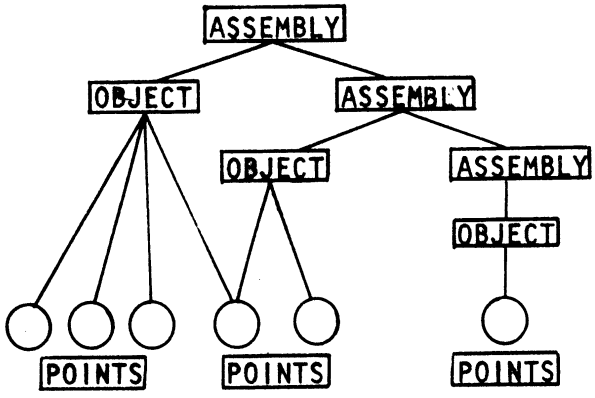


Figure 3 DRAWL Hierarchy

CALL NAMTRA

and

CALL TRANSF.

Several other key words are provided for executing drawings and other convenient functions. The following paragraphs describe the rules associated with these key words and provide illustrative examples.

Definition of Points

One defines a point by specifying its homogeneous coordinates x, y, z, and w, together with a switch that determines the character of the line element. The switch must be set to define the line connecting the last point to the present point as "seen" or "unseen." The location of the new point may be specified by its "absolute coordinates," x, y, z, w, or its "incremental coordinates," $\Delta x, \Delta y, \Delta z, \Delta w$, with respect to the last point.

These four coordinates, x, y, z, and w, are the homogeneous coordinates from which the physical coordinates X, Y, Z, are obtained by

$$X = \frac{x}{w}, \; Y = \frac{y}{w}, \; Z = \frac{z}{w} .$$

The X, Y, and Z coordinates of the point are given in terms of the homogeneous variables XVALUE, YVALUE, ZVALUE, and WVALUE as

$$X = XVALUE/WVALUE$$
$$Y = YVALUE/WVALUE$$
$$Z = ZVALUE/WVALUE$$

where WVALUE is actually a scaling factor. It can be verified that if all points of an object were specified with $w = 3$ rather than $w = 1$ the effect would be to decrease the overall size of the object by one-third (1/3). Two points

$$P1 = (SWITCH1, X1, Y1, Z1, W1)$$

and

$$P2 = (SWITCH2, X2, Y2, Z2, W2)$$

are therefore equivalent if and only if

$$X1/W1 = X2/W2 ,$$

$$Y1/W1 = Y2/W2 ,$$

and

$$Z1/W1 = Z2/W2 .$$

The main purpose of the W component, however, is not to facilitate a change in the scale or size of an object alone, but this formulation permits a uniform treatment of all transformations, rotation, translation, and projection, as will be seen subsequently.

THE NAMING AND DEFINING SEQUENCE FOR OBJECTS

DRAWL utilizes homogeneous coordinates* in defining points. The following sequence is the rule for naming and defining objects

```
CALL NAMOBJ('Name of Object')

CALL POINTS  (SWITCH, XVALUE, YVALUE, ZVALUE, WVALUE)
     INCRPT

.... ....

.... ....

CALL INCRPT  (SWITCH, XVALUE, YVALUE, ZVALUE, WVALUE)
     POINTS

CALL FINOBJ
```

where

CALL NAMOBJ initiates the naming of an object

CALL POINTS is used for specifying absolute point data

CALL INCRPT is used for specifying incremental point data

CALL FINOBJ terminates the sequence of point definition for this object

Name of Object may be any character string of one to eight characters terminated by an @ sign.

The parameters associated with POINTS are:

---

*See Appendix A.

CALL POINTS (SWITCH, XVALUE, YVALUE, ZVALUE, WVALUE)

Scale factor, typically set to unity (must be floating point).

The values of the three-space coordinates (must be floating point variables or numbers).

Switch for specifying logical information (must be integer).

SWITCH is an integer providing required logical signals:

If 1 then line ending at this point will be visible

If 0 then line ending at this point will be invisible.

For example, the points of the object in Figure 4 are specified by

```
CALL POINTS (0,0.,1.,0.,1.)
CALL POINTS (1,1.,1.,0.,1.)
CALL POINTS (1,0.,2.,0.,1.)
CALL POINTS (1,-1.,1.,0.,1.)
CALL POINTS (1,0.,1.,0.,1.)
```

Note that the first point of every object should be the end of an invisible line.
INCRPT uses the same logical switches, but the remaining parameters are somewhat different. CALL INCRPT (SWITCH, X, Y, Z, W) has the effect of adding $\frac{X}{W}$ to the XVALUE of the previous CALL POINTS, $\frac{Y}{W}$ to the previous YVALUE, and $\frac{Z}{W}$ to the previous ZVALUE. Conveniently then an object can be defined relative to a starting point given by CALL POINTS.

THE NAMING AND CREATING SEQUENCE FOR ASSEMBLIES

The rule for naming and creating assemblies is:

```
CALL NAMASM ('Name of Assembly')

CALL NAME   ('Name of First Element')
            ....................
            ....................
CALL NAME   ('Name of Last Element')

CALL FINASM
```

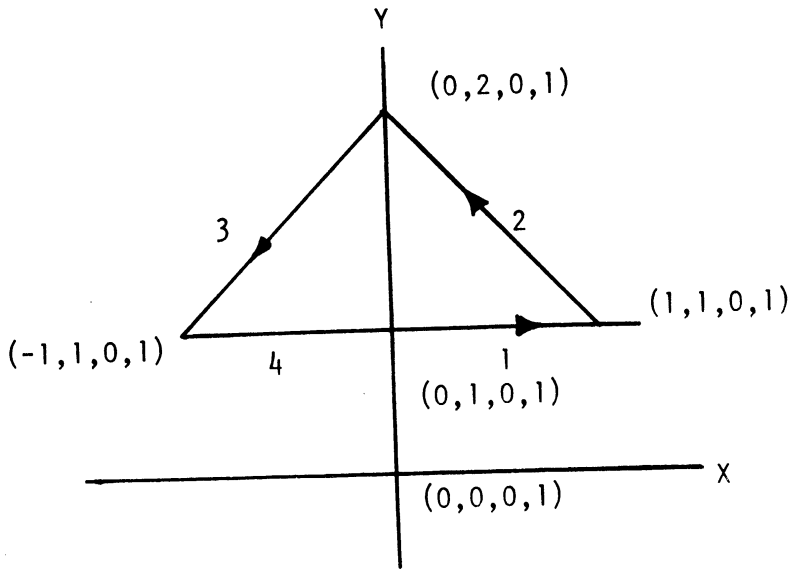


Figure 4. Specifications of a Triangle.

where

CALL NAMASM initiates the naming of an assembly

CALL NAME names an element

CALL FINASM terminates the naming of the elements of an assembly and closes the assembly.

Name of Assembly may be any character string of eight or fewer characters, terminated by an @ sign.

Name of Element must be the name of a defined assembly or object.

TRANSFORMATIONS IN DRAWL

By appropriately defining a transformation a user can control the rotation and translation of an object or assembly. Other uses include copying, scaling, and projection, including perspectives. There are two DRAWL instructions associated with transformations. The first names the transformation and specifies its elements, while the second is a transformation operator. The first is

```
CALL NAMTRA('Name of Matrix', ARG(1,1),ARG(1,2),ARG(1,3),ARG(1,4),
                              ARG(2,1),ARG(2,2),ARG(2,3),ARG(2,4),
                              ARG(3,1),ARG(3,2),ARG(3,2),ARG(3,4),
                              ARG(4,1),ARG(4,2),ARG(4,3),ARG(4,4))
```

where

CALL NAMTRA stores the name of the transformation, the values of the elements of the transformation matrix (A(1,1),..., A(4,4)) and their association.

Name of Matrix may be any character string of eight or fewer characters terminated by an @ sign.

The transformation is specified as a single four-by-four matrix; it can accomplish a full projective transformation because DRAWL employs homogeneous coordinates.

The elements of the transformation matrix, ARG(1,1) through ARG(4,4), can be separated into various functional groups. Figure 5 depicts the breakdown of the transformation matrix. Any one transformation matrix can also be thought of as the concatenation of the four functionally independent matrices.*

---

* For a general discussion of transformations see Appendix A.

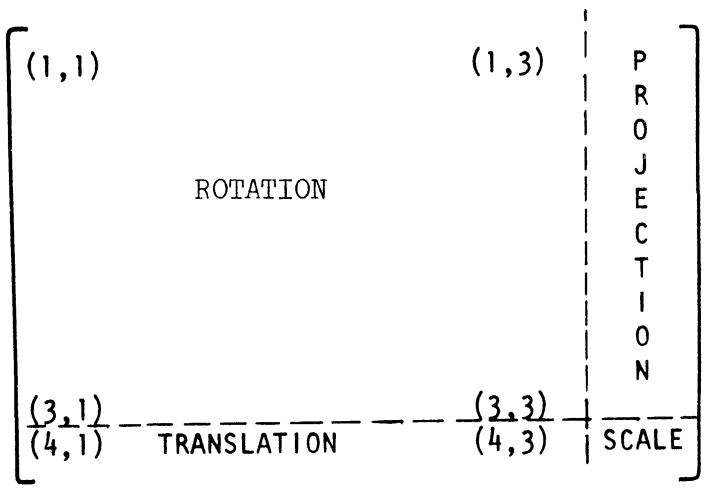


Figure 5. DRAWL Transformation Matrix Breakdown

Combination of the functionally independent submatrices is possible and at times very desirable since sometimes one transformation matrix can perform all desired functions. It should be noted that if both a translation and rotation are specified the translation is performed after the rotation. In general, the commutative law does not apply to matrix multiplication. Therefore, when transformation matrices are concatenated their sequence is very important.

PERFORMING A TRANSFORMATION

The command for performing a transformation after the transformation matrix has been defined is

```
CALL TRANSF ('New Entity Name', 'Source Entity
              Name', 'Transformation Name')
```

where

New Entity Name is the name of the new entity resulting from the transformation.

Source Entity Name is the name of the old entity to be transformed.

Transformation Name specifies the name of the transformation matrix used.

See the section on program conventions for a discussion of the assumptions made when New Entity Name is the same as Old Entity Name.

EXECUTING THE DRAWING

The command

```
CALL DRAW('ENTITY@')
```

will produce a Calcomp plot of any previously defined object or assembly named 'ENTITY'. This command can be programmed as soon as 'ENTITY' has been fully defined. Each DRAW command produces one plot only. A new plot is started with each DRAW command.

A drawing is prepared by using only the X, Y coordinates of the points in objects or assemblies to be drawn.

Specifying

```
CALL SHOW('ENTITY@')
```

will cause 'ENTITY' to be displayed on the CRT display on which the user is running.

CALL PRTPLT('ENTITY@')

will cause 'ENTITY' to be drawn without connecting lines on the computer's line-printer.

CALL PRTOUT('ENTITY@')

will print the switch and x-y values of 'ENTITY' to be printed out.

CALL PRTPUT('ENTITY@',LDN)

will cause the switch and X-Y values of 'ENTITY' to be written on the file or device attached to Logical Device Number LDN at RUN time.

COMMENTS ON DRAWL NAMES AND OTHER PROGRAM CONVENTIONS

DRAWL Names are character strings* of from one to eight characters. If there are less than eight characters, the string must be terminated by an "@" (at-sign). The character "@" may not be imbedded in a DRAWL name, and when used as a terminal character, is not part of the name. FORTRAN IV requires that character strings be enclosed in primes (').

The following are legal DRAWL names:

'NAME@'	
'NAMOFOBJ'	No "@" required since length is eight
'N''OBJECT'	See the FORTRAN IV rules for imbedding primes
'1@'	A non-alphabetic name
'@'	A null or all-blank name.

Note that the length of the string 'NAME@' is five while the length of a DRAWL name is always eight. The character "@" acts as a signal to the DRAWL processor to blank out all the characters to its right (including the "@") up to a total string length of eight.

The following are illegal DRAWL names:

'NAMOFOBJ'	String too long
'NAME'	No terminal "@"
'N'OBJECT	Violates FORTRAN IV syntax rules.

---

* A character string is a series of adjacent letters, numerals, and/or special characters (e.g., * , . % - + & $ ¢, etc.) Blank or space is also a special character. In FORTRAN IV character strings are read-in and written-out in A-format. All DRAWL names are character strings.

A DRAWL name may not be defined twice in a single DRAWL run. The following routines define DRAWL parts:

```
NAMOBJ  ( or NAMOBA, or NAMOBR)
NAMASM
NAMTRA  ( or NMTRA1, or CMBMAT)
TRANSF  ( first argument only).
```

The only way the values associated with an entity can be altered is via TRANSF, when the first and second arguments are the same (e.g., CALL TRANSF('A@', 'A@', 'Q@')). Users are cautioned that the values associated with an entity name at the time that the entity is included in an assembly will remain associated with the assembly even if the included entity is later transformed. For example:

```
CALL INIT
CALL NAMOBJ('1@')
CALL POINTS(1, 2., 3., 2., 1.)
CALL FINOBJ
CALL NAMASM('1ASM@')
CALL NAME('1@')
CALL FINASM
CALL TRANSF('1@', '1@', TRLATE(1.,1.,1.))
CALL DRAW('1ASM@')
```

will yield a drawing of a point at (2,3,2), not at (3,4,3).

APPENDIX A

Homogeneous Coordinate Projective Geometry

We shall represent a point in space as a row vector whose elements are the three components measuring the location of the point from the origin, say, in a cartesian x-y-z system. Thus a general point P is represented by

$$P = [\, x \; y \; z \,].$$

The manipulations that may be performed on a single point, or a set of points are known as transformations. We may wish to view the same point P from another coordinate system x'-y'-z', or

$$P' = [\, x' \; y' \; z' \,].$$

We ask: What might the values x', y', and z' be if we knew x, y and z ? To answer this we must know the relationship between the two coordinate systems. A unified treatment of all necessary transformations, such as those for displacement or rotation of coordinate systems, for scaling, and for projective geometry, occurs if we utilize the homogeneous coordinate representation of points. We shall now introduce this representation, and then follow with the derivation of a set of transformations.

Homogeneous Coordinates

The same point P is represented by a four-element row vector, thus

$$P = [X \; Y \; Z \; H]$$

from which the conventional components are obtained by

$$x = \frac{X}{H}, \; y = \frac{Y}{H}, \text{ and } z = \frac{Z}{H} \quad .$$

We achieve a certain amount of notational elegance by using biliteral symbols to signify the positional names in the homogeneous row vector. If we replace, for example, X by hx, and H by h , the expression becomes

$$P = [\, hx \; hy \; hz \; h \,].$$

The fourth element may be regarded as a scale factor. Hence, doubling the value of the fourth element, h , halves the physical or usual coordinates x, y and z . Given a homogeneous representation of a point, the physical coordinates can always be computed. The converse is not true, as shown with the physical point [3, 4, 5]. All three of the following homogeneous representations are equally valid and will yield the same specified physical coordinates:

$$P_1 = [\ 3\ 4\ 5\ 1\ ]$$

$$P_2 = [\ 6\ 8\ 10\ 2\ ]$$

$$P_3 = [\ 30\ 40\ 50\ 10\ ]\ .$$

General Formulation of Transformations

Since transformations really concern the relationship among coordinate systems, and since computer graphics problems or applications involve several such relationships, the conventional xyz or x'y'z' notations soon become unwieldy and we need a more generalized notation. We will use right-handed systems having a 1-axis, a 2-axis and a 3-axis. To distinguish one coordinate system, say the α-system, from another, the β-system, we shall call the α-system axes, α_1, α_2 and α_3, and the β-system axes, β_1, β_2 and β_3.

Further, rather than write out the row vector of the coordinates, we can designate a coordinate in the α-system by $[hv]_\alpha$:

$$[hv]_\alpha = [\ hx\ hy\ hz\ h\ ]_\alpha\ .$$

In the notation, then, the name of the coordinate system is a subscript. Thus the relation of a point in the α-system to a point in the β-system is given by :

$$[hv]_\beta = [hv]_\alpha\ M_{\alpha\beta}\ ,$$

where $M_{\alpha\beta}$ is a four-by-four transformation matrix.

Obviously there exists an inverse relationship, that is, given $[hv]_\beta$ we obtain $[hv]_\alpha$ by :

$$[hv]_\alpha = [hv]_\beta\ M_{\beta\alpha}\ ,$$

where $M_{\beta\alpha}$ is the transformation matrix for mapping β representations into α representations. Matrix algebra states that each such transformation matrix is the inverse of the other:

$$M_{\alpha\beta} = M_{\beta\alpha}^{-1}$$

and

$$M_{\beta\alpha} = M_{\alpha\beta}^{-1}.$$

Within each transformation matrix there is a three-by-three partition, R , which effects rotation in a general transformation.

Similarly the row T , effects translation; the column P , projection; and the scalar S , scaling (Fig. A-1). Any general transformation, then, may be broken down into the four sequential transformations: rotation, translation, projection, and scale. Without providing proof, we will move directly to simple cases of each of these situations.

Translation

In Figure A-2 the α-system has been moved with respect to the β-system . The corresponding axes remain parallel, but the origin of the α -system has been displaced by

$[\ \Delta X\ \Delta Y\ \Delta Z\ ]$ in ordinary coordinates,

or

$[\ h\Delta X\ \ h\Delta Y\ \ h\Delta Z\ \ h]$ in homogeneous coordinates.

This situation and the corresponding transformation matrix $M_{\alpha\beta}$ is shown in Figure A-2.

Rotation

For rotation we retain coincidence of the origin but the α axes will be rotated with respect to the β axes. For simplicity, however, we will consider three cases in which the coincidence of one set of corresponding axes is maintained in addition to the coincidence of the origins.

First consider the case where the α_1 and β_1 axes coincide and we rotate the α- system with respect to the β system through an angle θ as shown in Figure A-3. The corresponding values of the transformation $M_{\alpha\beta}$ are also shown in Figure A-3. For the two other cases, rotation about the β_2 axis and rotation about the β_3 axis (see Figures A-4 and A-5) is shown.

Perspective Projection

To illustrate this concept consider the x-y plane as the picture plane, and the eye point to be located at a distance a from the origin on the negative z axis. Then, using the transformation M (Figure A-6) if an object is given in the system, the transformed x,y data will represent the picture coordinates. This is readily proven by the special case of the point Pl in the y-z plane, Figure A-7. The projected point is Pl′.

A word of caution: typically the picture plane lies between the eye point and the object. Mathematically, if the eye point

should lie on the object, this makes z+a have the value zero and requires special computational considerations. Furthermore, should the eye point lie within an object as in simulating an observer walking through an architectural space, we obtain inverted images of that part of the object behind the viewer; special computations could handle this problem. DRAWL notes the case where the eye point coincides with a point in an object, but will draw the inverted images brought about when the eyepoint is "inside" the object.

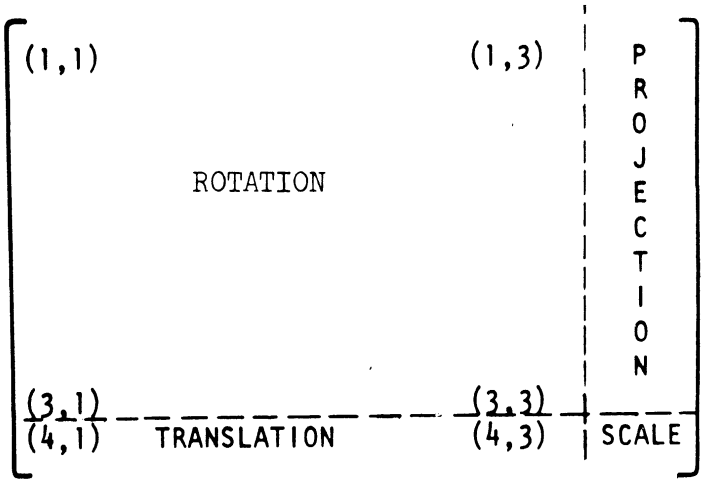
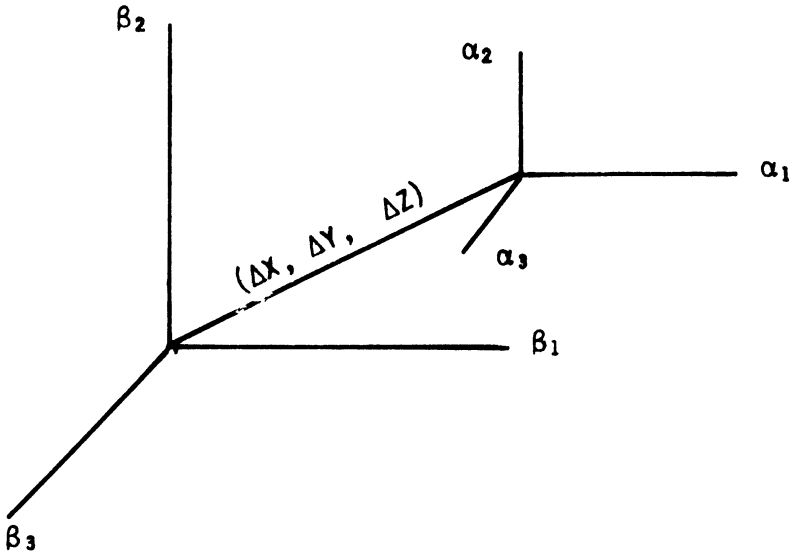


Figure A-1. DRAWL Transformation Matrix Breakdown.

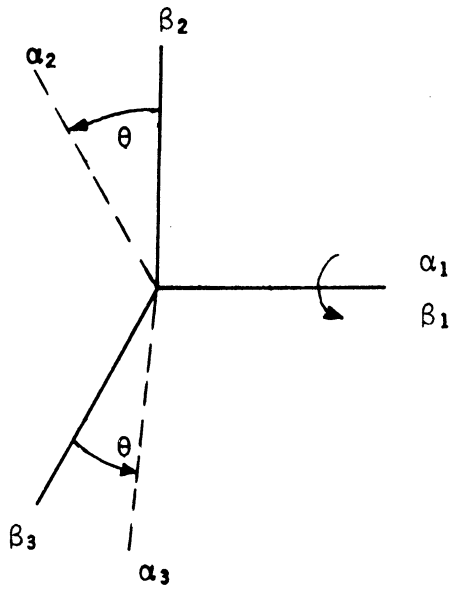


$$M_{\alpha\beta} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \Delta X & \Delta Y & \Delta Z & 1 \end{bmatrix}$$

or,

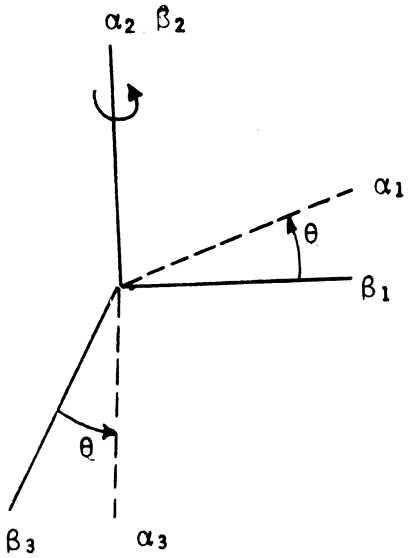
$$M_{\alpha\beta} = \begin{bmatrix} \tilde{h} & & & \\ & \tilde{h} & & \\ & & \tilde{h} & \\ \tilde{h}\Delta X & \tilde{h}\Delta Y & \tilde{h}\Delta Z & \tilde{h} \end{bmatrix}$$

Figure A-2 Translation.



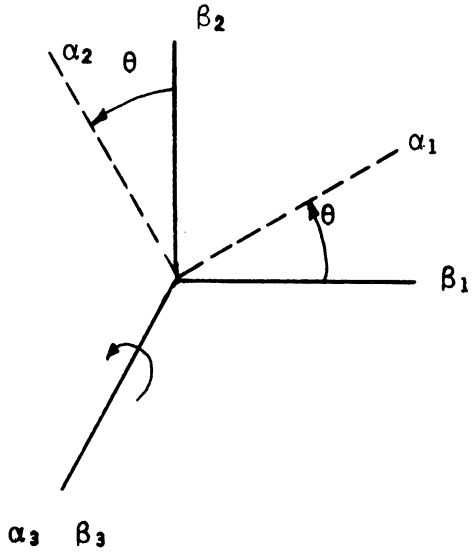
$$M_{\alpha\beta} = \begin{bmatrix} 1. & 0. & 0. & 0. \\ 0. & \cos\theta & \sin\theta & 0. \\ 0. & -\sin\theta & \cos\theta & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

Figure A-3 Rotation about the β_1 axis.



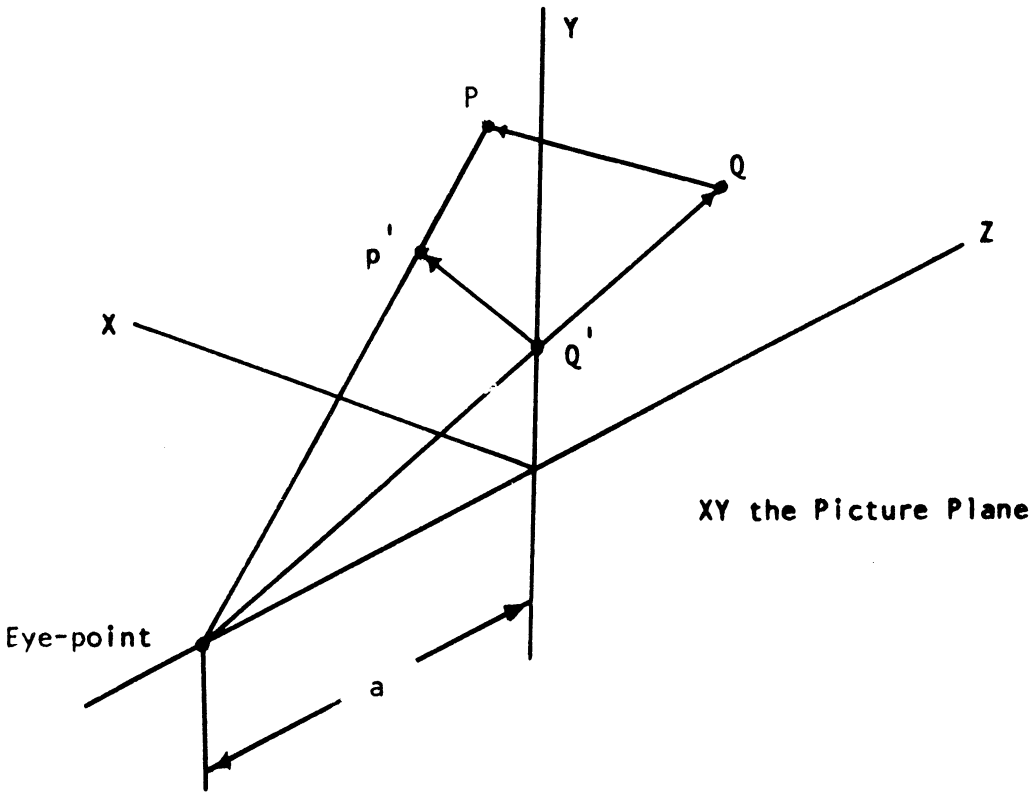
$$M_{\alpha\beta} = \begin{bmatrix} \text{COS}\theta & 0. & -\text{SIN}\theta & 0. \\ 0. & 1. & 0. & 0. \\ +\text{SIN}\theta & 0. & \text{COS}\theta & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

Figure A-4 Rotation about the β_2 axis.



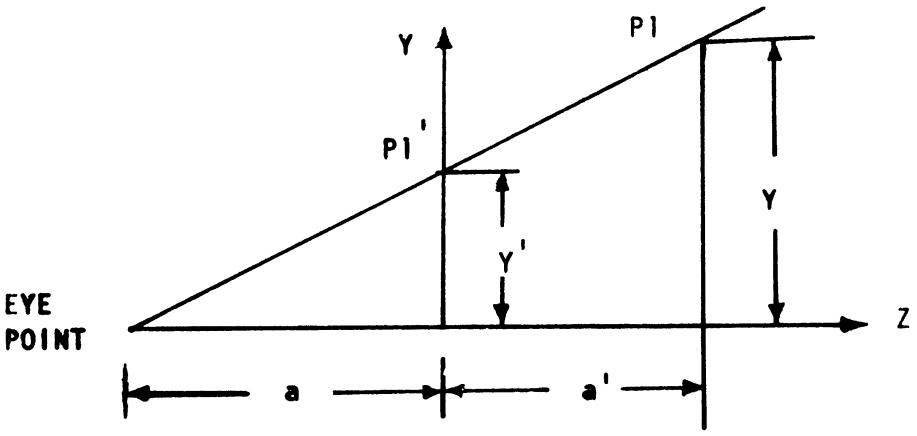
$$M_{\alpha\beta} = \begin{bmatrix} \text{COS}\theta & \text{SIN}\theta & 0. & 0. \\ -\text{SIN}\theta & \text{COS}\theta & 0. & 0. \\ 0. & 0. & 1. & 0. \\ 0. & 0. & 0. & 1. \end{bmatrix}$$

Figure A-5 Rotation about the β_3 axis.



$$M = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1/a \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure A-6 The Perspective Transformation



By similar triangles,

$$\frac{Y}{a+Z} = \frac{Y'}{a}$$

Thus,

$$Y' = Y\frac{S}{a+Z} = \frac{Y}{1+\frac{Z}{a}}$$

Figure A-7 Illustration of the perspective transformation matrix.

APPENDIX B

DRAWL Subroutine Descriptions

INIT

NAME: INIT

FUNCTION: Initialize all counters and pointers within DRAWL, and determine whether the user is operating in batch or conversational mode.

CALLING SEQUENCE:

FORTRAN IV:

```
CALL INIT
```

ARGUMENTS: None

RETURN CODES: None

COMMENTS: If the user is operating in batch mode, headers are printed on each page and an abbreviated newsletter is printed on the first page. The program flow is monitored by printed comments. All such printing is suppressed for conversational users. (See options "PRNT" and "NOPRNT" to override these default settings.) INIT also places the point (0,0,0) in the points table, effectively making the origin the initial pen or light-beam position. Each execution of INIT causes all pointers to be cleared, and all future DRAWL operations are performed as if none had occurred before the call to INIT.

EXAMPLE:

```
CALL INIT
```

NAME:	NAMOBJ
FUNCTION:	Enter the name given as its argument in the name table, open the object, and open the points table.
CALLING SEQUENCE:	FORTRAN IV: CALL NAMOBJ (ENTITY, &1)
ARGUMENTS:	ENTITY - A DRAWL name which will be the name of the object about to be formed via calls to POINTS and INCRPT.
RETURN CODES:	&1 - ENTITY has been used as a DRAWL name before ***DRAWL ERROR-ENTITY ALREADY USED AS DRAWL NAME.
COMMENTS:	This must be the first use of ENTITY as a DRAWL name.
EXAMPLES:	CALL NAMOBJ ('I@') CALL NAMOBJ ('QQSVNUM1')

NAME:	NAMOBA
FUNCTION:	See NAMOBJ
CALLING SEQUENCE:	FORTRAN IV: CALL NAMOBA (ENTITY, &1)
ARGUMENTS:	See NAMOBJ
RETURN CODES:	See NAMOBJ
COMMENTS:	NAMOBA is an alternate entry to NAMOBJ. It is identical to NAMOBJ except that it calls POINTS (0,0.,0.,0.,1.) before exit. Thus objects created via NAMOBA have the origin as their base.
EXAMPLES:	CALL NAMOBA ('3@') CALL NAMOBA ('ABSOLUTE')

NAME: NAMOBR

FUNCTION: See NAMOBJ

CALLING SEQUENCE:

FORTRAN IV:
CALL NAMOBR (ENTITY, &1)

ARGUMENTS: See NAMOBJ

RETURN CODES: See NAMOBJ

COMMENTS:

NAMOBR is an alternate entry to NAMOBJ. It differs from NAMOBJ by calling INCRPT (0,0.,0.,0.,1.) immediately before exit. Thus an object created via NAMOBR has the point last placed in the most recently defined object as its base. A call to NAMOBR before any calls to POINTS elicits:

```
***WARNING--THERE IS NOTHING TO USE AS
   BASE FOR INCREMENTAL CALL.
   (0.,0.,0. ) ASSUMED.
```

EXAMPLE:

```
CALL NAMOBR ('OBJECT #1')
```

NAME: POINTS

FUNCTION: Enter the point defined in the argument list and the associated line-type switch in the points table. The point is considered to be part of the currently opened object.

CALLING SEQUENCE:

FORTRAN IV:
CALL POINTS (ISWIT, X,Y,Z,W,&1)

ARGUMENTS:

ISWIT - A full-word integer line-type switch.
On-line plotter and CRT display:

ISWIT	Function
0	Leave the beam off (pen up) while moving to this point.
1	Leave the beam on (pen down) while moving to this point.

Off-line, wide-bed plotter:

ISWIT	Function
0	Leave the pen up while moving to this point
1	Leave the pen down while moving to this point.
2	Draw a dashed line to this point.
3	Draw a center-line to this point.

X - The floating-point, full-word value of the absolute X-coordinate, the first homogeneous coordinate.

Y - The floating-point, full-word value of the absolute Y-coordinate, the second homogeneous coordinate.

Z - The floating-point, full-word value of the absolute Z-coordinate, the third homogeneous coordinate.

W - The floating-point, full-word value of the scale factor, the fourth homogeneous coordinate.

RETURN CODES:

&1 - If POINTS was called before NAMOBJ, NAMOBA, or NAMOBR was called, i.e., before an object was opened:
***DRAWL ERROR - POINTS CALLED WITH NO NAMED OBJECT VIA NAMOBJ.

COMMENTS:

1. If the fourth homogeneous coordinate, also viewed as a scale factor, is zero (W = 0.) it will be replaced by one (W = 1.) and cause the comment:
***WARNING-SCALE FACTOR OF ZERO, CHANGED TO ONE.
2. If more than 1000 calls to POINTS or INCRPT are executed, execution will be terminated:
***DRAWL ERROR-OVER 1000 POINTS WERE SPECIFIED. TERMINATING VIA "ABORT" SPECIFICATION.
This limit can be altered by changing a DIMENSION statement in the DRAWL source code.

EXAMPLES:

```
CALL POINTS (0,1.,1.,1.,1.,&336)
CALL POINTS (0,0.,0.,0.,1.)
CALL POINTS (Z, 1.,1./16.,1./16.,16.)
CALL POINTS (ISWIT,A,SIN(A),COS(A),1.)
```

NAME: INCRPT

FUNCTION: See POINTS

CALLING SEQUENCE: FORTRAN IV

ARGUMENTS: ISWIT - Same as for POINTS

X - The floating-point, full-word value of the incremental X-coordinate — the first homogeneous coordinate.

Y - The floating-point, full-word value of the incremental Y-coordinate — the second homogeneous coordinate.

Z - The floating-point, full-word value of the incremental Z-coordinate — the third homogeneous coordinate.

W - The floating-point. full-word value of the scale factor — the fourth homogeneous coordinate.

RETURN CODES: See POINTS

COMMENTS: INCRPT executes:

CALL POINTS(ISWIT,X/W+PX,Y/W+PY,Z/W+PZ,W)

where PX is value of the first homogeneous coordinate of the last point defined, PY is the value of the previous second homogeneous coordinate, and PZ is the value of previous third homogeneous coordinate. X, Y, and Z are thus incremental values from the previous (X,Y,Z). In addition to the error comments generated by POINTS, INCRPT can also produce:

***WARNING-THERE IS NOTHING TO USE AS BASE FOR RELATIVE OR INCREMENTAL CALL (0.,0.,0.)ASSUMED.

There must be a call to POINTS before a call to INCRPT or NAMOBR.

EXAMPLE: CALL INCRPT(1,3.,4.,5.,3.)

NAME:	FINOBJ
FUNCTION:	Close the current object definition and record the number of points in the object in the linkage table.
CALLING SEQUENCE:	FORTRAN IV: CALL FINOBJ
ARGUMENTS:	None
RETURN CODE:	None
COMMENTS:	The number of points included in the object just defined is printed.
EXAMPLE:	CALL FINOBJ

NAME:	NAMASM
FUNCTION:	Enter the name given as its argument in the name table, opening the assembly, and open the association table.
CALLING SEQUENCE:	FORTRAN IV CALL NAMASM (ENTITY,&1)
ARGUMENTS:	ENTITY--A DRAWL name which will be used as the name of the assembly about to be created via calls to NAME
RETURN CODES:	&1-ENTITY has been used as a DRAWL name before: ***DRAWL ERROR-ENTITY ALREADY USED AS DRAWL NAME
COMMENTS:	This must be the first appearance of ENTITY as a DRAWL name.
EXAMPLE:	CALL NAMASM ('ENT #1@')

NAME: NAME

FUNCTION: Declare the entity named as its argument to be a member of the currently open assembly.

CALLING SEQUENCE:

FORTRAN IV
CALL NAME (ENTITY,&1,&2,&3)

ARGUMENTS: ENTITY--The DRAWL name of the entity to be included in the currently open assembly. ENTITY must be defined at the time it is used as an argument to NAME.

RETURN CODES:

&1 - The specified name ENTITY was not not found by NAME.
***DRAWL ERROR--ENTITY WAS NOT FOUND BY NAME.

&2 - NAME was called before NAMASM or after FINASM.
***DRAWL ERROR--NAME CALLED WITH NO NAMED ASSEMBLY VIA NAMASM.

&3 - ENTITY was the name of a transformation
***DRAWL ERROR--'ENTITY' IS NOT ENTITY NAME.

COMMENTS: The values currently associated with the name ENTITY will be associated with assembly being defined. These values will remain associated with the assembly being defined despite any future transformation of ENTITY.

EXAMPLES:

```
CALL NAME ('OBJ@', &200,&100,&500)
CALL NAME ('1@', &3)
CALL NAME ('37THOBJ@')
```

NAME:	FINASM
PURPOSE:	Close the currently open assembly and enter the number of entities in the assembly in the association table.
CALLING SEQUENCE:	FORTRAN IV: CALL FINASM
ARGUMENTS:	None
RETURN CODES:	None
COMMENTS:	The number of entities included in the assembly just closed is printed.
EXAMPLE:	CALL FINASM

NAME:

NAMTRA

FUNCTION:

Define a DRAWL transformation matrix.

CALLING SEQUENCE:

FORTRAN IV:

```
CALL NAMTRA(TNAME, A1,A2,A3,A4,
1                  A5,A6,A7,A8,
2                  A9,A10,A11,A12,
3                  A13,A14,A15,A16,
4                  &1)
```

ARGUMENTS:

TNAME - The DRAWL name by which the transformation being defined will be referenced.
A1,...,A16 = Full-word, floating-point values of the desired DRAWL 4-by-4 transformation matrix, ordered by rows (see Appendix A for a detailed discussion of transformation matrices).

RETURN CODES:

&1 - TNAME has already been used as a DRAWL name:
***DRAWL ERROR - TNAME ALREADY USED AS DRAWL NAME.

COMMENTS:

Should an illegal DRAWL transformation matrix be defined (A16=0.), A16 will be set equal to one (A16=1.) with this warning printed:
***SCALE FACTOR OF ZERO,CHANGED TO ONE

EXAMPLE:

```
CALL NAMTRA ('TRI@',1.,0.,0.,0.,
1                 0.,1.,0.,0.,
2                 0.,0.,1.,0.,
3                 10.,-3.,7.,4.,
4                 &5)
```

(Note: four FORTRAN continuation cards have been used here so that the matrix's components R,T,P, and S (see Figure A-1) are readily discernible. Compare this example to the following one.)

```
CALL NAMTRA ('TRI@',1.,0.,0.,0.,0.,1.,0.,
1  0.,0.,0.,1.,0.,10.,-3.,7.,4.,&5)
```

NAME: NMTRA1

FUNCTION: See NAMTRA

CALLING SEQUENCE: FORTRAN IV
CALL NMTRA1 (TNAME,E,&1)

ARGUMENTS: TNAME - The DRAWL transformation name by which the transformation being defined can be referenced.

E - A four-by-four, full-word, floating-point array whose values are interpreted as a four-by-four DRAWL transformation matrix (see Appendix A)

RETURN CODES: See NAMTRA

COMMENTS: See NAMTRA

EXAMPLE: CALL NMTRA1 ('TR2@', ARRAY,&39)

NAME: TRANSF

FUNCTION: Establish the internal associations which will effect a transformation.

CALLING SEQUENCE: FORTRAN IV:

CALL TRANSF (ENTITY1, ENTITY2, TNAME, &1,&2,&3,&4).

ARGUMENTS:

ENTITY1 - The DRAWL name which will be assigned to the newly created entity.

ENTITY2 - The DRAWL name of the entity to be transformed. See the cautions on using ENTITY1 = ENTITY2 in the section on DRAWL names.

TNAME - The DRAWL name of the transformation by which ENTITY2 is to be transformed into ENTITY1.

RETURN CODES:

&1 - The third argument was incorrect:
***DRAWL ERROR--TNAME IS NOT A TRANSFORMATION NAME.

&2 - The second argument was incorrect:
***DRAWL ERROR--ENTITY2 IS A TRANSFORMATION NAME.

&3 - The second or third argument was undefined:
***DRAWL ERROR--ARGUMENT NOT FOUND BY TRANSF.

&4 - ENTITY1 has already been defined, and this is not a simple redefinition, i.e., ENTITY1 ≠ ENTITY2. (See the section on DRAWL names.)
***DRAWL ERROR-ENTITY1 ALREADY USED AS (OBJECT) (ASSEMBLY) (XFORM) NAME

COMMENTS: The user's attention is directed to the section on DRAWL names.

EXAMPLES:

```
CALL TRANSF ('A1@','A1@','TR1@',&70,&80,
1 &65,&75)
CALL TRANSF ('A1@','A2@','TR3@')
CALL TRANSF ('A17@','A1@','TR4@',&7,&9)
```

NAME:

DRAW

FUNCTION:

Produce the drawing of an entity on an off-line, thirty-inch CalComp plotter.

CALLING SEQUENCE:

FORTRAN IV:
CALL DRAW(ENTITY,&1,&2,&3)

ARGUMENTS:

ENTITY - The DRAWL name of the entity to be drawn.

RETURN CODES:

&1 - ENTITY was undefined when DRAW was called.
***DRAWL ERROR - ENTITY NOT FOUND BY DRA
&2 - ENTITY is a transformation name
***DRAWL ERROR - ATTEMPT TO DRAW TRANSFORMATION "ENTITY"
&3 - A perspective transformation operation caused division by zero:
***DRAWL ERROR - ATTEMPT TO PLACE EYEPOINT IN SAME X-Y PLANE AS POINT TO BE DRAWN . THE EYEPOINT WAS AT Z=K.

COMMENTS:

The output switch settings SHFT, NOSHFT, MOVE/NOMOVE, SCAL/NOSCAL are printed at output time. The VIEWXY, VIEWZX, and VIEWZY switches are also considered, but not printed. The maximum paper size is X=49 and Y=28".
***DRAWL ERROR - ONLY n POINTS/DRAWING means that over n points were requested for the current drawing. The first n points will be drawn and DRAWL execution will proceed. The current upper limit n can be found in the DRAWLNEWS file. Users should realize that it is only when DRAW or another of the output routines is executed that transformations are performed. If a drawing must be scaled to fit onto the output device, the scale factor is printed. DSRN 8 is used by DRAW to build a file of CalComp plotter commands. If DRAW is called, DSRN 8 must not be referenced by the user's program. See the MTS FORTRAN Users' Guide for more information on DSRN's (Data Set

Reference Numbers.)

EXAMPLES:

```
CALL DRAW('OBJ@',&1,&2)
CALL DRAW('ASM@',&999)
```

NAME: SHOW

FUNCTION: Display an entity on a cathode-ray-tube display.

CALLING SEQUENCE:
FORTRAN IV:
CALL SHOW(ENTITY,&1,&2,&3,&4)

ARGUMENTS:
ENTITY - The DRAWL name to be displayed on the CRT.

RETURN CODES:
&1 - ENTITY was undefined when SHOW was called.
***DRAWL ERROR ENTITY NOT FOUND BY SHOW
&2 - ENTITY is a transformation name.
***DRAWL ERROR _ ATTEMPT TO SHOW TRANSFORMATION "ENTITY".
&3 - See DRAW.
&4 - Because NOSCAL or NOSHFT was set, the picture is being drawn off the screen.
***DF - OFF THE SCREEN.

COMMENTS:
Return code four cannot be reached unless SHFT or SCAL were turned off before SHOW was called. SHOW is terminated the first time a point is drawn off the screen. The screen dimensions are 9.375" square. (0,0) is in the lower left-hand corner. DSRN 9 is used by SHOW to transmit display files to the CRT If SHOW is called, DSRN 9 must not be referenced by the user's program. Also see the comments under DRAW.

EXAMPLES:
```
CALL SHOW('PICT@',&1)
CALL SHOW('FIGURE@',&3,&6,&9)
CALL SHOW('FIG1@')
```

NAME: PRTPLT

FUNCTION: Produce a "drawing" on an on-line standard line-printer.

CALLING SEQUENCE:

FORTRAN IV:
CALL PRTPLT(ENTITY,&1,&2)

ARGUMENTS:

ENTITY - the DRAWL name of the entity to be plotted on the line printer.

RETURN CODES: See DRAW, substituting "PRTPLT" for "DRAW".

COMMENTS: These plots are not as accurate or informative as other output modes. Since they are printed along with other printed DRAWL output, they are inexpensive and waits for their production are not required. The plot will be one standard printer page, but true dimensions are not preserved. The plot is scaled to the 28" by 49" standard CalComp (under DRAW) size, and then scaled again to fit the printer's page size. Also see the comments under DRAW.

EXAMPLES:

```
CALL PRTPLT('Z1@')
CALL PRTPLT('DRAW@',&3)
```

NAME: PRTOUT

FUNCTION: Print the switch and (X,Y) values for each point in the entity on the line printer.

CALLING SEQUENCE:

FORTRAN IV:
CALL PRTOUT(ENTITY,&1,&2)

ARGUMENTS:

ENTITY - The DRAWL name of the entity whose points are to be printed.

RETURN CODES: See DRAW, substituting "PRTOUT" for "DRAW"

COMMENTS

This is the least expensive entity output method. Also see the comments for PRTPLT and DRAW. The values are scaled to standard CalComp output size, 28" by 49". Also see the comments under DRAW.

EXAMPLES:

```
CALL PRTOUT('ENT1@',&3)
CALL PRTOUT('FINAL@')
```

NAME: PRTPUT

FUNCTION: Write the switch and (X,Y) values for each point in the entity on a storage device.

CALLING SEQUENCE:

FORTRAN IV
CALL PRTPUT(ENTITY,LDN,&1,&2)

ARGUMENTS

ENTITY - The DRAWL name of the entity whose points are to be saved on a line file, sequential file, magnetic tape, or punched cards.
LDN - The full word integer Logical Device Number (or Data Set Reference) to which the storage device is attached at run time.

RETURN CODES: See DRAW, substituting "PRTPUT" for "DRAW"

COMMENTS: The device must be attached to the LDN (DSRN) at RUN time. If DRAW is called in the program, Data Set Reference Number 8 must not be used by PRTPUT. If SHOW is called, DSRN 9 must not be used. These DSRN's are used by these subroutines for data transmission. Values are scaled to standard CalComp output size: 28" by 49". For further information on DSRN's, see the MTS FORTRAN Users' Guide. Also see the comments under DRAW.

EXAMPLES:

```
CALL PRTPUT('NAME@',3,&5)
CALL PRTPUT('NAME2@',I)
```

NAME: DISPLA

FUNCTION: Produce a drawing of an entity on a ten-inch on-line CalComp plotter attached to a DEC-338 CRT display.

CALLING SEQUENCE:

FORTRAN IV
CALL DISPLA(ENTITY,&1,&2)

ARGUMENTS:

ENTITY - The DRAWL name of the entity to be drawn.

RETURN CODES: See DRAW, substituting "DISPLA" for "DRAW"

COMMENTS: This routine functions only when the user is operating in interactive mode from a terminal equipped with a plotter. The drawing size is 10" by 30". Also see the comments under DRAW.

EXAMPLE:

```
CALL DISPLA('OBJECT@',&1)
```

APPENDIX C

DRAWL Output Parameters

Table C1. Default settings for DRAWL RUN parameters

Parameter		Default value	Page
Drawing Scaling		SCAL	C-2
Drawing Shifting		SHFT	C-3
Paper Advance		MOVE	C-4
Viewing Plane		VIEWXY	C-5
Descriptive Printing	Batch:	PRNT	C-6
	Interactive:	NOPRNT	C-6
Pagination	Batch:	PAGE	C-7
	Interactive:	NOPAGE	C-7
Error Handling:		RETURN	D-3

SCAL
NOSCAL

NAME: SCAL, NOSCAL

FUNCTION: Turn drawing scaling on (SCAL) and off (NOSCAL).

CALLING SEQUENCE: FORTRAN IV:

```
CALL SCAL
CALL NOSCAL
```

DEFAULT SETTING: SCAL is on.

COMMENTS: If an entity is too large to fit onto the output device specified for the drawing, the drawing will be scaled (if SCAL is on) to fit onto the device. The device sizes are

Off-line Cal Comp Plotter
(CALL DRAW):
XMAX=49", YMAX=28"
D.E.C. 338 C.R.T. Display
(CALL SHOW):
XMAX=9.375", YMAX=9.375"
Line printer or file
(CALL PRTPUT, PRTPLT, PRTOUT):
XMAX=49",YMAX=28"
On-line Cal Comp Plotter
(CALL DISPLA):
XMAX=30", YMAX=10"

If NOSCAL is on, and the device size is exceeded, windows of the above sizes are effected, except for SHOW, where the drawing is terminated the first time the device size is exceeded:

**DF-OFF THE SCREEN

Each call to INIT restores the default value. Setting SCAL on turns NOSCAL off. Setting NOSCAL off turns SCAL on.

SHFT
NOSHFT

NAME: SHFT, NOSHFT

FUNCTION: Turn drawing shifting on (SHFT) and off (NOSHFT)

CALLING SEQUENCE:

FORTRAN IV:

```
CALL SHFT
CALL NOSHFT
```

DEFAULT SETTING: SHFT is on.

COMMENTS: If a drawing has any negative coordinate values, it will be shifted onto the positive X-Y plane (if SHFT is on). PRTOUT, PRTPLT, and PRTPUT allow negative values, but also use SHFT as their default setting. Used in conjunction with SCAL, it allows any drawing to be displayed in its entirety. If negative values are sent to a device while NOSHFT is on, a window will be effected at X=0, Y=0, except when SHOW is specified. SHOW will be terminated at the first negative coordinate encountered:

```
**DF-OFF THE SCREEN
```

Each call to INIT restores the default value. Setting SHFT on sets NOSHFT off. Setting NOSHFT on sets SHFT off.

NAME:

MOVE, NOMOVE

FUNCTION:

Allow drawings to be overlaid (NOMOVE is on) or produce the drawing on a fresh output surface (MOVE is on).

CALLING SEQUENCE:

FORTRAN IV:

```
CALL MOVE
CALL NOMOVE
```

DEFAULT SETTING:

MOVE is on

COMMENTS:

MOVE and NOMOVE do not affect PRTOUT, PRTPLT or PRTPUT. Separate drawings are generally scaled and shifted by different values if DRAWL scales or shifts them (see SCAL and SHFT output options.) Users should be sure overlaid drawings are altered by the same amount (or not altered at all). If DRAW is called when NOMOVE is in effect, PLTEND must be called before program termination, or before the next call to MOVE, whichever occurs first. In the example below, A and B will be overlaid, while C will be drawn separately. The default is restored by each call to INIT. Setting MOVE on sets NOMOVE off. Setting NOMOVE on sets MOVE off.

EXAMPLE:

```
CALL NOMOVE
CALL DRAW('A@')
CALL DRAW('B@')
CALL PLTEND
CALL MOVE
CALL DRAW('C@')
```

VIEWXY
VIEWXZ
VIEWZY

NAME:

VIEWXY, VIEWXZ, VIEWZY

FUNCTION:

Set the VIEW SWITCH, which selects front, top, or right-side views for drawings, respectively.

CALLING SEQUENCE:

FORTRAN IV

```
CALL VIEWXY
CALL VIEWXZ
CALL VIEWZY
```

DEFAULT SETTING:

VIEWXY is on.

COMMENTS:

Each call to INIT sets VIEWXY on. The view switch is reset only by calling VIEWXY, VIEWXZ, VIEWZY, or INIT.

EXAMPLE:

To produce all three views of an entity and reset the view switch:

```
CALL DRAW('A@')
CALL VIEWXZ
CALL DRAW('A@')
CALL VIEWZY
CALL DRAW('A@')
CALL VIEWXY
```

If the entity is properly translated between calls to DRAW, three-view drawings can be produced in conjunction with NOMOVE.

NAME: PRNT, NOPRNT

FUNCTION: Turn DRAWL execution time printing on (PRNT) and off (NOPRNT).

CALLING SEQUENCE: FORTRAN IV:

```
CALL PRNT
CALL NOPRNT
```

DEFAULT SETTING: Batch users—PRNT is on.
Terminal users—NOPRNT is on.

COMMENTS: Error comments are not supressed by NOPRNT. Warning comments are supressed. The default is restored by each call to INIT. Setting PRNT on turns NOPRNT off. Setting NOPRNT on turns PRNT off.

PAGE
NOPAGE

NAME: PAGE, NOPAGE

FUNCTION: Turn pagination* on (PAGE) and off (NOPAGE)

CALLING SEQUENCE:

FORTRAN IV
```
CALL PAGE
CALL NOPAGE
```

DEFAULT SETTING:

Batch Users - PAGE is on.
Terminal Users - NOPAGE is on.

COMMENTS:

Batch users who wish to do printing in addition to DRAWL's printing, and who do not want to use LCOUNT to maintain pagination can turn pagination off. Headers will not be printed. Pagination can be restored at any time, but page numbering will be continued from where it left off. Setting PAGE on sets NOPAGE off. Setting NOPAGE on sets PAGE off.

---

* Pagination is the maintenance of a page count by DRAWL, and the printing of a header and current page number at the top of every fresh page printed. The header includes the DRAWL version number, user I.D. number, date, time, and page number.

APPENDIX D

Special DRAWL routines

SPECIAL POINTS
SPECIAL INCRPT

Several special calls for POINTS and INCRPT are provided.

I. Absolute Coordinates

Special POINTS entry:	Equivalent to calling:
VISA(X,Y,Z)	POINTS(1,X,Y,Z,1.)
INVA(X,Y,Z)	POINTS(0,X,Y,Z,1.)
VISA2(X,Y)	POINTS(1,X,Y,0.,1.)
INVA2(X,Y)	POINTS(0,X,Y,0.,1.)

II. Incremental Relative Coordinates

Special INCRPT entry:	Equivalent to calling:
VISR(X,Y,Z)	INCRPT(1,X,Y,Z,1.)
INVR(X,Y,Z)	INCRPT(0,X,Y,Z,1.)
VISR2(X,Y)	INCRPT(1,X,Y,0.,1.)
INVR2(X,Y)	INCRPT(0,X,Y,0.,1.)

The usual return codes and error messages for POINTS and INCRPT also apply to these special calls.

EXAMPLES:

```
CALL VISA(3.,4.,9.,&6)
CALL VISR2(A,12.)
CALL INVA2(6.,7.,&1)
CALL INVR2(2.15,COS(ALPHA),&10)
CALL INVA(10.,A,A)
```

NAME: RETREV

FUNCTION: Print the values of the DRAWL transformation matrix, object, or assembly named as argument.

CALLING SEQUENCE:

FORTRAN IV:

```
CALL RETREV(NAME,&1)
```

ARGUMENTS:

NAME - The DRAWL name of the transformation matrix, object, or assembly to be retrieved.

RETURN CODES:

&1 - An undefined name was used as argument to RETREV:
***DRAWL ERROR-NAME NOT FOUND BY RETR

COMMENTS: RETREV prints the values DRAWL has associate with the DRAWL names used in a run. It is especially useful in debugging.
CALL TDUMP calls RETREV for every DRAWL name in the program. When TDUMP is called just before program termination, a complete traceback through the DRAWL storage hierarchy is provided. TDUMP has no arguments or return codes.

EXAMPLES:

```
CALL RETREV('T1@')
CALL RETREV('ASM@')
CALL TDUMP
```

NAME: ABORT

FUNCTION: Designates the error-handling procedure to follow upon encountering errors.

CALLING SEQUENCE: CALL ABORT(ACTION,N)

ARGUMENTS:

ACTION - The type of action to be taken upon encountering the Nth error. ACTION may be:

'QUIT@' - Sign the job off the system.

'SYSTEM@' - Terminate DRAWL execution but do not sign off.

'MTS@' - Terminate execution, but do do not unload DRAWL. DRAWL may be restarted with $RESTART

'RETURN@' - Continue to use the error exits, continue execution.

N - The number of errors allowed before action is requested.

RETURN CODES: None

DEFAULT VALUES:

ACTION - 'RETURN@'
N = 100

COMMENTS: All errors listed under "DRAWL Error and Warning Messages" are included in the error count. In general, terminal users should not use QUIT, and batch users should not use MTS. If ACTION is an unrecognizable string, RETURN will be assumed:

***DRAWL ERROR - ACTION IS AN ILLEGAL ARGUMENT FOR ABORT. "RETURN" ASSUMED.

EXAMPLES:

```
CALL ABORT('QUIT@',3)
CALL ABORT('SYSTEM@',1)
```

NAME: CMBMAT

FUNCTION: Name and initiate the formation of combined DRAWL transformation matrices.

CALLING SEQUENCE: FORTRAN IV:
CALL CMBMAT(NAME)

ARGUMENTS: NAME - The DRAWL name of the transformation matrix to be formed.

RETURN CODES: None.

COMMENTS: CMBMAT (CoMBine MATrices) is used in conjunction with the single transformation matrix defining functions. Termination of the list of functions is signalled by a call to ENDCMB, which calls NAMTRA. The permissible single transformati matrix defining functions are:

NAME	EFFECT	ARGUMENT
TRLATE(X,Y,Z)	Translation	The X,Y, and Z values (in inches) for the translatio row of the transformation matrix.
CSCALE(W)	Scaling	The sixteenth entry in the transformation matrix.
PERSPC(S)	Perspective	The location on the Z axis (in inches) of the eyepoir
ROTX(R)	Rotation about the X-axis	The amount of rotation (in radians)
ROT1(R)	Rotation about the X-axis	The amount of rotation (in radians)
ROTXD(D)	Rotation about the X-axis	The amount of rotation (in degrees)
ROTY(R)	Rotation about the Y-axis	The amount of rotation (in radians)
ROT2(R)	Rotation about the Y-axis	The amount of rotation (in radians)

ROTYD(D)	Rotation about the Y-axis	The amount of rotation (in degrees)
ROTZ(R)	Rotation about the Z-axis	The amount of rotation (in radians)
ROT3(R)	Rotation about the Z-axis	The amount of rotation (in radians)
ROTZD(D)	Rotation about the Z-axis	The amount of rotation (in degrees)

Each single transformation matrix defining function generates a four-by-four matrix according to the following table:

TRLATE(X,Y,Z)

1.	0.	0.	0.
0.	1.	0.	0.
0.	0.	1.	0.
X	Y	Z	1.

CSCALE(W)

1.	0.	0.	0.
0.	1.	0.	0.
0.	0.	1.	0.
0.	0.	0.	W

ROTX(R), ROT1(R), ROTXD(R*360./6.28)

1.	0.	0.	0.
0.	COS(R)	SIN(R)	0.
0.	-SIN(R)	COS(R)	0.
0.	0.	0.	1.

ROTY(R), ROT2(R), ROTYD(R*360./6.28)

COS(R)	0.	-SIN(R)	0.
0.	1.	0.	0.
SIN(R)	0.	COS(R)	0.
0.	0.	0.	1.

ROTZ(R), ROT3(R), ROTZD(R*360./6.28)

COS(R)	SIN(R)	0.	0.
-SIN(R)	COS(R)	0.	0.
0.	0.	1.	0.
0.	0.	0.	1.

PERSPC(S)

1.	0.	0.	0.
0.	1.	0.	0.
0.	0.	1.	-1./S
0.	0.	0.	1.

CMBMAT names a transformation matrix which is the product of the specified single transformation matrix defining functions' matrices.

The following sequences are equivalent:

(A)
```
CALL CMBMAT('A@')
CALL TRLATE(1.,2.,3.) and
CALL ENDCMB
```

```
CALL NAMTRA('A@',1.,0.,0.,0.,
1                0.,1.,0.,0.,
2                0.,0.,1.,0.,
3                1.,2.,3.,1.)
```

```
(B)  CALL CMBMAT('B@')          CALL NAMTRA('B@',1.,0.,0.,0.,
     CALL CSCALE(5.)          1                1.,0.,0.,
     CALL PERSPC( 10.) and    2                0.,0.,1.,-1/10.,
     CALL ENDCMB              3                0.,0.,0.,5.)
```

To generate a matrix named MAT1 which will

1) Translate three inches in X,
2) Rotate 35 degrees about the X-axis
3) Translate four inches in Y and two inches in Z,
4) Perform a perspective transformation, with the eyepoint at X=0, Y=0, Z=-35 inches,

the following sequence could be used:

```
CALL CMBMAT('MAT1@')
CALL TRLATE(3.,0.,0.,)
CALL ROTXD(35.)
CALL TRLATE(0.,4.,2.)
CALL PERSPC(-35.)
CALL ENDCMB
```

The single transformation matrix defining functions can also be used to implicitly name DRAWL transformation matrices. Using the name and argument of any of the functions as the third argument to TRANSF will name and produce a transformation matrix as specified by the function with the newly generated matrix. The transformation matrices defined in this manner are usable only once, although they are in storage until termination of the run.

For example:

```
     RAD=DEG*6.28/360.
     CALL NAMTRA('ZROT@',COS(RAD),SIN(RAD),0.,0.,
    1                   -SIN(RAD),COS(RAD),0.,0.,
    2                      0.,      0.,    1.,0.,
    3                      0.,      0.,    0.,1.)
     CALL TRANSF('OBJ@','OBJ1@','ZROT@')
```

is equivalent to:

```
     CALL TRANSF('OBJ@','OBJ1@', ROTZD(DEG))
```

Note that the function name is not enclosed in primes.

NAME:

LCOUNT

FUNCTION:

Allow accurate pagination by DRAWL in batch mode when users do printing in addition to the DRAWL processor's.

CALLING SEQUENCE:

FORTRAN IV:

```
CALL LCOUNT
```

ARGUMENTS:

N - The number of lines about to be printed by the user's FORTRAN IV PRINT or WRITE statement (including carriage control).

RETURN CODES:

None

COMMENTS:

DRAWL's pagination algorithm maintains a count of the lines printed per page. When a page has been filled, the counter is reset to zero, a page-eject is given, and the header printed. If there are less than N lines left on the page, a page-eject is given before the printing is done. LCOUNT has no effect if NOPRNT is on (whether explicitly set or by default). At most 56 lines are allowed on a page. CALL LCOUNT(57) gives a page-eject and header before printing.

EXAMPLES:

```
   DO   10   I=1,5
   CALL LCOUNT(1)
   PRINT 17, X(I),Y(I)
17 FORMAT(3X,2F4.1)
 5 CONTINUE
      .
      .
      .
   CALL LCOUNT(2)
   PRINT 99
99 FORMAT('0 TWO LINES PRINTED (COUNTING
  *CARRIAGE CONTROL)')
```

APPENDIX E

DRAWL Error Messages

DRAWL Error Comments

Two classes of error comments are implemented in DRAWL--Warnings and Errors. Many errors which would ultimately cause program termination are intercepted and corrected with "educated guesses." For example, missing decimal points are inserted, and zero scale factors changed to one. Each time such an error is intercepted, "***DRAWL WARNING - ", followed by a description of the error, is printed. When errors are made which cannot be easily second-guessed, "***DRAWL ERROR -", followed by a description of the error, is printed. Whenever a DRAWL name is the cause of an error, it is printed in hexadecimal along with its character representation, so that, for example, the letter O can be distinguished from the digit zero. Note that each error and warning adds to the count maintained by subroutine ABORT (see the ABORT description in the Appendix.)

A. DRAWL WARNINGS

1. SCALE FACTOR OF ZERO. CHANGED TO ONE.
   (NAMTRA,POINTS,NMTRAl,INCRPT,CMBMAT entries)

2. THERE IS NOTHING TO USE AS BASE FOR RELATIVE OR INCREMENTAL CALL
   (0.,0.,0.) ASSUMED.
   (INCRPT,NAMOBR)

3. ILLEGAL CALL TO [FINASM] [FINOBJ] [ENDCMB].
   [FINOBJ] [ENDCMB] [FINASM] CALLED FOR YOU.
   ([FINASM] [FINOBJ] [ENDCMB])

4. UNEXPECTED DECIMAL POINT FOUND. XX.YY CHANGED TO XX
   (POINTS,INCRPT)

5. EXPECTED DECIMAL POINT MISSING. XX CHANGED TO XX.0
   (POINTS,INCRPT,NAMTRA,NMTRAl,CMBMAT entries)

6. FAILURE TO FINISH OBJECT A, FINOBJ CALLED FOR YOU.
   (Any DRAWL routine except POINTS,INCRPT,FINOBJ)

7. FAILURE TO FINISH ASSEMBLY A, FINASM CALLED FOR YOU.
   (Any DRAWL routine except NAME,FINASM)

8. FAILURE TO END CMBMAT A, ENDCMB CALLED FOR YOU.
   (DRAWL routines other than single transformation
   matrix defining functions or ENDCMB.)

9. FAILURE TO CALL INIT.
   (Any DRAWL routine other than INIT.)

10. 'STRING' IS AN ILLEGAL ARGUMENT FOR ABORT.
"RETURN" ASSUMED.
(ABORT)

B. DRAWL ERRORS.

1. OVER 1000 POINTS WERE SPECIFIED. TERMINATING VIA "ABORT" SPECIFICATIC
(POINTS,INCRPT)

2. OVER 200 NAMES WERE SPECIFIED. TERMINATING VIA "ABORT" SPECIFICATION.
(NAMOBJ,NAMASM,NAMTRA,NMTRA1,CMBMAT,TRANSF)

3. OVER 5000 LINKAGES WERE SPECIFIED. TERMINATING VIA "ABORT"
SPECIFICATION.
(NAMTRA,NAMOBJ,TRANSF,POINTS,INCRPT,NMTRA1)

4. POINTS CALLED WITH NO NAMED OBJECT VIA NAMOBJ.
(POINTS,INCRPT)

5. 'STRING' ALREADY USED AS OBJ/ASM NAME.
(NAMTRA,NMTRA1,CMBMAT)

6. NAME CALLED WITH NO NAMED ASSEMBLY VIA NAMASM.
(NAME)

7. TRANSFORMATION 'STRING' USED AS ARGUMENT TO NAME.
(NAME)

8. 'STRING' NOT FOUND BY NAME.
(NAME)

9. 'STRING' IS NOT A TRANSFORMATION NAME.
(TRANSF)

10. 'STRING' ALREADY USED AS [OBJECT] [ASSEMBLY] [XFORM] NAME.
(TRANSF)

11. 'STRING' IS A TRANSFORMATION NAME.
(TRANSF)

12. 'STRING' NOT FOUND BY TRANSF.
(TRANSF)

13. ATTEMPT TO [DRAW] [SHOW] [PRTPLT] TRANSFORMATION 'STRING'.
([DRAW] [SHOW] [PRTPLT])

14. 'STRING' NOT FOUND BY [SHOW] [DRAW] [PRTPLT]
([SHOW] [DRAW] [PRTPLT])

15. ONLY 2730 POINTS/DRAWING ALLOWED.
([SHOW] [DRAW] [PRTPLT])

16. ATTEMPT TO PLACE EYEPOINT IN SAME X-Y PLANE AS POINT TO BE DRAWN.
THE EYEPOINT WAS AT Z=K.
([SHOW] [DRAW] [PRTPLT])

17. 'STRING' NOT FOUND BE RETREV.
(RETREV,TDUMP)

18. 'STRING' ALREADY USED AS DRAWL NAME.
(NAMASM,NAMOBJ,NAMTRA,CMBMAT)

APPENDIX F

Sample DRAWL Programs

```
C  THIS DRAWL PROGRAM PRODUCES A DRAWING OF A RECTANGULAR
C  PARALLELOPIPED (AS IN FIG. 2)
      CALL INIT
C  CREATE AN OBJECT
      CALL NAMOBJ('RECTPARa')
C  MOVE THE PEN TO THE ORIGIN
      CALL POINTS(0,0.,0.,0.,1.)
C  DRAW LINES 1-8 (AS IN FIG. 2)
      CALL POINTS(1,4.,0.,0.,1.)
      CALL POINTS(1,5.,2.,0.,1.)
      CALL POINTS(1,1.,2.,0.,1.)
      CALL POINTS(1,0.,0.,0.,1.)
      CALL POINTS(1,0.,0.,8.,1.)
      CALL POINTS(1,1.,2.,8.,1.)
      CALL POINTS(1,5.,2.,8.,1.)
      CALL POINTS(1,5.,2.,0.,1.)
C  LINE 9 (FIG. 2) IS "INVISIBLE" - THE SWITCH IS ZERO
      CALL POINTS(0,4.,0.,0.,1.)
      CALL POINTS(1,4.,0.,8.,1.)
      CALL POINTS(1,5.,2.,8.,1.)
      CALL POINTS(0,4.,0.,8.,1.)
      CALL POINTS(1,0.,0.,8.,1.)
      CALL POINTS(0,1.,2.,8.,1.)
      CALL POINTS(1,1.,2.,0.,1.)
C  FINISH THE OBJECT
      CALL FINOBJ
C  DRAW IT. SEE FIG. F-1
      CALL DRAW('RECTPARa')
C  THAT WAS THE X-Y PROJECTION OF THE OBJECT ORIENTED IN ITS
C  ORIGINAL POSITION. NOW LET'S ORIENT IT AS IN FIG. 2
C
C  ROTATE IT ABOUT THE Y-AXIS. USING FOUR CARDS FOR THE
C  FORTRAN CALL STATEMENT MAKES THE MATRIX EASIER TO READ
C  THAN IF IT WAS ALL ON ONE CARD. NOTE THAT FORTRAN'S SIN AND COS
C  FUNCTIONS USE RADIANS AND NOT DEGREES.
      RADS30=(6.28/360.)*30.
      CALL MAMTRA('RYa', COS(RADS30),     0.     ,-SIN(RADS30),  0.,
     1                     0.     ,     1.     ,      0.     ,  0.,
     2                SIN(RADS30),     0.     , COS(RADS30),  0.,
     3                     0.     ,     0.     ,      0.     ,  1. )
      CALL TRANSF('RECTPARa','RECTPARa','RYa')
      RADS45=(6.28/360.)*45.
C  NOW ROTATE THE ROTATED OBJECT ABOUT THE X-AXIS
      CALL MAMTRA('RXa',      1.     ,     0.     ,      0.     ,  0.,
     1                     0.     , COS(RADS45), SIN(RADS45),  0.,
     2                     0.     ,-SIN(RADS45), COS(RADS45),  0.,
     3                     0.     ,     0.     ,      0      ,  1. )
      CALL TRANSF('RECTPARa','RECTPARa','RXa')
C  DRAW THE TWICE ROTATED OBJECT (SEE FIG. F-2)
      CALL DRAW('RECTPARa')
      STOP
      END
```

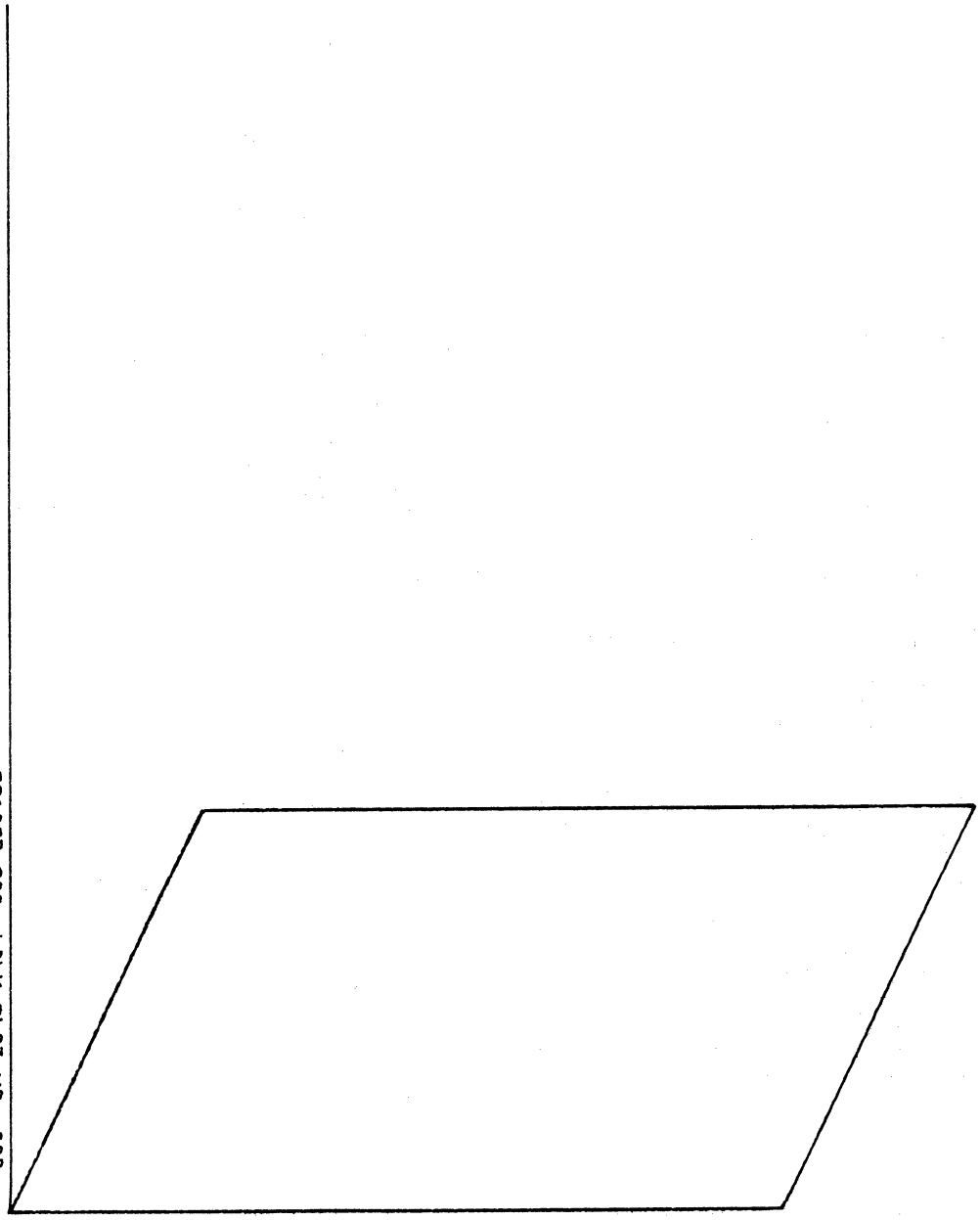


Figure F-1

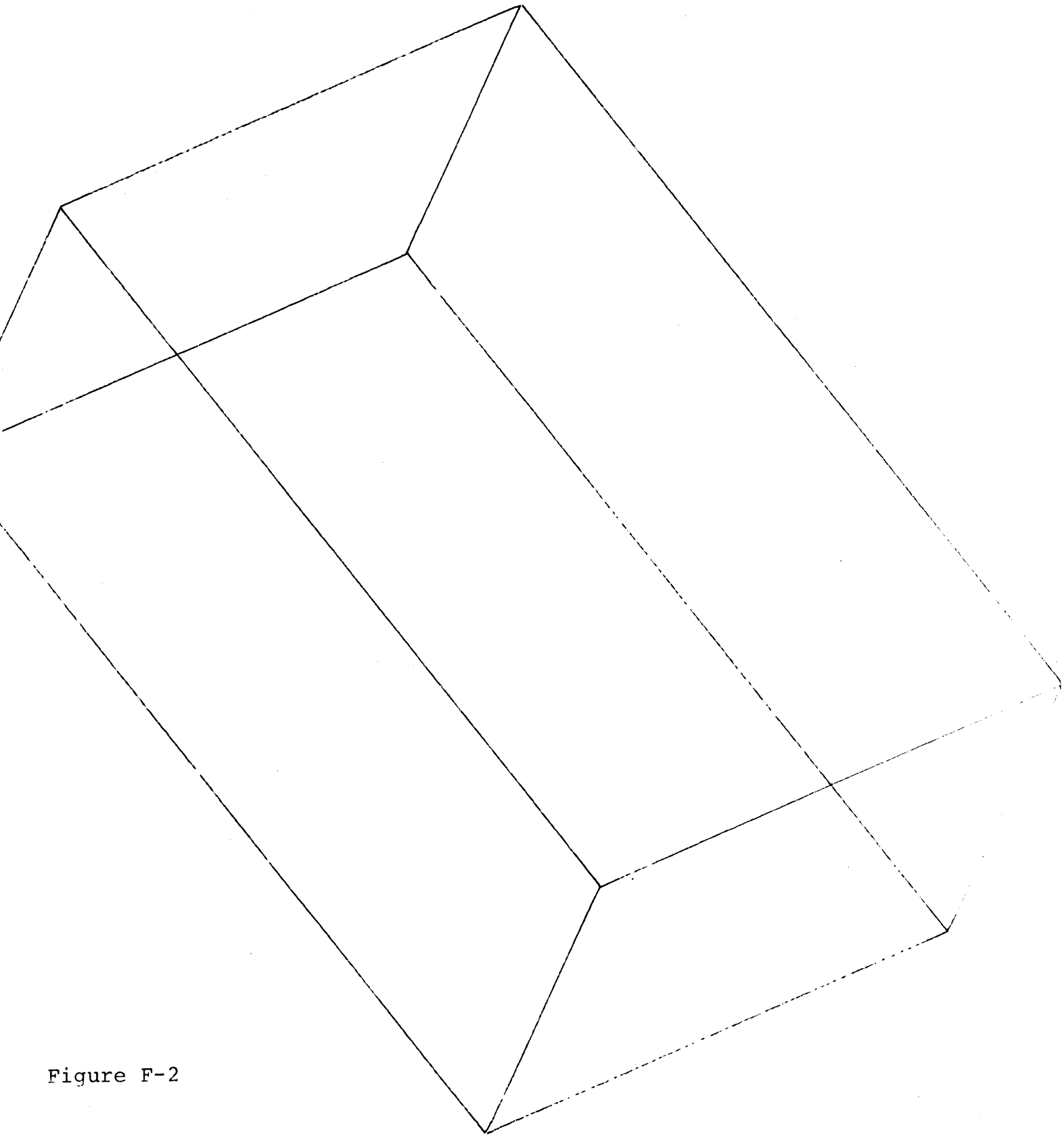


Figure F-2

```
C  USING THE FORTRAN IV DO STATEMENT, WE CAN DRAW A
C  CIRCLE: (X-2)**2+(Y-4)**2=9
      XC=2.
      YC=4.
      R=3.
C  WE WILL USE 30 LINE SEGMENTS= 31 POINTS.
      T=6.28/30.0
      TS=-T
      CALL INIT
      CALL NAMOBJ('CIRCa')
C  MOVE PEN TO X=5, Y=7
      CALL INVA2(5.,4.)
      DO 1000 I=1,31
      TS=TS+T
 1000 CALL POINTS(1,XC+R*COS(TS),YC+R*SIN(TS),0.,1.)
      CALL FINOBJ
C  NOW A SPIRAL
      CALL NAMOBJ('SPIRALa')
      CALL POINTS(0,5.,4.,0.,1.)
C  LET'S MAKE FOUR LOOPS, TWO INCHES APART.
      DELZ=2./30.
      Z1=-DELZ
      DO 1010 I=1,4
      TS=-T
      DO 1010 J=1,31
      Z1=Z1+DELZ
      TS=TS+T
 1010 CALL VISA(XC+R*COS(TS),YC+R*SIN(TS),Z1+DELZ,1.)
      CALL FINOBJ
C  LET'S PUT CIRCLES ON THE ENDS OF THE SPIRAL AS ENDCAPS.
C  THE FIRST ONE IS IN PLACE AT X=2, Y=4, Z=0. LET'S MOVE
C  A COPY TO Z=6.
      CALL NAMTRA('ZMOVEa',1.,0.,0.,0.,
     1                     0.,1.,0.,0.,
     2                     0.,0.,1.,0.,
     3                     0.,0.,6.,1.)
C  PERFORM THE TRANSFORMATION
      CALL TRANSF('NEWCIRCa','CIRCa','ZMOVEa')
C  PUT IT ALL TOGETHER INTO AN ASSEMBLY
      CALL NAMASM('OUTPUTa')
      CALL NAME('CIRCa')
      CALL NAME('SPIRALa')
      CALL NAME('NEWCIRCa')
      CALL FINASM
C  SCALE IT TO FIT ON 8 BY 10 PAGE, AND ROTATE IT
      CALL TRANSF('OUTPUTa','OUTPUTa',ROTY(-15))
      CALL TRANSF('OUTPUT1a','OUTPUTa',SCALE(2.))
C  ( SEE PAGE 6-4 FOR AN EXPLANATION OF THE ABOVE COMMANDS)
C  NOW WE'LL TAKE A PERSPECTIVE VIEW, EYEPOINT AT (0,0,-20).
C  TO PLACE THE EYEPOINT  AT Z=-20., WE ENTER -(-1./20.)
C  (SEE APPENDIX A).
      CALL NAMTRA('PERa',1.,0.,0.,0.,
     1                   0.,1.,0.,0.,
     2                   0.,0.,1.,-1./20.,
     3                   0.,0.,0.,1.)
      CALL TRANSF('OUTPUT1a','OUTPUT1a','PERa')
C  AND SO WE ARE READY TO DRAW IT. (SEE FIG. 6-3).
      CALL DRAW('OUTPUT1a')
      STOP
```

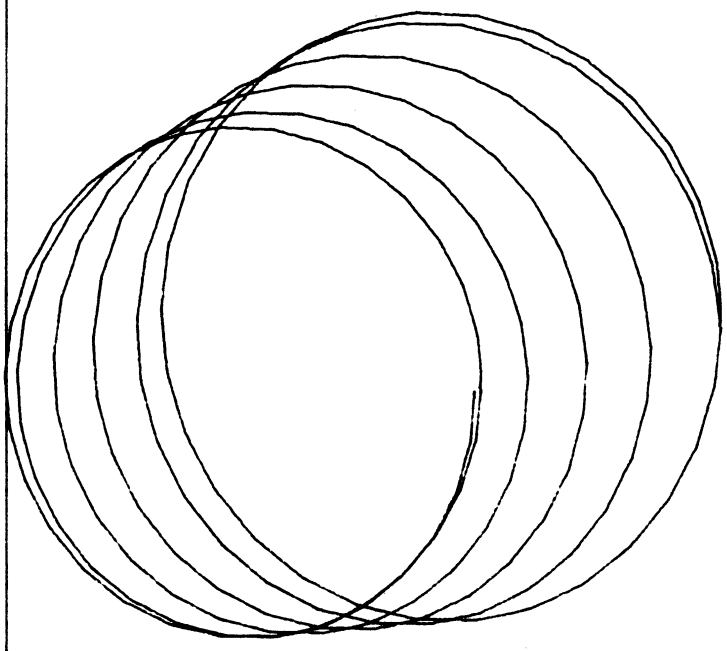


Figure F-3. DRAWL Output.

APPENDIX G

The DRAWL 70 Data Structure

DRAWL 70 is written in FORTRAN IV for the IBM 360/67 computer (see Appendix 4 for a listing of the FORTRAN source code). (In the following description, it will be assumed that the reader can follow the FORTRAN code while referring to Figures G1 and G2.) To the user, DRAWL seems like a tree-type data structure (Figure 3). Internally, DRAWL 70 is a two-way linked list, implemented in four logical tables: a points table, linkage table, transformation table (physically combined with the linkage table) and a directory of DRAWL names.

An object (see Figure G1) consists of an entry in the name table (DIREC) with a pointer (IDIREC) to the linkage table (TABLE). A type is assigned with each entry in DIREC-object is type one. (The 360/67 is a drum-paging machine - that is, pages of core storage are swapped onto and off of high-speed drums. If DIREC and IDIREC were physically separate areas in core, one could be paged out without the other. The FORTRAN IV "EQUIVALENCE" statement allows them to be overlaid. Thus if the page containing DIREC is paged in, IDIREC is automatically brought in with it. The same scheme is used for POINT and IPOINT, and TABLE and ITABLE. This scheme may not be as efficient or desirable on other computers.) After an object has been named via NAMOBJ, points are created with pointers in ITABLE to POINT and IPOINT. The actual X, Y, Z, W values are stored in POINT, and the line-type switch is stored in ITABLE.

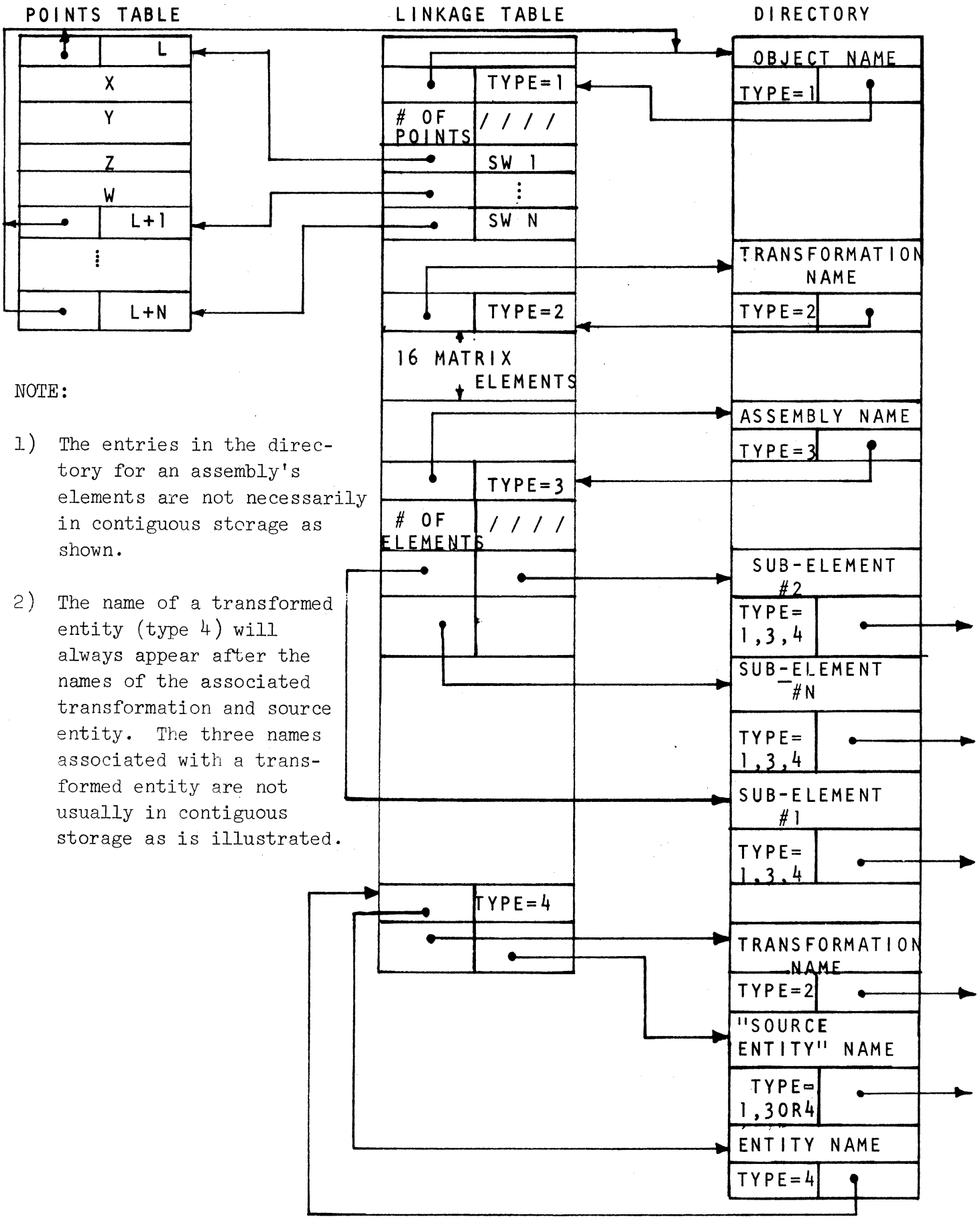
Assemblies are similar to objects in construction. The name is recorded, the type (assembly type = 3) noted, and as NAME is called, pointers are placed in ITABLE. Instead of pointing into POINT, however, they point to the location in DIREC of the entities named in successive calls to NAME. FINASM records the entity count in ITABLE. Transformations (type = 2) have pointers into ITABLE. Their sixteen values are stored in TABLE. Their ITABLE entries are only types and back-pointers into DIREC. The fourth DRAWL entity type is the transformed assembly or object (type = 4). The name and type are recorded in DIREC, and two pointers are placed in ITABLE — the locations in DIREC of:

1) The source entity (second argument to TRANSF),
2) The transformation being used (third argument to TRANSF).

The usual back-pointer from ITABLE to DIREC is also maintained.

Figure G2A shows TABLE and DIREC after the following sequence has been executed:

```
CALL NAMOBJ('Q@')
CALL POINTS(SW1, . . .)
CALL POINTS(SW2, . . .)
   .
   .
   .
```

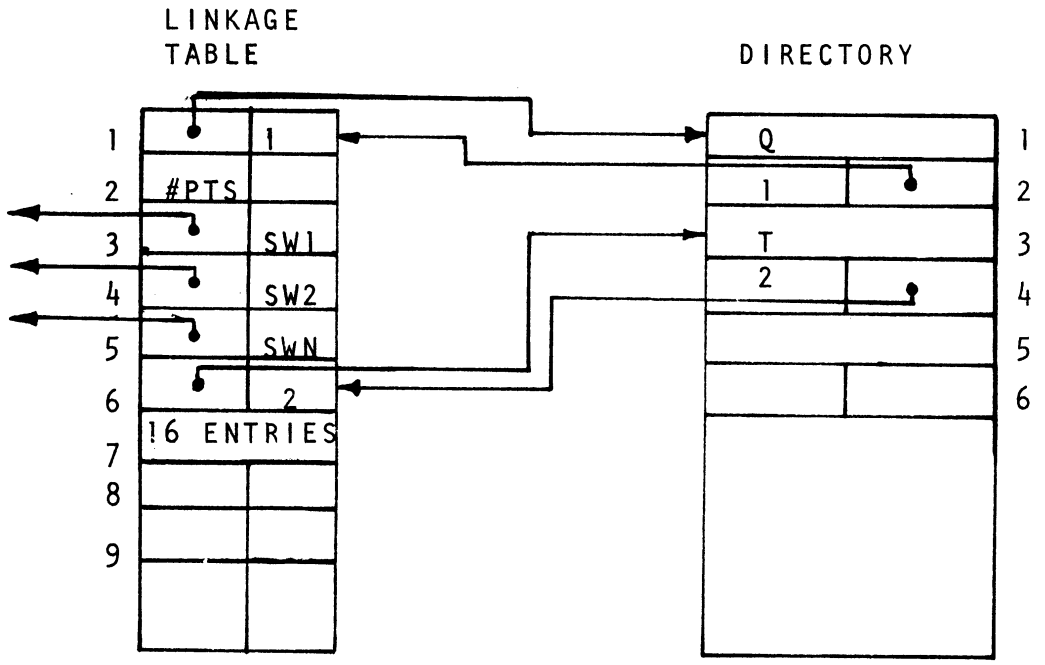


NOTE:

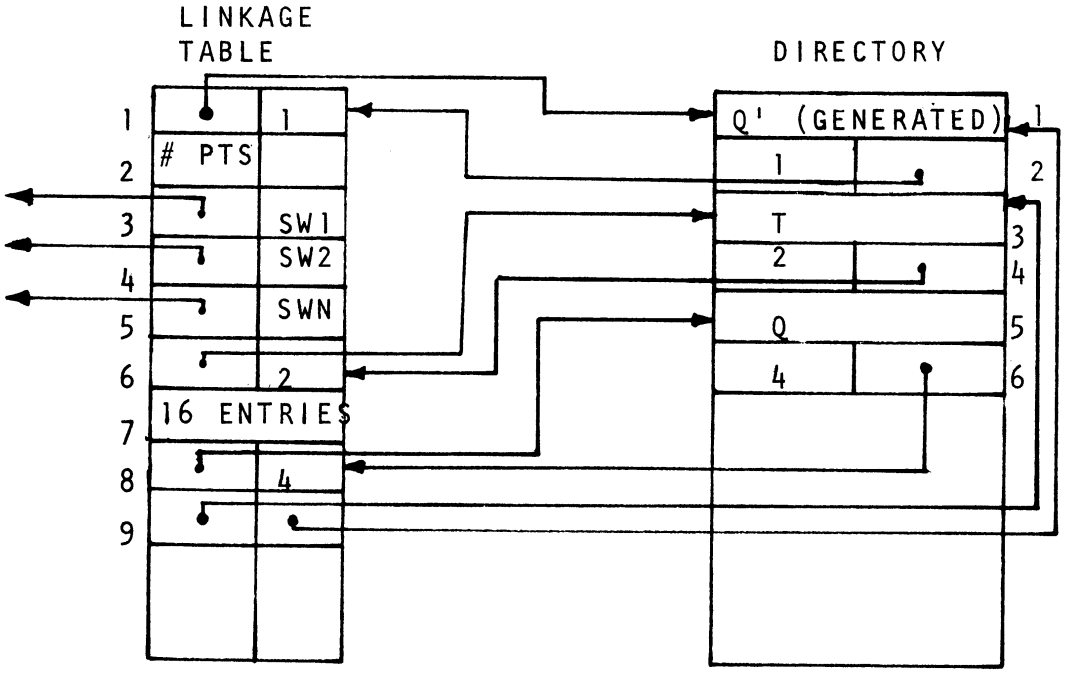
1) The entries in the directory for an assembly's elements are not necessarily in contiguous storage as shown.

2) The name of a transformed entity (type 4) will always appear after the names of the associated transformation and source entity. The three names associated with a transformed entity are not usually in contiguous storage as is illustrated.

Figure G1. DRAWL Data Structure.



(a)



(b)

Figure G-2. Internal Operation of TRANSF.

```
      .
      .
CALL POINTS(SWN, . . .)
CALL FINOBJ
CALL NAMTRA('T@',A1, . . .,A16)
```

Execution of the DRAWL command:

```
CALL TRANSF('Q@','Q@','T@')
```

changes the system to appear as in Figure G2B."Q" now refers to a transformed object, and all references to it in the future will cause the transformed entity to be used. There is no explicit way to refer to the values originally associated with "Q". (The reader is encouraged to work his way through the above commands with Appendix H, working out all the subscript computations).

APPENDIX H

DRAWL 70 Listing

```
      BLOCK DATA
C       ATABL CONTAINS THE POINT, NAME, AND LINKAGE TABLES
C       CURRENTLY, THE LINKAGE TABLE ALSO CONTAINS THE VALUES
C       FOR TRANSFORMATION MATRICES AND SWITCH SETTINGS FOR POINTS
      COMMON /ATABL/ IDIREC(800),POINT(5000),TABLE(5000)
C
C       VAR CONTAINS PARAMETERS FOR THE TABLES IN ATABL
C       PTMAX IS THE MAXIMUM NUMBER OF POINTS ALLOWED
C       PTPTR IS THE INDEX FOR ARRAY POINT
C       TABMAX IS THE UPPER DIMENSION FOR ARRAY TABLE
C       TABPTR IS THE INDEX INTO TABLE
C       DIRMAX IS THE UPPER LIMIT FOR ARRAY DIREC
C       DIRPTR IS THE INDEX INTO DIREC
C       PTCTR IS THE NUMBER OF POINTS IN AN OBJECT DEFINITION
C       PT IS AN INTERIM COUNTER FOR POINT DEFINITION
C       ITP IS ALSO AN INTERIM COUNTER FOR POINT DEFINITION
C       TPTCTR COUNTS THE TOTAL NUMBER OF POINTS DEFINED IN A RUN
C       OPEN IS USED TO CHECK THAT ENTITIES, ONCE OPENED, ARE
C          CLOSED, AND THAT INIT IS CALLED
C       NASM COUNTS THE NUMBER OF ENTITIES INCLUDED IN AN
C         ASSEMBLY DEFINITION UNDER NAMASM
C       GENNAM IS THE CHARACTER STRING USED FOR GENERATED
C         NAMES FOR USE BY TRANSF
C       NSCAL IS THE GLOBAL SWITCH SET BY SCAL AND NOSCALE
C       NMOVE IS THE GLOBAL SWITCH SET BY MOVE AND NOMOVE
C       NSHFT IS THE GLOBAL SWITCH SET BY SHFT AND NOSHFT
C       LNM KEEPS TRACK OF THE NUMBER OF DRAWINGS PER RUN
C       NPRT IS THE GLOBAL PARAMETER SET BY PRNT AND NOPRNT
C       BATCH IS THE GLOBAL SWITCH SET BY PAGE AND NOPAGE
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
C       CLSE HAS TO DO WITH ERROR CHECKING IN SUBROUTINE CLOSE
C         AND FINISHES THE CMBMAT...ENDCMB SEQUENCE BY CALLING
C         NMTRAL WITH TO AS ARGUMENT
      COMMON /CLSE/TO(4,4),XNAME,ISW
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM,XNAME
      DATA PTPTR,TABPTR,DIRPTR/2,3,2/,TPTCTR/1/,OPEN/4/,ISW/0/
      DATA PTMAX,TABMAX,DIRMAX/5000,10000,400/,GENNAM/'%GNAM000'/
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      DATA NSCAL,NMOVE,NSHFT,NPRT/.FALSE.,.TRUE.,.FALSE.,.TRUE./
C       IN THIS IMPLEMENTATION, TABLE AND ITABLE ARE OVERLAID;
C         DIREC AND DIREC; AND TABLE AND ITABLE. ON DIFFERENT
C         MACHINES THIS IS VARIABLY EFFICIENT.
      EQUIVALENCE (IPOINT(1),POINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
      END
C
C   INIT
C
      SUBROUTINE INIT
C       INIT INITIALIZES ALL VARIABLES EXCEPT OPEN (WHICH IS
C       SET AT LOAD TIME.) IT DOES NOT ZERO OUT ANY TABLES, BUT
C       MERELY RESETS ALL THEIR POINTERS. GLOBAL VARIABLES ARE
C       ALSO RESET VIA A CALL TO 'SPEC'.
      COMMON /ATABL/ IDIREC(800), POINT(5000),TABLE(5000)
```

```
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
C       DRWCOM ('DRAW COMMON') IS AN AREA USED FOR COMMUNICATION
C       BETWEEN THE THREE PIECES OF THE OUTPUT ROUTINES. IT PASSES
C       LIMITS AND HAS THE OUTPUT ARRAYS.
      COMMON /DRWCOM/DVEC(50),CURMAT(50),EXPIRE(50),ID,IC,
     1  NPTS,ICMAX,ICMAX,XMIN,XMAX,YMIN,YMAX,I(5460),IT(2730),IL
      INTEGER*2 DVEC,CURMAT,EXPIRE,IC,IC,IT,LEN,LINE(48)
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR,ADROF,ARRAY(5)
      EXTERNAL GETFD,FREEFD
      REAL*8 GENNAM,PAD,NGNAM/'%GNAMCOC'/,DIREC(400)
      EQUIVALENCE (IPOINT(1),POINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       SPEC SETS MANY OF THE GLOBAL SWITCHES TO THEIR DEFAULT
C       VALUES. IT IS IMPLEMENTED IN THIS MANNER SO THAT THOSE
C       WHO DO NOT WISH TO IMPLEMENT THESE FEATURES CAN SPLIT
C       THEM OFF EASILY.
      CALL SPEC
C       CREPLY DETERMINES WHETHER A USER IS IN CONVERSATIONAL
C       MODE (THE BRANCH TO 1198 IS NOT TAKEN, AND PAGINATION
C       AND TRACE PRINTING ARE SUPRESSED) OR BATCH MODE (THE
C       BRANCH TO 1198 IS TAKEN)
      CALL CREPLY(&1198)
      BATCH=.FALSE.
      NPRT=.FALSE.
      GO TO 1199
 1198 BATCH=.TRUE.
      NPRT=.TRUE.
C       IF THE USER IS ON A TERMINAL, HE WILL GET THIS ONE HEADER
C       ONLY ONCE.
 1199 IF(BATCH)GO TO 1254
      PRINT 1200
 1200 FORMAT('- * * * DRAWL - AUG 10, 1970 VERSION  * * *')
      PRINT 1202
 1202 FORMAT('USERS WHO HAVE NOT COPIED SAVE:DRAWLNEWS SINCE')
      PRINT 1203
 1203 FORMAT(' AUG 10, 1970 SHOULD   $COPY SAVE:DRAWLNEWS')
C       IDTIME RETURNS THE USER I.D. NUMBER IN ARRAY(5), THE TIME
C       OF DAY IN ARRAY(2) AND (3), AND THE DATE IN ARRAY(4) AND (5)
      CALL IDTIME(ARRAY)
      PRINT 1201,ARRAY(5),(ARRAY(I),I=1,4)
 1201 FORMAT('0USER ',A4,' INITIATED DRAWL AT ',2A4,' ON ',2A4//)
      GO TO 1256
C       CHKFIL CHECKS TO SEE WHETHER THE ABBREVIATED NEWS FILE WHICH
C       IS SUPPOSED TO BE PRINTED IS AVAILABLE. IF NOT, THE BRANCH
C       TO 1258 IS TAKEN.
 1254 CALL CHKFIL('SAVE:NEWS',&1258)
C       RCALL AND GETFD "GET" THE FILE, WHICH CAN BE READ BY
C       SUBROUTINE READ AND PRINTED BY SUBROUTINE SPRINT.
      CALL RCALL(GETFD,2,0,ADROF('SAVE:NEWS '),1,IFDUB)
C       LCOUNT(57) FORCES A PAGE EJECT.
      CALL LCOUNT(57)
 1250 CALL READ(LINE,LEN,0,IDUM,IFDUB,&1255)
      CALL SPRINT(LINE,LEN,0)
      GO TO 1250
C       RCALL AND FREEFD 'FREE UP' THE NEWS FILE
```

```
 1255 CALL RCALL(FREEFC,1,IFCUB,0)
C       ANOTHER PAGE EJECT. THE NEWS LETTER IS ON A PRIVATE PAGE
 1258 CALL LCCUNT(57)
C       SET ALL THE DEFAULT VALUES:
 1256 PTPTR=2
      PTMAX=4995
      TABMAX=9990
      DIRMAX=398
      TABPTR = 3
      DIRPTR=2
      LNM=0
      IFLG=0
      TPTCTR = 1
      OPEN=0
      GENNAM=NGNAM
      CALL VIEWXY
      T(1)=0.
      T(2)=0.
      IT(1)=0
      NMOVE=.TRUE.
      NSHFT=.FALSE.
      NSCAL=.FALSE.
      RETURN
C
C     TDUMP  TRACEBACK FACILITY
C
      ENTRY TDUMP
C       TDUMP CALLS RETREV FOR EACH ENTRY IN THE NAME TABLE.
C       IT PROVIDES A THOROUGH LINK-BACK FOR DEBUGGING AND
C       INSTRUCTIONAL PURPOSES.
      IF(BATCH)CALL LCCUNT(57)
      IF(BATCH)CALL LCCUNT(2)
      PRINT 780
 780  FORMAT(' DRAWL  TRACEBACK FACILITY'/' ')
      ILIM=DIRPTR-2
      DO 783 I=2,ILIM,2
 783  CALL RETREV(DIREC(I))
      RETURN
      END
C
C   SPECIAL ROUTINES
C
      SUBROUTINE SPEC
C       SPEC SETS THE DEFAULT GLOBAL SWITCHES WHICH MAKE
C       LIFE A LITTLE EASIER
      COMMON /DATAL/ IDIREC(800), PCINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTOTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH,LRET/.FALSE./
      INTEGER*2 IPCINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 OPEN,DIRMAX,DIRPTR,PTOTR,PT,TPTCTR,ERRCNT(30)/30*0/,
     1 ARRAY(5), NUMCNT(3)/1000,200,5000/,PAGES/0/,NLINE/2/
      REAL*8 ERRTYP(3)/' POINTS ',' NAMES  ','LINKAGES'/
      REAL*8 ARG,GENNAM,PAD,TITLE(4)/'RETURN  ',
     1 'QUIT    ','SYSTEM  ','MTS     '/,DIREC(400)
      EQUIVALENCE (IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       'PAGES' COUNTS THE PAGES PRINTED, NLINE IS THE CURRENT
```

```
C       LINECOUNT FOR THIS PAGE, LEVABT IS THE NUMBER OF ERRORS
C       TO TOLERATE BEFORE STOPPING THE JOB,LERCNT IS THE
C       NUMBER OF ERRORS (AND WARNINGS) ENCOUNTERED THUS FAR,
C       LABORT IS THE KEY SET FOR TYPE OF TERMINATION REQUESTED.
      PAGES=0
      NLINE=2
      LEVABT=100
      LERCNT=0
      LABORT=1
      GO TO 1
C       NOSCAL, NOMOVE, AND NOSHFT CONTROL DRAWING SCALING,MOVING,
C       AND SHIFTING, IN CONJUNCTION WITH SCAL, MOVE, AND SHFT.
C       PRNT AND NOPRNT CONTROL WHETHER TRACE COMMENTS ARE TO BE
C       PRINTED OR NOT. PAGE AND NOPAGE TURN PAGINATION
C       ON AND OFF.
      ENTRY NOSCAL
      NSCAL=.TRUE.
      GO TO 1
      ENTRY NOMOVE
      NMOVE=.FALSE.
      GO TO 1
      ENTRY NOSHFT
      NSHFT=.TRUE.
      GO TO 1
      ENTRY SCAL
      NSCAL=.FALSE.
      GO TO 1
      ENTRY MOVE
      NMOVE=.TRUE.
      GO TO 1
      ENTRY SHFT
      NSHFT=.FALSE.
      GO TO 1
      ENTRY PRNT
      NPRT=.TRUE.
      GO TO 1
      ENTRY PAGE
      BATCH=.TRUE.
      GO TO 1
      ENTRY NOPAGE
      BATCH=.FALSE.
      GO TO 1
      ENTRY NOPRNT
      NPRT=.FALSE.
      GO TO 1
C       ABORT SETS LABORT TO THE TYPE OF EXIT TO BE TAKEN ON
C       ENCOUNTERING THE N-TH ERROR (OR WARNING).
C       EXIT IS VIA ROUTINES 'SYSTEM' (STOP AND UNLOAD DRAWL),
C       'MTS' (STOP BUT DO NOT UNLOAD DRAWL), AND 'QUIT' (STOP
C       DRAWL AND IMMEDIATELY SIGN-OFF THE SYSTEM. DO NOT FINISH
C       THE JOB.)
      ENTRY ABORT(ARG,N)
      LERCNT=0
      LRET=.FALSE.
      DO 10 I=1,4
      IF(PAD(ARG).EQ.TITLE(I))GO TO 12
   10 CONTINUE
      PRINT 11,ARG
   11 FORMAT(' ***DRAWL WARNING - ''',A8,''' IS AN ILLEGAL ARGUMENT '/
```

```
     1'    FOR ABCRT. "RETURN" ASSUMED.')
      LABORT=1
      GO TO 13
 12   LABORT=I
 13   LEVABT=N
      IF(I.NE.1)LRET=.TRUE.
      GO TO 1
C       ABORTZ COUNTS (WITH LERCNT) THE NUMBER OF ERRORS AND
C       WARNINGS ENCOUNTERED.
      ENTRY ABORTZ(K)
      IF(K.GT.C)GO TO 8
      PRINT 7,NUMCNT(K+4),ERRTYP(K+4)
 7    FORMAT('0***DRAWL ERROR - OVER ',I4,' ',A8,' WERE SPECIFIED.'/
     14X,'TERMINATING VIA "ABORT" SPECIFICATION')
      GO TO 6
 8    LERCNT=LERCNT+1
      ERRCNT(K)=ERRCNT(K)+1
      IF(LERCNT.LE.LEVABT) GO TO 1
      PRINT 5,ERRCNT
C       ULTIMATELY, DRAWL WILL SENSE IMPENDING PROGRAM TERM-
C       INATION AND PRINT THE ERROR SUMMARY FOR EVERYONE, NOT
C       JUST THOSE WHO ARE FORCING THEMSELVES TO STOP.
 5    FORMAT('0DRAWL ERROR SUMMARY:'/3013)
 6    IF(LABORT.NE.1)IF(LABORT-3)2,3,4
      GO TO 1
 2    CALL QUIT
 3    CALL SYSTEM
 4    CALL MTS
      GO TO 1
C       LCOUNT ('LINE COUNT') MAINTAINS A COUNT OF THE NUMBER
C       OF LINES PRINTED ON EACH PAGE. IF THE TOTAL WILL BE FORCED
C       OVER 57 IF THE NEXT PRINT IS EXECUTED, A HEADER IS PRINTED,
C       AND THE COUNT RESTARTED. 'PAGES' KEEPS TRACK OF THE PAGE
C       NUMBER,WHILE IDTIME GETS THE TIME OF DAY FOR EACH PAGE.
      ENTRY LCOUNT(M)
      N=M
      IF(NLINE+N.GT.58)GO TO 58
      NLINE=NLINE+N
      GO TO 1
 58   CALL IDTIME(ARRAY)
      PAGES=PAGES+1
      PRINT 59,ARRAY(5),(ARRAY(I),I=1,4),PAGES
      IF(N.EQ.57)N=0
      NLINE=2+N
      PRINT 56
 56   FORMAT('C')
 59   FORMAT('1      DRAWL',8X,'G-LEVEL PROCESSOR (MODEL 08:01)',6X,
     1 'USER ',A4,6X,'TIME ',2A4,6X,'DATE ',2A4,12X,'PAGE ',I3)
 1    RETURN
      END
C
C   NAMOBJ
C
      SUBROUTINE NAMOBJ(OBJNAM,*)
      COMMON /ATABL/ IDIREC(800), POINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,IPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
```

```
      INTEGER*2 IPCINT(10CCC),ITABLE(10CCC),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(4CC),OBJNAM,GENNAM,PAC
      EQUIVALENCE(IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     *(DIREC(1),IDIREC(1))
C       ISW IS SET TO ZERO FOR ENTRY FROM NAMCBJ.
      ISW=0
      GO TO 14
      ENTRY NAMCBR(OBJNAM)
C       ISW IS SET TO ONE FOR ENTRY FROM NAMOBR
      ISW=1
      GO TO 14
      ENTRY NAMCBA(OBJNAM)
C       ISW IS SET TO MINUS ONE FOR ENTRY FROM NAMOBA
      ISW=-1
C       CHECK TO SEE IF THIS CALL WAS LEGALLY PLACED (OPEN=0)
C       IF NOT HAVE SUBROUTINE CLOSE STRAIGHTEN THINGS OUT
 14   IF(OPEN.NE.0)CALL CLOSE
C       SET 'OBJECT DEFINITION UNDERWAY' KEY
      OPEN=1
C        CHECK TO SEE IF NAME OK. IF NOT, RETURN 1
      CALL FNDNAM(IFINC,PAD(OBJNAM),ITYP,&15)
      DIREC(DIRPTR)=PAD(OBJNAM)
      PRINT 16, DIREC(DIRPTR)
 16   FORMAT('0***DRAWL ERROR - ''',A8,''' ALREADY  USED AS DRAWL NAME')
      CALL ABORTZ(3)
      RETURN 1
C       PUT NAME (PADDED) INTO NAME TABLE
 15   DIREC(DIRPTR)=PAD(OBJNAM)
C       SET OBJECT TYPE IN NAME TABLE
      IDIREC(DIRPTR*2+1)=1
C       POINTER TO LINKAGE TABLE
      IDIREC(DIRPTR*2+2)=TABPTR+1
C       OBJECT TYPE IN LINKAGE TABLE
      ITABLE(TABPTR+1)=1
C       BACK POINTER TO NAME TABLE
      ITABLE(TABPTR)=DIRPTR
C       PRINT ENTRY MESSAGE
      IF(BATCH)CALL LCOUNT(2)
      IF(NPRT)PRINT 1CC,DIREC(DIRPTR)
 100  FORMAT('0ENTRY TO NAMOBJ(''',A8,2H'))
C       UPDATE NAME TABLE INDEX
      DIRPTR=DIRPTR+2
C       IF OVERFLOW IS HERE, GET OUT VIA USERS SPECIFICATION.
      IF(DIRPTR.GT.DIRMAX)CALL ABORTZ(-2)
C       INITIALIZE OBJECT POINT COUNTER
      PTCTR=0
C       ESTABLISH INTERIM COUNTER FOR LINKAGE TABLE
      ITP=TABPTR+2
      IF(ISW)3C,4C,5C
C       FOR CALL NAMCBA
 30   CALL POINTS(0,C.,0.,0.,1.)
      PTCTR=1
      GO TO 40
C       FOR CALL NAMCBR
 50   CALL INCPPT(0,C.,0.,0.,1.)
C       COUNT THIS FIRST POINT IN THE POINT TOTAL
      PTCTR=1
 40   RETURN
```

```
      END
C
C  POINTS
C
      SUBROUTINE PCINTS(IS,X,Y,Z,W,*)
      COMMON /ATAEL/ IDIREC(800), PCINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,CPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPCINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM,PAD
      EQUIVALENCE (IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       IF WE ARE NOT IN OBJECT DEFINITION MODE (I.E., OPEN¬=1)
C       WE DO NOT BELONG HERE.
      IF(OPEN.EQ.1)GO TO 635
      IF(BATCH)CALL LCOUNT(2)
      PRINT 634
 634  FORMAT('0***DRAWL ERROR - PCINTS CALLED WITH NO NAMED ',
     1  'OBJECT VIA NAMOBJ.')
      CALL ABORTZ(1)
      RETURN1
C       INCREMENT PCINT COUNTER
 635  PTCTR=PTCTR+1
C       TPTCTR IS A "NAME" FOR THIS POINT
      IPOINT(PTPTR*2-1)=DIRPTR
      IPOINT(PTPTR*2)=TPTCTR
C       PUT THE X VALUE INTO PCINTS. NOTE THAT ITS TYPE
C       IS CHECKED. IT MUST BE A FLOATING PCINT NUMBER.
C       SIMILARLY INSERT THE Y AND Z VALUES.
      PT=PTPTR+1
      PCINT(PT)=CHECKF(X)
      PT=PT+1
      PCINT(PT)=CHECKF(Y)
      PT=PT+1
      PCINT(PT)=CHECKF(Z)
      PT=PT+1
C       MAKE SURE THERE IS NOT A ZERO SCALE FACTOR (W)
      W1=CHECKF(W)
      IF(ABS(W1-0.0)-LT..0001)GO TO 616
      PCINT(PT)=W1
      GO TO 617
 616  IF(BATCH)CALL LCOUNT(1)
      IF(NPRT)PRINT 550
 550  FORMAT(' ***WARNING. SCALE FACTOR OF ZERO. CHANGED TO ONE')
      CALL ABORTZ(2)
C       IF W WAS ZERO, ASSUME HE MEANT ONE, AND PUT IT IN
      POINT(PT)=1.
C       UPDATE THE TOTAL PCINT COUNTER
 617  TPTCTR =TPTCTR +1
      ITP = ITP+2
      ITABLE(ITP)=PTPTR
C       PUT THE PENUP/PENDOWN SWITCH IN THE LINKAGE TABLE, MAKING
C       SURE THAT IT IS AN INTEGER (VIA ICHKF)
      ITABLE(ITP+1) = ICHKF(IS)
C       UPDATE PCINTS TABLE INDEX
      PTPTR = PT+1
```

```
C       MAKE SURE WE ARE NOT READY TO OVERFLOW THE POINTS TABLE
      IF(PTPTR.GT.PTMAX)CALL ABORTZ(-1)
      RETURN
      END
C
C    INCRPT
C
      SUBROUTINE INCRPT(IS,X,Y,Z,W)
      COMMON /ATABL/ IDIREC(800),POINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM
      EQUIVALENCE (IPOINT(1),POINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       IF THERE WERE NO PREVIOUSLY DEFINED POINTS, WE CAN'T BE RELATIVE
C       TO THE PREVIOUS VALUE
      IF(TPTCTR.NE.1)GO TO 10
      IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 20
 20   FORMAT(' ***WARNING: THERE IS NOTHING TO USE AS BASE FOR',
     * ' RELATIVE CR'/'    INCREMENTAL CALL. (0.,0.,0.) ASSUMED.')
      CALL ABORTZ(16)
C       SO WE ASSUME HE MEANT TO BE RELATIVE TO (0.,0.,0.)
      CALL POINTS(IS,0.,0.,0.,1.)
      RETURN
C       GET THE PREVIOUS X,Y,AND Z VALUES
 10   IROW=TPTCTR*5-7
C       AND ADD THEM IN
      X1=TABLE(IROW)+X/W
      Y1=TABLE(IROW+1)+Y/W
      Z1=TABLE(IROW+2)+Z/W
C       LET POINTS PUT THEM INTO THE POINT TABLE
      CALL POINTS(IS,X1,Y1,Z1,1.)
      RETURN
      END
C
C     SPECIAL ENTRIES FOR POINTS
C
      SUBROUTINE VISA(X,Y,Z)
C       THESE ARE SELF EXPLANATORY
      CALL POINTS(1,X,Y,Z,1.)
      GO TO 1
      ENTRY INVA(X,Y,Z)
      CALL POINTS(0,X,Y,Z,1.)
      GO TO 1
      ENTRY VISR(X,Y,Z)
      CALL INCRPT(1,X,Y,Z,1.)
      GO TO 1
      ENTRY INVR(X,Y,Z)
      CALL INCRPT(0,X,Y,Z,1.)
      GO TO 1
      ENTRY VISA2(X,Y)
      CALL POINTS(1,X,Y,0.,1.)
      GO TO 1
      ENTRY INVA2(X,Y)
```

```
      CALL PCINTS(C,X,Y,0.,1.)
      GO TO 1
      ENTRY VISR2(X,Y)
      CALL INCRPT(1,X,Y,0.,1.)
      GO TO 1
      ENTRY INVR2(X,Y)
      CALL INCRPT(C,X,Y,0.,1.)
      GO TO 1
 1    RETURN
      END
C
C   NAMTRA
C
      SUBROUTINE NAMTRA(OBJNAM,A1,A2,A3,A4,A5,A6,A7,A8,A9,
     * A10,A11,A12,A13,A14,A15,A16,*)
      COMMON /ATABL/ IDIREC(800), PCINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPCINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR,Q(2),
     * TEST/Z6CCCCCCC/
      REAL*8 DIREC(400),OBJNAM,GENNAM,PAD,OBJNM1
      EQUIVALENCE (IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1)),(Q(1),OBJNM1)
C       ARRAY E WILL BE USED BY NMTRA1
      DIMENSION E(4,4)
C       IFLG = 0 IF ENTRY WAS VIA NAMTRA
      IFLG=0
      GO TO 335
      ENTRY NMTRA1(OBJNAM,E,*)
C       IFLG = 1 IF ENTRY WAS VIA NMTRA1
      IFLG=1
C       DO WE REALLY BELONG HERE?
 335  IF(OPEN.NE.0)CALL CLOSE
      OPEN=0
C       PAD OUT THE NAME PASSED AS ARGUMENT
      OBJNM1=PAD(OBJNAM)
C       WE WILL TRY TO FIND THE NAME TO SEE IF IT
C       HAS BEEN USED BEFORE. (DIRPTR=2 ===> NULL NAME TABLE)
      IF(DIRPTR.EQ.2)GO TO 115
C       FNDNAM WILL DETERMINE WHETHER THE NAME HAS BEEN USED BEFORE.
C       IF THIS IS THE FIRST DEFINITION OF THE NAME, IT WILL EXIT
C       TO 115. OTHERWISE, THE TYPE ASSOCIATED WITH THE NAME WILL
C       BE IN ITYP, AND THE LOCATION IN  DIREC IN IFIND.
      CALL FNDNAM(IFIND,OBJNM1,ITYP,&115)
      IF(BATCH)CALL LOCUNT(2)
      PRINT 101,OBJNAM
C       IF IT WAS PREVIOUSLY USED AS AN ENTITY NAME, THAT'S A FATAL
C       ERROR.
 101  FORMAT('0***DRAWL ERROR - ',A8,' ALREADY USED AS DRAWL NAME')
      CALL ABORT2(3)
      RETURN 1
C       WE REACHED THIS POINT ONLY IF THIS IS A 'NORMAL DEFINITION'.
C       PUT THE NAME INTO THE NAME TABLE.
 115  DIREC(DIRPTR)=OBJNM1
C       OVERFLOW THE NAME TABLE?
      IF(DIRPTR.GT.DIRMAX)GO TO 776
```

```
C       PUT IN TYPE 2 FOR TRANSFORMATION TYPE
      IDIREC(DIRPTR*2+1) = 2
C       POINTER INTO LINKAGE TABLE
      IDIREC (DIRPTR*2+2) = TABPTR+1
C       BACK POINTER
      ITABLE (TABPTR)=DIRPTR
C       CHECK AND SEE WHETHER THE FIRST CHARACTER IN THE NAME IS
C       A PERCENT SIGN (%). IF SO, IT WAS PROBABLY A CALL BY CMBMAT.
      IF(LAND(Q(1),TEST).EQ.TEST)GO TO 938
C       PRINT NORMAL ENTRY MESSAGE
      IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 103,OBJNM1
  103 FORMAT('0ENTRY TO NAMTRA(''',A8,2H'))
      GO TO 939
  938 IF(BATCH)CALL LOCUNT(2)
C       INTERNAL CALL MESSAGE
      IF(NPRT)PRINT 940,OBJNM1
  940 FORMAT('0INTERNAL CALL TO NAMTRA(''',A8,2H'))
C       UPDATE DIREC INDEX
  939 DIRPTR = DIRPTR + 2
C       CHECK FOR NAME AND LINKAGE OVERFLOWS
      IF(DIRPTR.GT.DIRMAX)GO TO 776
      IF(TABPTR+34.GT.TABMAX)CALL ABORTZ(-3)
      ITABLE(TABPTR+1) = 2
      L=(TABPTR+3)/2
      IF(IFLG.EQ.1)GO TO 106
C       THIS SECTION HANDLES  'NORMAL' ENTRIES
C       NOTE THAT EACH ARGUMENT A1...A16 IS CHECKED FOR TYPE
  105 TABLE(L)=CHECKF(A1)
      L=L+1
      TABLE(L)=CHECKF(A2)
      L=L+1
      TABLE(L)=CHECKF(A3)
      L=L+1
      TABLE(L)=CHECKF(A4)
      L=L+1
      TABLE(L)=CHECKF(A5)
      L=L+1
      TABLE(L)=CHECKF(A6)
      L=L+1
      TABLE(L)=CHECKF(A7)
      L=L+1
      TABLE(L)=CHECKF(A8)
      L=L+1
      TABLE(L)=CHECKF(A9)
      L=L+1
      TABLE(L)=CHECKF(A10)
      L=L+1
      TABLE(L)=CHECKF(A11)
      L=L+1
      TABLE(L)=CHECKF(A12)
      L=L+1
      TABLE(L)=CHECKF(A13)
      L=L+1
      TABLE(L)=CHECKF(A14)
      L=L+1
      TABLE(L)=CHECKF(A15)
      L=L+1
      A17=CHECKF(A16)
```

```
      GO TO 773
C       NMTRAI HANDLING SECTION. JUST LOOP THROUGH ARRAY E
 106  DO 772 J=1,4
      DO 772 I=1,4
      TABLE(L)=CHECKF(E(J,I))
 772  L=L+1
      L=L-1
      A17=CHECKF(E(4,4))
C       CHECK THE 16TH ELEMENT TO MAKE SURE ITS NOT ZERO (OR
C       SMALL ENOUGH TO BE THE NEXT BEST THING TO ZERO.)
 773  IF(ABS(A17-0.).LT..0001)GO TO 774
C       IF NOT ZERO, STORE IT.
      TABLE(L)=A17
      GO TO 775
 774  IF(BATCH)CALL LOCUNT(1)
      IF(NPRT)PRINT 550
 550  FORMAT(' ***WARNING - SCALE FACTOR OF ZERO. CHANGED TO ONE.')
C       IF IT WAS ZERO, ONE IS A PRETTY GOOD GUESS AS TO WHAT HE MEANT.
      A17=1.
      CALL ABORTZ(4)
      TABLE(L)=A17
C       UPDATE LINKAGE INDEX
 775  TABPTR=TABPTR+34
C       RESET IFLG FOR NEXT TIME
      IFLG=0
      RETURN
 776  CALL ABORTZ(-2)
      RETURN
      END
C
C   NAMASM
C
      SUBROUTINE NAMASM(OBJNAM,*)
      COMMON /ATABL/ IDIREC(800), POINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM,PAC,OBJ
      EQUIVALENCE (IPOINT(1),POINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       NAMSUB WAS USED IN THE EARLIEST VERSIONS OF DRAWL, WHEN THE ONL
C       LEVELS WERE POINTS, SUBASSEMBLIES, AND SUBASSEMBLIES. ASSEMBLIE
C       COULD CONTAIN ONLY SUBASSEMBLIES AND POINTS, AND SUBASSEMBLIES
C       COULD CONTAIN ONLY POINTS. WE'VE COME A LONG WAY.....
C       THIS WAS LEFT IN FOR COMPATIBILITY.
C       FIRST QUESTION, AS USUAL, IS 'DOES HE BELONG HERE?'
      IF(OPEN.NE.0)CALL CLOSE
C       IF SO, OPEN = 3 ====> MODE IS 'ASSEMBLY DEFINITION'
      OPEN=3
C         SEE IF NAME IS ALREADY DEFINED. IF SO, RETURN 1
      CALL FNDNAM(IFIND,PAC(OBJNAM),ITYP,&200)
      DIREC(DIRPTR)=PAC(OBJNAM)
      PRINT 201,DIREC(DIRPTR)
 201  FORMAT('0***DRAWL ERROR - ''',A6,''' ALREADY USED AS DRAWL NAME')
      CALL ABORTZ(3)
      RETURN 1
```

```
C       STORE NAME, TYPE, AND POINTER TO TABLE IN DIREC
 200  DIREC(DIRPTR)=PAD(OBJNAM)
      IDIREC(DIRPTR*2+1)=3
      IDIREC(DIRPTR*2+2)= TABPTR+1
C       BACK POINTER
      ITABLE(TABPTR)=DIRPTR
      IF(BATCH)CALL LCOUNT(2)
      IF(NPRT)PRINT 301,DIREC(DIRPTR)
C       ENTRY MESSAGE
 301  FORMAT('0ENTRY TO NAMASM(''',A8,2H'))
C         UPDATE NAME INDEX AND CHECK FOR OVERFLOW
      DIRPTR=DIRPTR+2
      IF(DIRPTR.GT.DIRMAX)CALL ABORTZ(-2)
      IF(TABPTR+3.GT.TABMAX)CALL ABORTZ(-3)
C       STORE TYPE IN LINKAGE TABLE AND ZERO TWO UNUSED HALF-WORDS
      ITABLE(TABPTR+1)=3
      ITABLE(TABPTR+2)=0
      ITABLE(TABPTR+3)=0
C         INTITALIZE THE 'ENTITIES IN THIS ASSEMBLY' COUNTER
      NASM =0
      RETURN
C
C  NAME
C
      ENTRY NAME(OBJNAM,*,*,*)
C       MAKE SURE HE IS IN THE RIGHT PLACE
      IF(OPEN.EQ.3)GO TO 105
      IF(BATCH)CALL LCOUNT(2)
      PRINT 104
 104  FORMAT('0***DRAWL ERROR - NAME CALLED WITH NO ',
     1  'NAMED ASSEMBLY VIA NAMASM')
      CALL ABORTZ(5)
      RETURN 2
 105  OBJ=PAD(OBJNAM)
C         FIND THE NAME IN THE NAME TABLE
      CALL FNDNAM(I,OBJ,ITYP,&100)
C       IT HAD BETTER NOT BE TRANSFORMATION TYPE (2)
      IF(ITYP.NE.2)GO TO 107
      IF(BATCH)CALL LCOUNT(2)
      PRINT 108,OBJ
 108  FORMAT('0***DRAWL ERROR - TRANSFORMATION ''',A8,''' USED',
     1 ' AS ARGUMENT TO NAME.')
      CALL ABORTZ(6)
      RETURN 3
 107  NASM=NASM+1
      INDX=TABPTR+5+NASM
C       HAVING INCREMENTED THE ENTITY COUNTER, AND CALCULATED
C       AN INDEX INTO THE LINKAGE TABLE, MAKE SURE WE'RE NOT ABOUT TO
C       OVERFLOW SAID IMPORTANT TABLE.
      IF(INDX.GT.TABMAX)CALL ABORTZ(-3)
C       I WAS RETURNED BY FNDNAM
      ITABLE(INDX)=I
      RETURN
C       IF THE NAME WAS NOT FOUND, COME DOWN HERE
 100  IF(BATCH)CALL LCOUNT(2)
      PRINT 110,OBJ,OBJ
 110  FORMAT('0***DRAWL ERROR.''',A8,2H'(,Z16,') NOT FOUND BY NAME.')
      CALL ABORTZ(7)
      RETURN 1
```

```
      END
C
C    FNDNAM
C
      SUBROUTINE FNDNAM(IFIND,NAM,ITYP,*)
      COMMON /ATABL/ IDIREC(800), PCINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPCINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC ,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM,FAC,NAM
      EQUIVALENCE (IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       ESTABLISH THE UPPER LIMIT OF DIREC. WE WILL COUNT BACK-
C       WARDS THROUGH THE NAME TABLE.
      NUP=DIRPTR-2
      DO 100 J=2,NUP,2
      I=DIRPTR-J
C       IF THIS IS THE NAME, BRANCH
      IF(NAM.NE.DIREC(I))GO TO 100
C       SEND BACK THE TYPE
C       (WHERE 1===>OBJECT,2===>TRANSFORM,3===>ASSEMBLY,4===>TRANSFORMED
C       OBJECT OR ASSEMBLY)
      ITYP=IDIREC(I*2+1)
C       ALSO SEND BACK THE LOCATION WHERE IT CAN BE FOUND
      IFIND=I
      RETURN
  100 CONTINUE
      RETURN 1
      END
C
C    FINOBJ
C
      SUBROUTINE FINOBJ
      COMMON /ATABL/ IDIREC(800), PCINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPCINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
C       ERROR IS USED IN THE PRINT STATEMENT BELOW
      REAL*8 DIREC(400),OBJNAM,GENNAM,ERROR(3)/'FINOBJ  ','ENDCMB  ',
     1 'FINASM  '/,XNAME
C       TF AND THE VARIABLES IN /CLSE/ ARE USED BY ENDCMB
      REAL TF(16)
      COMMON /CLSE/TC(4,4),XNAME,ISW
      EQUIVALENCE (IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       DO WE BELONG HERE?
C       TO HAVE ACCESSED THIS ROUTINE, WE SHOULD HAVE 'OBJECT
C       DEFINITION STATE' (OPEN=1) SET ON
      IF(OPEN.EQ.1)GO TO 110
      IF(BATCH)CALL LOCUNT(1)
C       ERROR(OPEN) IS WHAT SHOULD HAVE BEEN CALLED
      IF(NPRT)PRINT 111,ERROR(1),ERROR(OPEN)
C        NOW GO AND SEE WHERE HE SHOULD HAVE BEEN
```

```
      IF(OPEN.EQ.3)GO TO 160
      GO TO 210
C       STORE THE NUMBER OF POINTS IN LINKAGE TABLE
 110  ITABLE(TABPTR+2)=PTCTR
C       UPDATE LINKAGE INDEX
      TABPTR =ITP+2
      IF(BATCH)CALL LCCUNT(1)
      IF(NPRT)PRINT 101,DIREC(DIRPTR-2),PTCTR
C       OUTPUT THE NUMBER OF POINTS IN THE OBJECT (A NICE CHECK
C       ON DO LOOPS, LOOPING READS, ETC.)
 101  FORMAT(3H  ',A8,''' HAS ',I4,' POINTS.' )
C       RESET OPEN ("I'M READY FOR ANYTHING STATE")
      OPEN=0
      RETURN
C
C   FINASM
C
      ENTRY FINASM
C       IF WE KEEP NAMSUB, WE JUST GOTTA HAVE A FINSUB
 150  IF(OPEN.EQ.3)GO TO 160
C       ONCE AGAIN, IF HE DOESN'T BELONG HERE, TELL HIM SO
      IF(BATCH)CALL LCCUNT(1)
      IF(NPRT)PRINT 111,ERROR(3),ERROR(OPEN)
      IF (OPEN.EQ.1)GO TO 110
      GO TO 210
C       RESET OPEN
 160  OPEN=0
C       OUTPUT THE NUMBER OF ENTITIES INCLUDED
      IF(NPRT)PRINT 170,NASM
 170  FORMAT('  THERE ARE ',I3,' CONSTITUENT ENTITIES.')
C       STORE AWAY THE NUMBER
      ITABLE(TABPTR+4)=NASM
C       EVEN-ALIGNMENT IS REQUIRED IN LINKAGE TABLE
      IF((NASM/2*2).NE.NASM) TABPTR=TABPTR+1
C       UPDATE LINKAGE INDEX
      TABPTR = TABPTR + NASM + 6
C       RESET NASM FOR NEXT TIME
      NASM = 0
      RETURN
C
C   ENDCMB
C
      ENTRY ENDCMB
C       HE TRULY BELONGS HERE IFF OPEN=2 (CMBMAT WAS CALLED)
 205  IF(OPEN.EQ.2)GO TO 210
      IF(BATCH)CALL LCCUNT(1)
      IF(NPRT)PRINT 111,ERROR(2),ERROR(OPEN)
      IF(OPEN.EQ.1)GO TO 110
      GO TO 160
 111  FORMAT(' ***WARNING - ILLEGAL CALL TO ',A7,1H;,A7,
     1  'CALLED FOR YOU.')
      CALL ABORTZ(15)
 210  OBJNAM=XNAME
C       RESET OPEN
      OPEN=0
C       SINCE CMBMAT SET A SERIES OF (AT LEAST ONE) MATRIX MULTIPLIES
C       INTO ACTION, THE MATRIX  IN TO IS ALL READY FOR TRANSFORMATION
C       DEFINITION. PASS THE NAME GIVEN TO CMBMAT AND THE MULTIPLIED
C       ARRAY IN TO TO NMTRAI (THE ALTERNATE ENTRY TO NAMTRA).
```

```
      CALL NMTRA1(OBJNAM,TC)
C       RESET ISW (CONTAINED IN /CLSE/)
      ISW=0
      RETURN
      END
C
C   TRANSF
C
      SUBROUTINE TRANSF(NEW,OLD,XFORM,*,*,*,*)
C       IT SHOULD BE POINTED OUT THAT TRANSF WILL NOT PERFORM
C       THE MATRIX MULTIPLICATIONS REQUIRED FOR A TRANSFORMATION.
C       IT MERELY SETS UP THE LINKAGES, AND THE MULTIPLICATIONS
C       ARE PERFORMED WHEN DRAW, SHOW, PRTPLT,PRTOUT,ETC., ARE CALLED
      COMMON /ATABL/ IDIREC(800), POINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR,Q(2)
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      REAL*8 DIREC(400),OBJNAM,GENNAM,PAD,NEW,OLD,XFORM
C       ERROR MESSAGE DATA IS IN TYPE
      REAL*8 TYPE(4)/' OBJECT ',' XFORM  ','ASSEMBLY','ASSEMBLY'/
      REAL*8 NEW1,XFORM1,OLD1
      EQUIVALENCE (IPOINT(1),POINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1)),(GENNAM,Q(1))
C       MASK WILL BE USED TO DETERMINE WHETHER THE CHARACTER
C       MANIPULATION SUPPORTING NAME-GENERATION WILL BECOME
C       ULTRA-COMPLEX, OR MERELY REMAIN MODERATELY CONFUSING....
      DATA MASK/ZCCCCCCOA/
C       AS USUAL, DOES HE BELONG HERE?
      IF(OPEN.NE.0)CALL CLOSE
      OPEN=0
C       PAD OUT THE THREE ARGUMENTS
      NEW1=PAD(NEW)
      OLD1=PAD(OLD)
      XFORM1=PAD(XFORM)
      IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 663,NEW1,OLD1,XFORM1
C       PRINT THE ENTRY MESSAGE
 663  FORMAT('0ENTRY TO TRANSF(',2(1H',A8,2H',),1H',A8,2H'))
C       FIND THE TRANSFORMATION MATRIX.  IF ITS NOT THERE, BRANCH TO 357
      CALL FNDNAM(J,XFORM1,ITYP,&357)
C       MAKE SURE THE NAME GIVEN WAS A TRANSFORMATION
C       BRANCH TO 665 IF IT WAS
      IF(ITYP.EQ.2)GO TO 665
      IF(BATCH)CALL LOCUNT(2)
      PRINT 664,XFORM1
C       IF IT WASN'T, TELL HIM ALL ABOUT IT
 664  FORMAT('0***CRAWL ERROR - ''',A8,''' IS NOT A TRANSFORMATION ',
     1 'NAME')
      CALL ABORT2(17)
      RETURN 1
C       FIND THE 'SOURCE' NAME IN DIREC. IF IT WASN'T THERE,
C       BRANCH TO 356
 665  CALL FNDNAM(I,OLD1,ITYP,&356)
C       IF IT WAS A TRANSFORMATION, GO TO 834
      IF(ITYP.EQ.2)GO TO 834
C       IF THIS CALL TO TRANSF WAS EQUIVALENT TO THE FORTRAN STATEMENT
```

```
C                A=A*B
C       (I.E., THE FIRST ARGUMENT WAS THE SAME AS THE SECOND),KEEP GOING
C       OTHERWISE BRANCH TO 800
      IF(OLD1.NE.NEW1) GO TO 800
C       Q IS INVOLVED IN BYPASSING THE FORTRAN IV CHARACTER MANIP-
C       ULATION RESRICTIONS. THE FUNCTION 'LAND' DOES ITS SHARE
C       TOO. LAND IS A BITWISE LOGICAL FUNCTION. A BIT IS ON IN
C       LAND'S RETURNED VALUE ONLY IF IT WAS ON IN THE SAME POSITION
C       IN BOTH OF THE ARGUMENTS. THIS SEQUENCE MERELY CHECKS TO
C       SEE IF OVERFLOW IN THE EBCDIC REPRESENTATION OF THE GENER-
C       ATED NAME REQUIRED BY TRANSF AT THIS JUNCTURE IS OCCURRING.
      Q(2)=Q(2)+1
      IF(LAND(MASK,Q(2)).EQ.MASK)Q(2)=Q(2)+246
C       IN THIS IMPLEMENTATION, THE VALUES ASSOCIATED WITH
C       THE NAME GIVEN AS FIRST AND SECOND ARGUMENTS ARE PRESERVED
C       AND GIVEN A NEW (GENERATED) NAME. THE NEW NAME IS PRINTED
C       HERE. NOTE THAT THIS ALLOWS THE OLD VALUES TO STILL BE
C       REFERENCED INTERNALLY, BUT REFERENCES TO THE OLD NAME NOW POINT
C       TO THE TRANSFORMED VALUE
C       I WAS RETURNED BY FNDNAM. PUT THE GENERATED NAME IN THE
C       TABLE IN PLACE OF THE OLD NAME.
  100 DIREC(I)=GENNAM
C       PUT THE NAME JUST REPLACED INTO THE NEXT SLOT IN DIREC
      DIREC(DIRPTR)=NEW1
C       PUT IN THE 'XFORMED-ENTITY' KEY (=4)
      IDIREC(DIRPTR*2+1)=4
C       INDEX INTO LINKAGE TABLE
      IDIREC(DIRPTR*2+2)=TABPTR+1
C       BACK POINTER
      ITABLE(TABPTR)=DIRPTR
C       TYPE 4 AGAIN
      ITABLE(TABPTR+1)=4
C       PUT LOCATION OF THE XFORM  INTO LINKAGE TABLE
C       (AS RETURNED BY FNDNAM ABOVE)
      ITABLE(TABPTR+2)=J
C       PUT IN LOCATION OF SOURCE ENTITY
      ITABLE(TABPTR+3)=I
C       UPDATE INDICES FOR NAME AND LINKAGE TABLES
      DIRPTR=DIRPTR+2
      TABPTR=TABPTR+6
  200 RETURN
C       THIS SECTION IS FOR 'NORMAL' TRANSFORMATIONS
C       (I.E., THE FIRST TWO ARGUMENTS ARE NOT THE SAME.)
C
C       CHECK AND SEE WHETHER THE NAME HAS BEEN USED BEFORE
C       IF IT HASN'T, BRANCH TO 801
  800 CALL FNDNAM(K,NEW1,ITYP1,&801)
      IF(BATCH)CALL LCOUNT(2)
      PRINT 803,NEW1,TYPE(ITYP1)
C       TELL HIM THAT HE'S USED THE NAME BEFORE.
C       ALSO TELL HIM WHAT MODE WAS ASSOCIATED WITH THE NAME.
  803 FORMAT('0***CRAWL ERROR - ''',A8,''' ALREADY USED AS ',
     1 A8,' NAME.')
      CALL ABORT2(18)
      RETURN 4
C       PUT AWAY THE NEW NAME (THIS IS AN ENTITY DEFINITION)
  801 DIREC(DIRPTR)=NEW1
C       BACKPOINTER IN LINKAGE TABLE
      ITABLE(TABPTR)=DIRPTR
```

```
C         TYPE
      ITABLE(TABPTR+1) =4
C         LOCATION OF TRANSFORMATION (AS RETURNED BY FNDNAM)
      ITABLE(TABPTR+2)=J
C         LOCATION OF SOURCE ENTITY (SECOND ARGUMENT)
      ITABLE(TABPTR+3)=I
C         POINTER TO LINKAGE TABLE FROM NAME TABLE
      IDIREC(DIRPTR*2+2)=TABPTR+1
C         TYPE
      IDIREC(DIRPTR*2+1)=4
C         INCREMENT INDICES AND CHECK FOR TABLE OVERFLOWS
      DIRPTR=DIRPTR+2
      TABPTR=TABPTR+6
      IF(DIRPTR.GT.DIRMAX)CALL ABORTZ(-2)
      IF(TABPTR.GT.TABMAX)CALL ABORTZ(-3)
      RETURN
C         ERROR MESSAGES FOR THE ABOVE ROUTINE.....
  834 IF(BATCH)CALL LOCUNT(2)
      PRINT 835,OLD1
  835 FORMAT('0***DRAWL ERROR - ''',A8,''' IS A TRANSFORMATION NAME')
      CALL ABORTZ(19)
      RETURN 2
  356 IF(BATCH)CALL LOCUNT(2)
      PRINT 110,OLD1,OLD1
      GO TO 358
  357 IF(BATCH)CALL LOCUNT(2)
      PRINT 110,XFORM1,XFORM1
  110 FORMAT('0***DRAWL ERROR - ''',A8,2H'(,Z16,') NOT FOUND BY TRANSF')
      CALL ABORTZ(20)
  358 RETURN 3
      END
C
C  DRAW SUBROUTINE
C
      SUBROUTINE DRAW(OBJNAM,*,*,*,*)
C         THIS SECTION OF CODE IS VIRTUALLY IDENTICAL FOR
C         DRAW, SHOW,PRTFLT,DISPLA,PRTOUT,PRTPLT. THUS ENTRIES ARE
C         PROVIDED IN THIS AREA FOR ALL SUCH ROUTINES.
C         IDEV IS SET TO INDICATE WHICH ENTRY WAS TAKEN:
C         1===>SHOW
C         0===>DRAW
C         -1===>DISPLA
C         2===>PRTOUT
C         3===>PRTFLT
C         4===>PRTPLT
      COMMON /ATABL/ IDIREC(800), POINT(5000),TABLE(5000)
      COMMON /DRWCOM/DVEC(50),CURMAT(50),EXPIRE(50),ID,IC,
     1  NPTS,ICMAX,ICMAX,XMIN,XMAX,YMIN,YMAX,T(5460),IT(2730),IL
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      INTEGER BLNK/'    '/,NC/'NC  '/,TPTCTR
      INTEGER*2 INUM,DVEC,CURMAT,EXPIRE,ID,IC,IT
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC ,OPEN,DIRMAX,DIRPTR,PTCTR,PT
      REAL*8 DIREC(400),OBJNAM,GENNAM,DRWPRT(6)/'DISPLA  ',' DRAW   ',
     + ' SHOW   ','PRTOUT  ','PRTPLT  ','PRTPLT  '/,PAD,FOUND,XFOUND,
     + OBJNM1
```

```
      EQUIVALENCE (IPCINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
      IDEV=0
      GO TO 38
      ENTRY SHOW(OBJNAM)
      IDEV=1
C       OFSN BLANKS THE DISPLAY SCREEN
      IF(NMOVE)CALL OFSN
      GO TO 38
      ENTRY DISPLA(OBJNAM)
      IDEV=-1
      GO TO 38
      ENTRY PRTCLT(OBJNAM)
      IDEV=2
      GO TO 38
      ENTRY PRTPLT(OBJNAM)
      IDEV=3
      GO TO 38
      ENTRY PRTPLT(OBJNAM,IPLT)
      IDEV=4
      GO TO 38
C       DOES HE BELONG HERE?
 38   IF(OPEN.NE.0)CALL CLOSE
      OPEN=0
C       INITIALIZE THE INDEX FOR THE CLTPLT ARRAY (IL=1)
      IL=1
C       CALL TIME(C) INITIALIZES THE CPU-TIME COUNTER (USED FOR
C       TESTING PURPOSES)
      IF(NPRT)CALL TIME(0)
C       THE NEXT FOUR PARAMETERS ARE FOR THE SCALING ALGORITHM
      XMAX=-1.E6
      XMIN=1.E6
      YMAX=-1.E6
      YMIN=1.E6
C       THE NEXT SIX STATEMENTS SET UP THE PRINT LINE FOR ENTRY
C       MESSAGE.
      NSC=BLNK
      NMO=NO
      NSH=BLNK
      IF(NSCAL)NSC=NO
      IF(NMOVE)NMO=BLNK
      IF(NSHFT)NSH=NO
C       THE NEXT FOUR PARAMETERS ARE INITIALIZED FOR DRETRV
      ID=1
      IC=0
      IDMAX=0
      ICMAX=0
C       NPTS IS THE NUMBER OF POINTS. IF THIS WAS THE FIRST
C       DRAWING (AS SHOWN BY LNM) WE WILL PUT ON AN EXTRA POINT
C       AT THE BEGINNING OF THE OUTPUT VECTOR. IT WILL BE
C       EQUIVALENT TO CALL PCINTS(0,0.,0.,0.,1.)
      NPTS=1
      IF(LNM.EQ.0)GO TO 8
      NPTS=0
      IL=-1
 8    LNM=LNM+1
      OBJNM1=PAD(OBJNAM)
C       FIND THE ENTITY IN THE NAME TABLE. IF NOT THERE, TO 10
      CALL FNDNAM(IFIND,OBJNM1,ITYP,&10)
```

```
      IF(BATCH)CALL LCCUNT(2)
C       DRWPRT(IDEV+2) WILL SET UP THE APPROPRIATE MESSAGE
      IF(NPRT)PRINT 9,DRWPRT(IDEV+2),OBJNM1
 9    FORMAT('CENTRY TO ',A6,2H('A8,2H'))
      IF(BATCH)CALL LCCUNT(1)
      IF(NPRT)PRINT 1,DRWPRT(IDEV+2),NSC,NMC,NSH
C       SET UP AND PRINT OUT THE APPROPRIATE PARAMETERS
 1    FORMAT(1X,A6,' PARAMETERS: ',A2,'SCAL, ',A2,'MOVE, ',A2,'SHFT')
C       MAKE SURE AN ENTITY NAME WAS PASSED.IF SO, GO TO 11
      IF(ITYP.NE.2) GO TO 11
C       OTHERWISE, PRINT ERROR MESSAGE
      IF(BATCH)CALL LCCUNT(2)
      PRINT 3,DRWPRT(IDEV+2),OBJNM1
 3    FORMAT('0***DRAWL ERROR - ATTEMPT TO ',A6,' TRANSFORMATION '''
     1  ,A8,1H')
      CALL ABORTZ(21)
      RETURN 2
 10   IF(BATCH)CALL LCCUNT(2)
      PRINT 12,OBJNM1,OBJNM1,DRWPRT(IDEV+2)
C       IF THE ARGUMENT WASN'T FOUND,SET UP MESSAGE AND PRINT IT
 12   FORMAT('0***DRAWL ERROR - '''A8,2H'(,Z16,' NOT FOUND BY ',A6)
      CALL ABORTZ(22)
 14   RETURN 1
C       NOW FOR THE RETRIEVAL AND MULTIPLICATION PART OF IT ALL.
C       DRETRV IS A PSEUDO-RECURSIVE FUNCTION. IT IS NON-TRIVIAL
C       YET SIMPLE IN ITS OWN, EFFECTIVE WAY.
C
C       INITIALIZE DVEC FOR DRETRV
 11   DVEC(1)=-IFIND
 13   CONTINUE
C       CALL DRETRV. NOTE IT WILL RETURN TO THE STATEMENT BEFORE
C       THE CALL IF IT TAKES EXIT ONE AND KEEP CALLING ITSELF
      CALL DRETRV(&13,&19)
C       EXIT TO 19 IS TAKEN IF THE USER ATTEMPTED TO PLACE THE
C       EYE POINT IN THE SAME X-Y PLANE AS ONE OF THE POINTS TO
C       BE DRAWN (THIS WOULD CAUSE DIVISION BY ZERO)
C       DRETRV HAS NOW SET UP THE OUTPUT ARRAY. CALL OUTPUT, WHICH
C       LIKE DRAW, IS COMMON TO ALL TYPES OF OUTPUT.
 18   CALL OUTPUT(NSCAL,NSHFT,NMOVE,IDEV,NPRT,IPUT,BATCH,&16)
C       EXIT TO 16 IS TAKEN IF THE DRAWING GOES OFF THE SCREEN
C       WHEN EXECUTING A 'SHOW'. THIS SHOULD COME ABOUT ONLY
C       WHEN THE USER IS RUNNING WITH SHFT OR SCAL OFF.
C       HAVING GOTTEN HERE, THE DRAWING IS COMPLETE.
C       GET THE ELAPSED C.P.U.TIME IN INT (SINCE CALL TIME(0))
      IF(NPRT)CALL TIME(1,0,INT)
C       CONVERT IT FROM MILLISECONDS TO SECONDS
      ZTIME=INT/1000.
C       AND PRINT IT OUT
      IF(BATCH)CALL LCCUNT(1)
      IF(NPRT)PRINT 17,ZTIME
 17   FORMAT(F7.3,' C. P. U. SECONDS FOR DRAWING.')
C       THE  DRAWING IS COMPLETED. ALL IS WELL.
      RETURN
C       19 IS ACCESSED ONLY IF DRETRV HAD AN ERROR AS INDICATED ABOVE
 19   RETURN 3
C       16 IS ACCESSED ONLY IF OUTPUT HAD AN ERROR AS INDICATED ABOVE
 16   RETURN  4
      END
C
```

```
C OUTPUT
C
      SUBROUTINE OUTPUT(NSCAL,NSHFT,NMOVE,IDEV,NPRT,LDN,BATCH)
C       OUTPUT'S MAIN FUNCTION IS TO SCALE DRAWINGS (AND
C       SHIFT THEM IF NECESSARY) SO THEY FIT ONTO THE OUTPUT
C       DEVICE. IT ALSO SELECTS THE APPROPRIATE OUTPUT DEVICE
C       (ACCORDING TO THE INFORMATION PASSED IN IDEV)
      INTEGER*2 INUM,DVEC,CURMAT,EXPIRE,ID,IC,IT
      COMMON /DRWCOM/DVEC(50),CURMAT(50),EXPIRE(50),ID,IC,
     1  NPTS,ICMAX,ICMAX,XMIN,XMAX,YMIN,YMAX,I(5460),IT(2730),IL
      LOGICAL NSCAL,NSHFT,NMOVE,NPRT,BATCH
      INTEGER IMAGE(1570),NSCALE(5)/0,0,3,0,3/
C       THE FOLLOWING 4 VARIABLES ARE USED FOR SCALING
      XMIN1=1.E6
      XMAX1=-1.E6
      YMIN1=1.E6
      YMAX1=-1.E6
C       XLIM AND YLIM ARE THE MAXIMUM OUTPUT SIZES ALLOWED
C       FOR THE VARIOUS DEVICES. THIS SECTION SETS THEM.
      IF(IDEV.EQ.1.OR.IDEV.EQ.-1)GO TO 10
      YLIM=28.
      XLIM=49.
      IF(IDEV.NE.0)GO TO 12
C       THE UNIVERSITY OF MICHIGAN DRAWL IMPLEMENTATION USES
C       THE UNIVERSITY'S CALCOMP ROUTINE PACKAGE. ALL SUCH ROUTINES
C       ARE NAMED STARTING WITH A LETTER P
C       PFDNAM SETS THE LOGICAL DEVICE NUMBER ONTO WHICH PLOT FILES
C       (I.E., COMMANDS TO THE PLOTTER) ARE TO BE WRITTEN.
C       AT RUN TIME, THE USERS MUST SAY
C               $RUN MYPROGRAM+SS1O:DRAWL  8=PLOTFILE
C       THIS WILL WRITE THE PLOT FILE TO LDN 8 (IF DRAW IS CALLED)
C       8 SHOULD NOT BE USED FOR ANY OTHER PURPOSE
      CALL PFDNAM(8)
C       PLTXMX SETS THE MAXIMUM PAPER LENGTH
      CALL PLTXMX(49.)
      GO TO 12
 10   IF(IDEV.EQ.-1)GO TO 11
C       THE UNIVERSITY OF MICHIGAN DRAWL IMPLEMENTATION USES THE
C       DF ROUTINE (FOR DISPLAY FILE ROUTINE) PACKAGE OF THE
C       CLNCOMP PROJECT FOR THE DIGITAL EQUIPMENT CORPORATION
C       338. ALL DF ROUTINE NAMES START WITH 'DF'.
C       THE DISPLAY SCREEN IS 9.375 INCHES SQUARE
      XLIM=9.375
      YLIM=9.375
C       DFINI INITIALIZES THE  DISPLAY FILE BUFFER
      CALL DFINI(0,0)
      GO TO 12
 11   XLIM=30.
      YLIM=10.
C       TOPEN INITIALIZES THE ROUTINE FOR THE 10" ON-LINE
C       PLOTTER REFERENCED BY DISPLA.
C       IT HAS     BEEN IMPLEMENTED AT THIS POINT
       CALL TOPEN(.TRUE.)
 12   IF(BATCH)CALL LOCUNT(3)
      IF(NPRT)PRINT 34,XMIN,XMAX,YMIN,YMAX
C       PRINT OUT THE REAL DRAWING LIMITS AS DISCOVERED BY DRETRV
 34   FORMAT('     XMIN= ',F12.3,' XMAX= ',F12.3/
     1  '     YMIN= ',F12.3,' YMAX= ',F12.3)
C       START THE SHIFTING AND SCALING PROCEDURE
```

```
C       FIND OUT IF THE LIMITS AS DISCOVERED FALL WITHIN XLIM AND YLIM
C       MAKE SURE THERE'S NOTHING IN QUADRANTS II,III,OR IV UNLESS
C       NOSHFT IS ON.
      IF(XMIN.GT.0.)XMIN=0.
      IF(YMIN.GT.0.)YMIN=0.
      XSCALE=(XMAX-XMIN)/XLIM
      YSCALE=(YMAX-YMIN)/YLIM
      IF(NSCAL)YSCALE=1.
      IF(NSCAL)XSCALE=1.
      IF(XMAX-XMIN.LT.XLIM)XSCALE=1.
      IF(YMAX-YMIN.LT.YLIM)YSCALE=1.
      IF(XSCALE.LT.YSCALE)XSCALE=YSCALE
      IF(YSCALE.LT.XSCALE)YSCALE=XSCALE
      IF(NSHFT)XMIN=0.
      IF(NSHFT)YMIN=0.
C       J WILL COUNT THE NUMBER OF POINTS USED (ONE DIS-
C       PLAY FILE CAN HOLD ONLY 255 POINTS.)
      J=0
  976 CONTINUE
      IF(IDEV.GT.1)GO TO 426
C       LOOP THROUGH THE ARRAY T WHICH WAS SET UP BY DRETRV
C       AND SCALE AND SHIFT ALL POINTS (NOTE THE SCALE
C       FACTOR IS ONE IF NOSCAL IS ON AND THE SHIFT FACTOR
C       IS ZERO IF NOSHFT IS ON.
C       THE NEW MAXS AND MINS ARE RECORDED.
      DO 126 I=1,NPTS
      XR1=(T(I*2-1)-XMIN)/XSCALE
      YR1=(T(I*2)-YMIN)/XSCALE
      IF(XR1.LT.XMIN1)XMIN1=XR1
      IF(XR1.GT.XMAX1)XMAX1=XR1
      IF(YR1.LT.YMIN1)YMIN1=YR1
      IF(YR1.GT.YMAX1)YMAX1=YR1
C       GO TO THE RIGHT OUTPUT DEVICE'S ROUTINES
      IF(IDEV)327,127,227
C       THIS IS CALCOMP TERRITORY. THE SWITCH WILL BE IN "IT".
C       ALLOWABLE SWITCHES ARE 0=INVISIBLE,1=VISIBLE,2=DASHED
C       LINE,3=CENTER-LINES ARE TO BE DRAWN.
C       THE SWITCHES HAVE TO BE BUMPED UP BY ONE TO BE MADE
C       COMPATIBLE WITH THE ROUTINES THEY FEED. 0 WAS USED FOR
C       INVISIBLE BECAUSE OF ITS LOGICAL CONNECTIONS.
  127 IS=IT(I)+1
C       EACH TYPE OF LINE TO BE DRAWN HAS IT'S OWN PLACE
      GO TO (120,121,122,123),IS
C       NOTE THAT IF AN ILLEGALLY HIGH VALUE IS FOUND, IT DEFAULTS
C       TO PENUP, OR INVISIBLE LINE.
C       PENUP'S ARGUMENTS ARE (X,Y) IN INCHES
  120 CALL PENUP(XR1,YR1)
      GO TO 126
C       PENDN ALSO GETS (X,Y) IN INCHES
  121 CALL PENDN(XR1,YR1)
      GO TO 126
C       PDSHLN (DASHED LINES) GETS (X,Y) AND 4 OTHER PARAMETERS
C       (UNRELATED TO GENERAL IMPLEMENTATIONS.)
  122 CALL PDSHLN(XR1,YR1,2,1,0,0)
      GO TO 126
C       PCTRLN(FOR CENTER LINES) GETS (X,Y) IN INCHES. OTHER THREE
C       PARAMETERS ARE NOT PROFOUND.
  123 CALL PCTRLN(XR1,YR1,2,1,0)
      GO TO 126
```

```
C       FOR THE CRT, IF THE SWITCH IS NON-ZERO, SET IT TO 1
 227  IS=0
      IF(IT(I).NE.C)IS=1
C       UPDATE NUMBER-OF-POINTS IN THIS DRAWING COUNTER
      J=J+1
C       MAKE THE FIRST VECTOR AN INVISIBLE ONE
      IF(J.EQ.1)IS=0
C       DFXYC BUILDS A DISPLAY FILE. THE ARGUMENTS ARE
C       X AND Y IN INCHES, THE ON/OFF SWITCH. EXIT TO 999 IS TAKEN
C       IF THE DRAWING WILL GO OFF THE SCREEN (COULD CRASH
C       THE DISPLAY'S SOFTWARE OTHERWISE.)
      CALL DFXYC(XR1,YR1,IS,&999)
      IF(J.LE.254)GO TO 126
C       IF THE CURRENT FILE IS FULL, SET UP ANOTHER ONE
      J=0
C       DFAVL RETURNS THE 'NAME' OF THE LAST USED
C       DISPLAY FILE. THE ASSUMPTION THAT DRAWL IS NOT THE
C       ONLY POSSIBLE SOURCE OF DISPLAY FILES ALLOWS GREATER
C       FLEXIBILITY.
      CALL DFAVL(NAM)
C       DF201 SENDS THE DISPLAY FILE TO THE REMOTE DISPLAY
      CALL DF201(C,NAM)
C       DFINI AGAIN INITIALIZES THE NEW D.F. BUFFER
      CALL DFINI(0,0)
      GO TO 126
C       ON-LINE 10" CALCOMP ROUTINE
 327  IS=0
      IF(IT(I).NE.C)IS=1
C       SEND THE PLOTTER THE X, AND Y IN INCHES, WITH PENUP/DOWN SW.
      CALL XYABS(X,Y,IS)
 126  CONTINUE
C         126 IS THE END OF THE LOOP.
C       IF WE GET HERE, THE DRAWING IS DONE. GET THE NAME OF
C       THE LAST D.F. FOR TRANSMISSION.
      CALL DFAVL(NAM)
      GO TO 333
C       THIS SECTION IS FOR PRINTED OUTPUT (PRTPUT,PRTPLT,PRTOUT)
 426  DO 424 I=1,NPTS
      I2=I*2
      I21=I*2-1
C       SCALE THINGS IN THE ARRAY 'T' AND LEAVE THEM THERE
C       PASS THE ARRAY TO THE APPROPRIATE ROUTINE
C       FOR FINAL OUTPUT
      T(I21)=(T(I21)-XMIN)/XSCALE
      IF(T(I21).LT.XMIN1)XMIN1=T(I21)
      IF(T(I21).GT.XMAX1)XMAX1=T(I21)
      T(I2)=(T(I2)-YMIN)/XSCALE
      IF(T(I2).LT.YMIN1)YMIN1=T(I2)
      IF(T(I2).GT.YMAX1)YMAX1=T(I2)
 424  CONTINUE
      IF(IDEV-3)425,525,625
C       525 IS FOR PRINTER PLOT ROUTINE
C       625 IS FOR WRITING ON A DEVICE (DON 0 - 9)
C       425 IS FOR PRINTING OUT THE VALUES IN TABULAR FORM
 425  IF(BATCH)CALL LCOUNT(57)
      DO 427 I=1,NPTS
      IF(BATCH)CALL LCOUNT(1)
C       PRINT OUT THE VALUES IN THE ARRAY 'T'
 427  PRINT 110,IT(I),T(I*2-1),T(I*2)
```

```
 110  FORMAT(3X,I2,4(2X,F7.2))
      IF(BATCH)CALL LCOUNT(57)
      GO TO 333
 625  DO 626 I=1,NPTS
C       WRITE OUT THE VALUES IN ARRAY 'T' ON THE REQUESTED LDN
 626  WRITE(LDN,110)IT(I),T(I*2-1),T(I*2)
      GO TO 333
 525  IF(BATCH)CALL LCOUNT(57)
      IF(BATCH)ILEN=100
      IF(.NOT.BATCH)ILEN=71
      IF(XMAX1.LT.(1.5*YMAX1))XMAX1=1.5*YMAX1
      IF(YMAX1.LT.(.666*XMAX1))YMAX1=XMAX1*.666
C       PLOT14 WILL PLOT THE ENDPOINTS OF ALL LINES ON THE PRINTER
C       IN ORDER, THE ARGUMENTS ARE:
C       SCALING INFORMATION (0 FOR US), # HORIZONTAL GRID LINES,
C       # SPACES BETWEEN HORIZ. LINES, # VERTICAL LINES, #SPACES
C       BETWEEN VERTICAL LINES, AN ARRAY IN WHICH THE OUTPUT IMAGE
C       IS TO BE SET UP,THE XMAX,XMIN,YMAX,YMIN,THE CHARACTER TO
C       BE USED IN PLOTTING,THE X-ARRAY,Y-ARRAY,NUMBER OF POINTS,
C       BYTE SIZE OF X &Y ARRAYS (8 FOR US), LABEL
C       SWITCH (0 MEANS NO LABEL), AND LOCATION OF LABEL.
C       THIS ROUTINE IS IN THE MTS SYSTEM LIBRARY. DIFFERENT
C       INSTALLATIONS WILL HAVE DIFFERENT ROUTINES AND ARGUMENTS.
      CALL PLOT14(NSCALE,2,53,2,ILEN,IMAGE,XMAX1,XMIN1,YMAX1,YMIN1,
     *  '@',T(1),T(2),NPTS,8,0,N)
      IF(BATCH)CALL LCOUNT(57)
 333  CONTINUE
C       IF THE DRAWING WAS SCALED, TELL HIM ABOUT IT
      IF(ABS(XSCALE-1.0).LT.0.005)GO TO 37
      IF(BATCH)CALL LCOUNT(1)
      IF(NPRT)PRINT 36,XSCALE
 36   FORMAT(' THIS DRAWING WAS SCALED BY ',F8.3)
 37   IF(BATCH)CALL LCOUNT(3)
      IF(NPRT)PRINT 35
C       PRINT THE ACTUAL BOUNDARIES
 35   FORMAT('   ACTUAL DRAWING BOUNDARIES:')
      IF(NPRT)PRINT 34,XMIN1,XMAX1,YMIN1,YMAX1
C       FINISH EVERYTHING OFF - PLTEND MOVES THE CALCOMP PAPER,
C       DF201 SENDS THE LAST D.F.
C       TCLOSE CLOSES THE ON-LINE PLOTTER'S BUFFER
      IF(IDEV.EQ.0.AND.NMOVE)CALL PLTEND
      IF(IDEV.EQ.1)CALL DF201(0,NAM)
      IF(NMOVE.AND.IDEV.EQ.-1)CALL TCLOSE
      RETURN
 999  RETURN 1
      END
C
C INTERNAL RETRIEVAL SUBROUTINE
C
C       DRETRV IS A PSEUDO-RECURSIVE ROUTINE. IT 'PUSHES'
C       AS MANY TIMES AS THERE ARE LEVELS OF NESTED ASSEMBLIES OR
C       APPLICATIONS OF TRANSFORMATION MATRICES. IT MAINTAINS THREE
C       PUSH-DOWN STACKS :
C       DVEC - A LIST OF THE ENTITIES NESTED WITHIN ENTITIES. THE
C       ENTRIES CONSIST OF THE POINTERS TO THE ENTITY'S
C       LOCATIONS IN THE NAME TABLE. THE POINTER TO THE NAME
C       WHICH CONTAINS OTHER ENTITIES HAS THE NEGATIVE OF ITS POINTER.
C       WHEN AN ENTITY IS PUSHED ONTO DVEC, ALL ITS CONSTITUENT
C       ENTITIES ARE ALSO PUSHED ONTO DVEC. THE ENTITIES
```

```
C       WHICH ARE CONTAINED IN THE LAST ENTITY (IF ANY) ARE THEN
C       PUSHED ONTO THE END OF THE STACK, WITH THE OLD END BEING
C       GIVEN THE NEGATIVE OF ITS VALUE. IC IS DVEC'S INDEX.
C       CURMAT - CONTAINS POINTERS TO THE TRANSFORMATION MATRICES
C       WHICH ARE TO BE APPLIED TO ALL POINTS PROCESSED. THIS KEEPS
C       TRACK OF THE LEVELS OF TRANSFORMATION. IC IS THE INDEX
C       FOR THIS STACK. TMAT CONTAINS THE PRODUCT OF ALL MATRICES
C       FROM CURMAT(1)...CURMAT(IC) AT ANY GIVEN TIME.
C       EXPIRE - KEEPS THE VALUES OF IC SUCH THAT WHEN EXPIRE(IC)
C       IS LESS THAN  ID, CURMAT(IC) IS TAKEN OFF THE LIST, AND
C       TMAT IS RE-EVALUATED. IC IS USED BY EXPIRE AS INDEX.
      SUBROUTINE DRETRV(*,*)
      COMMON /DATAEL/ IDIREC(800), PDINT(5000),TABLE(5000)
      COMMON /DRWCOM/DVEC(50),CURMAT(50),EXPIRE(50),ID,IC,
     1  NPTS,IDMAX,ICMAX,XMIN,XMAX,YMIN,YMAX,T(5460),IT(2730),IL
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      INTEGER*2 INUM,DVEC,CURMAT,EXPIRE,IC,IC,IT
      INTEGER*2 IPDINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC ,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR,IX/1/,IY/2/
      LOGICAL BATCH
      REAL*8 DIREC(400),PAD,GENNAM
      EQUIVALENCE (IPDINT(1),PDINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
C       OUTPTS WILL CONTAIN THE VALUES ULTIMATELY PASSED TO THE
C       OUTPUT ARRAY 'T'
C       TMAT, TMAT1, AND TPDINT ARE SCRATCH MATRICES.
      REAL OUTPTS(4),TPOINT(4),TMAT(4,4),TMAT1(4,4)
C       ID WAS SET TO 1 BY DRAW (STMT 11) FOR THE FIRST TIME THRU DRETRV
      INUM=ID
C       WHEN ID=0 WE'RE DONE.
      IF(INUM.EQ.0)RETURN
C       TO 4 IF THIS IS THE FIRST TIME THROUGH
      IF(INUM.EQ.1)GO TO 4
C       THIS SECTION 'POPS' ALL STACKS.
      DO 100 I=1,IC
      I1=ID-I+1
C       THE FIRST NEGATIVE DVEC(K) MEANS HEAD OF SOMETHING TO DO.
      IF(DVEC(I1).LT.0)GO TO 101
  100 CONTINUE
C       DOES THIS ENTITY HAVE SUB-ELEMENTS?
C       IF NOT, TO 102
  101 IF(DVEC(I1+1).EQ.0)GO TO 102
      DVEC(I1+1)=DVEC(I1+1)-1
C       INDICATE 'WORKING-ON-THIS-ELEMENT'.
      DVEC(ID)=-DVEC(IC)
      GO TO 4
C       ZERO IT IN DVEC
  102 DVEC(I1)=0
      ID=I1-1
C       IF DVEC FINALLY EMPTY, WE'RE DONE.
      IF(ID.EQ.0)RETURN
C       NO MATRIX TO USE? IF NOT, RETURN.
      IF(IC.LE.0)RETURN 1
C       SHOULD CURRENT TMAT MATRIX BE EXPIRED? IF NOT, RETURN
      IF(ID.GE.EXPIRE(IC)) RETURN 1
C       IF SO, ZERO IT OUT
      CURMAT(IC)=0
```

```
      EXPIRE(IC)=C
      IC=IC-1
      IF(IC.LT.1) RETURN 1
C       SET TMAT TO IDENTITY MATRIX
      DO 112 J=1,4
      DC 112 K=1,4
      TMAT1(J,K)=0.
      IF(J.EQ.K)TMAT1(J,K)=1.
 112  CONTINUE
C       IN THIS LOOP, BUILD TMAT WITH THE NECESSARY MULTIPLICATIONS.
      DO 115 J=1,IC
      IARG=IDIREC(CURMAT(J)*2+2)/2+1
      CALL MATMLT(TMAT,TABLE(IARG),TMAT1)
      IF(IC.EC.J)RETURN 1
      DC 115K=1,4
      DO 115 I=1,4
 115  TMAT1(I,K)=TMAT(I,K)
      RETURN 1
C       FOR THE FIRST TIME THROUGH, DVEC(1) WAS SET AT STMT 11 IN DRAW
C       TO THE -(LOCATION IN DIREC OF ENTITY TO BE DRAWN).
C       ISTART IS THE LOCATION IN TABLE OF THIS ENTITY.
C       IGO IS THE TYPE OF ENTITY.
 4    ISTART=IDIREC(-DVEC(INUM)*2+2)
      IGO=IDIREC(-DVEC(INUM)*2+1)
      GO TO(1CCC,2COC,3000,4COC),IGO
C       OBJECT RETRIEVAL - SET UP LIMITS IN 'POINT' FOR THIS OBJECT
 1000 ILOWR=ISTART+3
      IUPPR=ILOWR+(ITABLE(ISTART+1)-1)*2
C-------------------------------------------------------------
      DO 1CC9 J=ILOWR,IUPPR,2
      NDX=ITABLE(J)+1
      KUPPR=NDX+3
      K1=0
C       COPY THE NEXT POINT IN THE OBJECT INTO TPOINT
      DO 1011 K=NDX,KUPPR
      K1=K1+1
 1C11 TPOINT(K1)=POINT(K)
C       INCREMENT POINTS-TO-BE-DRAWN COUNTER
      NPTS=NPTS+1
C       IF IC < 1, PLINT IS UNTRANSFORMED
C       IF IC = 1, POINT IS TO BE TRANSFORMED BY ONLY ONE MATRIX
C       IF IC > 1, PLINT IS TO BE TRANSFORMED BY THE MATRIX PRODUCT IN T
C
C       UNTRANSFORMED POINT
      IF(IC-1)1015,1014,1012
 1015 DO 1013 K1=1,4
 1013 OUTPTS(K1)=TPOINT(K1)
      GO TO 1010
C       POINT TO BE TRANSFORMED BY TMAT
 1012 CALL PTMLT(OUTPTS,TPOINT,TMAT)
      GO TO 1010
C       POINT TO BE TRANSFORMED BY ONE MATRIX ONLY - TAKEN STRAIGHT
C       FROM TABLE.
 1014 IARG=IDIREC(CURMAT(1)*2+2)/2+1
      CALL PTMLT(OUTPTS,TPOINT,TABLE(IARG))
 1010 IF(ABS(OUTPTS(4)).LT..001)GO TO  2CC1
C       INCREMENT OUTPUT VECTOR INDEX, CHECKING FOR OVERFLOW.
      IL=IL+2
      IF(IL.GT.273C)GO TO 1235
```

```
C       HERE'S THE SCALE FACTOR DOINGS ITS THING.
C       IX AND IY ARE SET BY VIEWXY, VIEWXZ, AND VIEWZY. DEFAULT SETTING
C       IS X-Y PROJECTION.
C       CHECK THE DRAWING LIMITS - USED BY OUTPUT FOR SCALING.
      T(IL)=OUTPTS(IX)/OUTPTS(4)
      T(IL+1)=OUTPTS(IY)/OUTPTS(4)
      IF(T(IL).GT.XMAX)XMAX=T(IL)
      IF(T(IL).LT.XMIN)XMIN=T(IL)
      IF(T(IL+1).GT.YMAX)YMAX=T(IL+1)
      IF(T(IL+1).LT.YMIN)YMIN=T(IL+1)
C       TYPE-OF-LINE SWITCH (FIRST ARGUMENT TO "CALL POINTS")
 1009 IT(IL/2+1)=ITABLE(J+1)
C-------------------------------------------------------------
C       ZERO OUT THE LAST ENTRY IN DVEC.
      DVEC(ID)=0
C       DECREMENT DVEC INDEX
      ID=ID-1
      RETURN 1
 1235 IF(BATCH)CALL LOCUNT(2)
      PRINT 1236
 1236 FORMAT('0***DRAWL ERROR - ONLY 2730 POINTS/DRAWING.')
      RETURN
C       ASSEMBLY RETRIEVAL
 3000 NUM=ITABLE(ISTART+3)
C       GET THE # OF ENTITIES IN THIS ENTITY.
      DVEC(ID+1)=NUM
C       SET THE LOOP LIMITS TO DIG THE ENTITIES OUT OF TABLE
      NLOWR=ISTART+5
      NUPPR=NLOWR+NUM-1
C       WE'LL PUT THE POINTER TO EACH ENTITY INTO DVEC
      ID=ID+NUM+1
C       KEEP TRACK OF HOW BIG ID GETS. THIS WAS A DESIGN PARAMETER.
C       BY PRINTING IT OUT (ALONG WITH IDMAX),FIND OUT HOW BIG DIMENSION
C       LIMITS FOR DVEC AND CURMAT SHOULD BE
      IF(ID.GT.IDMAX) IDMAX=ID
      I1=ID+1
      DO 3200  I=NLOWR,NUPPR
      I1=I1-1
C       PICK UP POINTER
 3200 DVEC(I1)=ITABLE(I)
      RETURN 1
C       RETRIEVAL OF TRANSFORMED ENTITIES
 4000 IC=IC+1
      IF(IC.GT.ICMAX) ICMAX=IC
C       PUT POINTER TO THE XFORM IN CURMAT
      CURMAT(IC)=ITABLE(ISTART+1)
      DVEC(ID+1)=1
C       SET UP DVEC WITH PTR TO 'SOURCE' ENTITY
      DVEC(ID+2)=ITABLE(ISTART+2)
C       INDICATE WHEN THE MATRIX JUST SPECIFIED IS TO 'EXPIRE' - THAT IS
C       SHOULD NOT BE MULTIPLIED INTO POINTS ANY MORE IN THIS DRAWING.
      EXPIRE(IC)=ID
      ID=ID+2
      IF(ID.GT.IDMAX) IDMAX=ID
C       IF = 1, THIS IS FIRST MULTIPLICATION LEVEL.
      IF(IC.GT.1)GO TO 4106
      IF(IC.LE.0)RETURN 1
C       GET LOCATION OF VALUES IN TABLE.
      K=IDIREC(CURMAT(1)*2+2)/2
```

```
      DO 4107 J=1,4
      DO 4107 I=1,4
      K=K+1
C       PUT VALUES IN TMAT.
 4107 TMAT(I,J)=TABLE(K)
      RETURN 1
C       IF THIS IS NOT THE FIRST LEVEL DOWN, MULTIPLY THE NEW MATRIX
C       INTO WHAT CAME BEFORE.
 4106 DO 4105 J=1,4
      DO 4105 I=1,4
 4105 TMAT1(I,J)=TMAT(I,J)
      IARG=IDIREC(CURMAT(IC)*2+2)/2+1
      CALL MATMLT(TMAT,TABLE(IARG),TMAT1)
C       WE TRANSFERRED TO HERE ONLY AS A DUMMY, FOR MATRICES. THIS CASE
C       CAN NEVER BE REACHED
 2000 RETURN 1
 2001 IF(BATCH)CALL LOCUNT(3)
      PRINT 2002,OUTPTS(3)
 2002 FORMAT('0***DRAWL ERROR - ATTEMPT TO PLACE EYEPOINT IN SAME X-Y'/
     * '     PLANE AS POINT TO BE DRAWN. EYEPOINT WAS AT Z = ',F10.3)
      CALL ABORTZ(23)
      RETURN 2
      ENTRY VIEWXY
      IX=1
      IY=2
      RETURN
      ENTRY VIEWZY
      IX=3
      IY=2
      RETURN
      ENTRY VIEWXZ
      IX=1
      IY=3
      RETURN
      END
C
C PAD
C
C       THIS FUNCTION SEARCHES THE CHARACTER STRING PASSED AS ARGUMENT
C       FOR AN '@' (AT-SIGN) AND REPLACES IT AND EVERYTHING TO
C       ITS RIGHT WITH BLANKS (UP TO A TOTAL ARGUMENT LENGTH OF
C       EIGHT.) THIS IS DESIGNED TO OVERCOME A SPECIFICATION IN THE
C       FORTRAN IV RUNNING IN MTS - THE LITERALS ARE NOT INITIALIZED
C       AND CAN THUS CONTAIN MISC. LEFT-OVER TRASH - VERY
C       UNDESIREABLE IN COMPARES. THE DETAILS WILL VARY ON DIFFER-
C       ENT MACHINES AND AT DIFFERENT INSTALLATIONS. NOTE THAT ALL
C       THIS DOES IS ELIMINATE THE REQUIREMENT THAT ALL 8 CHARACTERS
C       INCLUDING TRAILING BLANKS BE KEYPUNCHED BY THE USER.
      REAL FUNCTION PAD*8 ( NAM1)
      REAL*8 NAM,NAM1
      INTEGER GET1,VAL/Z00000007C/,TNAM,PLT1,ADROF
      NAM=NAM1
      DO 10 I=1,8
      TNAM=GET1(ADROF(NAM),I-1)
      IF(TNAM.EQ.VAL)GO TO 20
 10   CONTINUE
      PAD=NAM
      RETURN
 20   DO 40 J=I,8
```

```
 4C    CALL PUT1(ADROF(NAM),J-1,' ')
       PAD=NAM
       RETURN
       END
C
C    ARGUMENT CHECKING FUNCTIONS - INTEGER
C
C       FORTRAN IV (G LEVEL) RUNNING UNDER MTS DOES NOT CHECK
C       ARGUMENT TYPES. TO PROVIDE ERROR CHECKING, ALL NUMBERS
C       PASSED AS ARGUMENTS BY THE USER ARE RUN THROUGH THESE FUNC-
C       TIONS. THEY CHECK THE HIGH ORDER BYTES (MAKING, THEREFORE,
C       ASSUMPTIONS ON THE ABSOLUTE SIZES OF THE NUMBERS THEY ARE
      FUNCTION ICHKF(IANS)
C         ICHKF CHECKS INTEGER VALUES FOR INTEGER-NESS.
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 TPTCTR ,OPEN,DIRMAX,DIRPTR,PTCTR,PT
      REAL*8 GENNAM
      DATA MASK /ZFC00CC00/
      EQUIVALENCE (IANS1,ANS1),(IA,ANS2)
C       USERS OF 360'S WHO ARE HAVING TROUBLE IMPLEMENTING THESE
C       FUNCTIONS ARE WELCOME TO ADDRESS THEIR QUESTIONS TO US.
      IANS1=IANS
      IF(IANS1)1,5,2
C       LAND IS A 'BIT-WISE' LOGICAL FUNCION. A BIT IS SET IN
C       THE RETURNED VALUE OF LAND IFF THE CORRESPONDING BIT
C       IS ON IN BOTH ARGUMNENTS.
 1    IF(MASK-LAND(ANS1,MASK))4,3,4
 2    IF(LAND(ANS1,MASK))4,3,4
 5    ICHKF=0
      RETURN
 3    ICHKF=IANS1
      RETURN
 4    ICHKF=ANS1
      IF(BATCH)CALL LCOUNT(1)
      IF(NPRT)PRINT 500,ANS1,ICHKF
 500  FORMAT(' ***WARNING. UNEXPECTED DECIMAL POINT FOUND.',F8.1
     1 ,' CHANGED TO ',I6)
      CALL ABORTZ(8)
      RETURN
C
C  ARGUMENT CHECKING FUNCTIONS - REAL
C
      ENTRY CHECKF(ANS)
C       CHECKF CHECKS FLOATING POINT ARGUMENTS.
      ANS2=ANS
      IF(IA)10,5C,2C
 10   IF(MASK-LAND(IA,MASK))30,4C,30
 2C   IF(LAND(IA,MASK))30,40,3C
 5C   CHECKF=0.
      RETURN
 3C   CHECKF=ANS
      RETURN
 40   CHECKF=IA
      IF(BATCH)CALL LCOUNT(1)
      IF(NPRT)PRINT 510,IA,CHECKF
```

```
 510  FORMAT(' ***WARNING. EXPECTED DECIMAL POINT MISSING.',1X,I6,
     1  ' CHANGED TO',F8.1)
      CALL ABORTZ(9)
      RETURN
      END
C
C    AUTOMATIC OBJECT-ASSEMBLY CLOSER
C
C       CLOSE MERELY CHECKS "OPEN" TO DETERMINE WHAT 'MODE' IS IN EFFECT
C       TYPE 1 ===> OBJECT DEFINITION UNDER WAY
C       TYPE 2 ===> MATRIX COMBINATION (VIA CMBMAT) UNDER WAY
C       TYPE 3 ===> ASSEMBLY DEFINITION UNDERWAY
C       TYPE 4 ===> INIT WAS NOT CALLED PRIOR TO THIS
C       TYPE 0 ===> CLEAR SAILING - SHOULD NEVER BE ZERO IF GOT HERE
      SUBROUTINE CLOSE
      COMMON /TABEL/ IDIREC(800), PCINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      COMMON /CLSE/TC(4,4),XNAME,ISW
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC ,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM,XNAME
      EQUIVALENCE (IPOINT(1),PCINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
      GO TO(100,200,300,400),OPEN
 100  IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 101,DIREC(DIRPTR-2)
 101  FORMAT(' ***WARNING: FAILURE TO FINISH OBJECT ''',A8,1H'/
     1 '    FINOBJ CALLED FOR YOU.')
      CALL FINOBJ
      CALL ABORTZ(11)
      RETURN
 300  IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 301,DIREC(DIRPTR-2)
 301  FORMAT(' ***WARNING: FAILURE TO FINISH ASSEMBLY ''',A8,1H'/
     1 '   FINASM CALLED FOR YOU.')
      CALL FINASM
      CALL ABORTZ(12)
      RETURN
 200  IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 500,XNAME
 500  FORMAT(' ***WARNING: FAILURE TO END CMBMAT(''',A8,2H')/
     1  '   ENDCMB CALLED FOR YOU.')
      CALL ENDCMB
      CALL ABORTZ(13)
      RETURN
 400  IF(BATCH)CALL LOCUNT(2)
      IF(NPRT)PRINT 401
 401  FORMAT(' ***WARNING - FAILURE TO CALL INIT.')
      CALL INIT
      CALL ABORTZ(14)
      RETURN
      END
C
C  MATMLT
C
C       MATMLT MULTIPLIES TWO FOUR-BY-FOUR MATRICES TOGETHER
```

```
C         A = B * C
      SUBROUTINE MATMLT(A,B,C)
      REAL A(4,4),B(4,4),C(4,4)
      DO 1 I=1,4
      DO 1 J=1,4
 1    A(J,I)=0.
      DO 2 K=1,4
      DO 2 J=1,4
      DO 2 I=1,4
 2    A(I,K)=A(I,K)+B(J,K)*C(I,J)
      RETURN
C
C  POINT MULTIPLICATION ROUTINE
C
C         PTMLT MULTIPLIES A ONE-BY-FOUR BY A FOUR-BY-FOUR
C         O = P * F
      ENTRY PTMLT(O,P,F)
      REAL*4 O(4),P(4),F(4,4),TA(4,4)
      DO 10I=1,4
 10   O(I)=0.
      DO 20 I=1,4
      DO 20 J=1,4
 20   O(I)=O(I)+P(J)*F(I,J)
      RETURN
      ENTRY IDENT(TA)
C         IDENT SETS UP A 4-BY-4 IDENTITY MATRIX
      DO 3 I=1,4
      DO 3 J=1,4
      TA(J,I)=0.
      IF(I.EQ.J)TA(I,J)=1.
 3    CONTINUE
      RETURN
      END
C
C  RETRIEVAL SUBROUTINE
C
C         RETREV DIGS OUT OF THE TABLES AND DISPLAYS THE VALUE ASSOCIATED
C         WITH THE NAME GIVEN AS ARGUMENT. FOUR TYPES ARE POSSIBLE:
C         A) OBJECT (TYPE 1)
C         B) ASSEMBLY (TYPE 3)
C         C) TRANSFORMATION MATRIX (TYPE 2)
C         D) TRANSFORMED OBJECT OR ASSEMBLY (TYPE 4)
      SUBROUTINE RETREV(TSTNAM,*)
      COMMON /DATAEL/ IDIREC(800), POINT(5000),TABLE(5000)
      COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
     *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
     *,LNM,NPRT,BATCH
      LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
      INTEGER*2 IPOINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPTR
      INTEGER*4 IDIREC ,OPEN,DIRMAX,DIRPTR,PTCTR,PT,TPTCTR
      REAL*8 DIREC(400),OBJNAM,GENNAM
      EQUIVALENCE (IPOINT(1),POINT(1)),(ITABLE(1),TABLE(1)),
     * (DIREC(1),IDIREC(1))
      REAL*8 TSTNAM,FOUND,XFOUND,NAM,PAD
C         DOES HE BELONG HERE (SHOULD HE HAVE FINISHED SOMETHING UP?)
      IF(OPEN.NE.0)CALL CLOSE
      OPEN=0
      NAM=PAD(TSTNAM)
C         FIND THE NAME. TYPE IN ITYP,LOCATION IN IFIND
```

```
C       IF NOT FOUND, GO TO 998
      CALL FNDNAM(IFIND,NAM,ITYP,&998)
C       INDEX INTO THE NAMES TABLE
      ISTART=IDIREC(IFIND*2+2)
      IGO=ITYP
C       GO TO THE RIGHT PLACE.
      GO TO(1000,2000,3000,4000),IGO
  998 IF(BATCH)CALL LOCUNT(2)
      PRINT 10,NAM,NAM
   10 FORMAT('0***CRAWL ERROR- ''',A8,2H'(,Z16,') NOT FOUND BY RETREV')
      CALL ABORTZ(10)
      RETURN 1
C       GET THE NAME FROM STORAGE
 1000 FOUND=DIREC(ITABLE(ISTART-1))
C       SET UPPER AND LOWER BOUNDS ON DO LOOP TO DIG OUT POINT VALUES
      ILOWR=ISTART+3
      IUPPR=ILOWR+(ITABLE(ISTART+1)-1)*2
      IF(BATCH)CALL LOCUNT(3)
      PRINT 1101,FOUND,ITABLE(ISTART+1)
 1101 FORMAT('OOBJECT ''',A8,''' HAS ',I4,' POINTS:'/
     1' SWITCH   X',9X,'Y',8X,'Z',8X,'W')
C       INDEX THROUGH POINTS TABLE SETTING UP PRINT LINES
      DO 1010 J=ILOWR,IUPPR,2
      NDX=ITABLE(J)+1
      KUPPR=NDX+3
      IF(BATCH)CALL LOCUNT(1)
 1010 PRINT 1102,ITABLE(J+1),(POINT(K),K=NDX,KUPPR)
 1102 FORMAT(3X,I2,4(2X,F7.2))
      RETURN
C       GET THE NAME FROM THE TABLE
 2000 FOUND=DIREC(ITABLE(ISTART-1))
      IF(BATCH)CALL LOCUNT(6)
      PRINT 2102,FOUND
 2102 FORMAT('0TRANSFORMATION MATRIX ''',A8,''' IS:')
C       SET THE UPPER AND LOWER BOUNDS ON PRINT STATEMENT TO
C       DISPLAY THE VALUES IN THE MATRIX
      ILOWR=(ISTART+2)/2
      IUPPR=ILOWR+15
      PRINT 2103,(TABLE(K),K=ILOWR,IUPPR)
 2103 FORMAT(2X,F7.3,2X,F7.3,2X,F7.3, 2X,F7.3)
      RETURN
C       GET THE NAME FROM THE TABLE
 3000 FOUND=DIREC(ITABLE(ISTART-1))
C       NUM IS THE NUMBER OF ELEMENTS IN THE ASSEMBLY
      NUM=ITABLE(ISTART+3)
      IF(BATCH)CALL LOCUNT(2)
      PRINT 3101,FOUND,NUM
 3101 FORMAT('OSIMPLE ASSEMBLY ''',A8,''' HAS ',I3,' ELEMENTS:')
C       SET DO LOOP LIMITS
      NLOWR=ISTART+5
      NUPPR=NLOWR+NUM-1
      DO 3103 K=NLOWR,NUPPR
      IF(BATCH)CALL LOCUNT(1)
C       USE ITABLE(K) AS SUBSCRIPT FOR NAME TABLE
 3103 PRINT 3102,DIREC(ITABLE(K))
 3102 FORMAT(2X,A8)
      RETURN
C       GET THE NAME OF THE ENTITY IN QUESTION
 4000 FOUND=DIREC(ITABLE(ISTART-1))
```

```
C         GET THE NAME OF THE TRANSFORMATION MATRIX ASSOCIATED WITH IT
        XFOUND=DIREC(ITABLE(ISTART+1))
C         GET THE NAME OF THE SOURCE ENTITY
        NAM=DIREC(ITABLE(ISTART+2))
C         PRINT THE WHOLE MESS OUT
        IF(BATCH)CALL LOCUNT(2)
        PRINT 4101,FOUND,NAM,XFOUND
 4101 FORMAT('0',A8,' = 'A8,' * 'A8)
        RETURN
        END
C
C    CMBMAT
C
        SUBROUTINE CMBMAT(OBJNAM)
        COMMON /VAR/ PTMAX,PTPTR,TABMAX,TABPTR,DIRMAX,DIRPTR,
       *PTCTR,PT,ITP,TPTCTR,OPEN,NASM,GENNAM,NSCAL,NMOVE,NSHFT
       *,LNM,NPRT,BATCH
        COMMON /CLSE/TC(4,4),XNAME,ISW
        INTEGER*2 PTMAX,PTPTR,TABMAX,TABPTR
        INTEGER*4 DIRMAX,DIRPTR,PTCTR,PT,TPTCTR,OPEN,Q(2)/'%00@1234'/,
       1 MASK/Z0CCCCAC0/,INCR/Z00C0F6C0/
        LOGICAL NSCAL,NMOVE,NSHFT,NPRT,BATCH
        REAL*8 TRLATE,ROTX,ROTY,ROTZ,CSCALE,PERSPC,NEWNAM,GENNAM,OBJNAM,
       1 ROTXD,ROTYD,ROTZD,ROT1,ROT2,ROT3
        EQUIVALENCE (Q(1),NEWNAM)
        REAL TC(4,4),TB(4,4),TA(4,4)
        IF(OPEN.NE.0)CALL CLOSE
C         DOES HE BELONG HERE? SET OPEN TO CMBMAT STATUS
        OPEN=2
        XNAME=OBJNAM
C         ISW=1 ===> THAT THE SINGLE-TRANSFORMATION-MATRIX-DEFINING
C         FUNCTIONS WHEN USED WILL BE A PART OF A CMBMAT SEQUENCE, AS
C         OPPOSED TO BEING A PART OF A CALL TO TRANSF.
        ISW=1
C         SET UP AN IDENTITY MATRIX
        DO 1 I=1,4
        DO 1 J=1,4
        TC(I,J)=0.
        IF(I.EQ.J)TC(I,J)=1.
  1     CONTINUE
C         TA IS A WORK MATRIX, INTO WHICH THE PIECES FROM CALLS TO
C         THE SINGLE-TRANSFORMATION-MATRIX-DEFINING FUNCTIONS ARE COLLEC
C         ED. TC WILL RECEIVE THE PRODUCT OF TA AND TB (WHERE TB CARRIES
C         THE CUMULATIVE PRODUCT FOR THE ENTIRE SEQUENCE.)
C         TD IS A COPY OF TC, WHICH WILL BE THE ARGUMENT TO NMTRAL,
C         ALONG WITH OBJNAM, WHEN ENDCMB IS FINALLY CALLED.
  2     DO 3 J=1,4
        DO 3 I=1,4
        TC(I,J)=TC(I,J)
        TB(I,J)=TC(I,J)
        TA(I,J)=0.
        IF(I.EQ.J) TA(I,J)=1.
  3     CONTINUE
        CMBMAT=NEWNAM
        RETURN
C         ALL THE SINGLE FUNCTIONS PERFORM IN THE SAME MANNER AS
C         TRLATE. AN EXPOSITION IS PROVIDED FOR TRLATE ONLY.
        ENTRY TRLATE(DX,DY,DZ)
C         IF TRANSF IS USING IT, INITIALIZE TA
```

```
      IF(ISW.NE.1)CALL IDENT(TA)
C       INSERT THE VALUES IN THE APPROPRIATE MATRIX LOCATIONS
      TA(4,1)=CHECKF(DX)
      TA(4,2)=CHECKF(DY)
      TA(4,3)=CHECKF(DZ)
C       IF PART OF A CMBMAT SEQUENCE, TO 99 TO MULTIPLY IT ALL TOGETHER
      IF(ISW.EQ.1)GO TO 99
C       OTHERWISE, WE HAVE TO GENERATE A NAME
      Q(1)=Q(1)+256
      IF(LAND(MASK,Q(1)).EQ.MASK)Q(1)=Q(1)+INCR
C       AND CALL NAMTRA (ACTUALLY AN ALTERNATE ENTRY)
      CALL NMTRA1(NEWNAM,TA,&999)
      TRLATE=NEWNAM
      RETURN
      ENTRY ROTXD(B)
      A=3.14159*CHECKF(B)/180.
      GO TO 50
      ENTRY ROT1(A)
      ENTRY ROTX(A)
 50   IF(ISW.NE.1)CALL IDENT(TA)
      TA(2,2)=COS(CHECKF(A))
      TA(2,3)=SIN(CHECKF(A))
      TA(3,2)=-SIN(CHECKF(A))
      TA(3,3)=COS(CHECKF(A))
      IF(ISW.EQ.1)GO TO 99
      Q(1)=Q(1)+256
      IF(LAND(MASK,Q(1)).EQ.MASK)Q(1)=Q(1)+INCR
      CALL NMTRA1(NEWNAM,TA,&999)
      ROTX=NEWNAM
      ROTXD=NEWNAM
      ROT1=NEWNAM
      RETURN
      ENTRY ROTYD(B)
      A=3.14159*CHECKF(B)/180.
      GO TO 51
      ENTRY ROT2(A)
      ENTRY ROTY(A)
 51   IF(ISW.NE.1)CALL IDENT(TA)
      TA(1,1)=COS(CHECKF(A))
      TA(1,3)=-SIN(CHECKF(A))
      TA(3,1)=SIN(CHECKF(A))
      TA(3,3)=COS(CHECKF(A))
      IF(ISW.EQ.1)GO TO 99
      Q(1)=Q(1)+256
      IF(LAND(MASK,Q(1)).EQ.MASK)Q(1)=Q(1)+INCR
      CALL NMTRA1(NEWNAM,TA,&999)
      ROTY=NEWNAM
      ROTYD=NEWNAM
      ROT2=NEWNAM
      RETURN
      ENTRY ROTZD(B)
      A=CHECKF(B)*3.14159/180.
      GO TO 52
      ENTRY ROT3(A)
      ENTRY ROTZ(A)
 52   IF(ISW.NE.1)CALL IDENT(TA)
      TA(1,1)=COS(CHECKF(A))
      TA(1,2)=SIN(CHECKF(A))
      TA(2,1)=-SIN(CHECKF(A))
```

```
      TA(2,2)=COS(CHECKF(A))
      IF(ISW.EQ.1)GO TO 99
      Q(1)=Q(1)+256
      IF(LAND(MASK,Q(1)).EQ.MASK)G(1)=G(1)+INCR
      CALL NMTRA1(NEWNAM,TA,&999)
      ROTZ=NEWNAM
      ROT3=NEWNAM
      ROTZD=NEWNAM
      RETURN
      ENTRY CSCALE(S)
      IF(ISW.NE.1)CALL IDENT(TA)
      TA(4,4)=CHECKF(S)
      IF(ISW.EQ.1)GO TO 99
      Q(1)=Q(1)+256
      IF(LAND(MASK,Q(1)).EQ.MASK)G(1)=G(1)+INCR
      CALL NMTRA1(NEWNAM,TA,&999)
      CSCALE=NEWNAM
      RETURN
      ENTRY PERSPC(DZ)
      IF(ISW.NE.1)CALL IDENT(TA)
C       CHECK AND SEE IF HE WANTS TO DIVIDE BY ZERO
      IF(ISW.EQ.1.AND.ABS(CHECKF(DZ)).LT..001)GO TO 99
      IF(ISW.EQ.0.AND.ABS(CHECKF(DZ)).LT..001)GO TO 98
      TA(3,4)=-1./DZ
 98   IF(ISW.EQ.1)GO TO 99
      Q(1)=Q(1)+256
      IF(LAND(MASK,Q(1)).EQ.MASK)G(1)=G(1)+INCR
      CALL NMTRA1(NEWNAM,TA,&999)
      IF(LAND(MASK,Q(1)).EQ.MASK)G(1)=G(1)+INCR
      PERSPC=NEWNAM
      RETURN
C       MULTIPLY IT ALL TOGETHER, AND GO BACK TO 2 TO MAKE COPIES
C       AND INITIALIZE THE ARRAYS WHICH NEED INITIALIZING.
 99   CALL MATMLT(TC,TA,TB)
      GO TO 2
 999  RETURN
      END
```

APPENDIX I

Alphabetical List of DRAWL Subroutines

APPENDIX I - INDEX OF DRAWL SUBROUTINES

Name	Definition	References
ABORT	D3	E1, E2, H4
CMBMAT	D4	D5, D6, E1, E2, E3, H32
CSCALE	D4	D5, D6, H34
DISPLA	B22	C2, H18
DRAW	B16	10, 12, B18, B19, B20, B21, B22, C2, C4, C5, E2, E3, F1, F4, H17
ENDCMB	D4	D5, D6, E1, H14
FINASM	B11	2, 6, 8, E1, F4, H14
FINOBJ	B8	2, 5, 12, E1, F1, F4, G4, H13
INCRPT	B7	2, 5, B2, B4, B6, D1, E1, E2, H8
INIT	B1	12, C2, C3, C4, C5, C6, C7, E1, F1, F4, H1
INVA	D1	D1, H8
INVA2	D1	D1, F4, H8
INVR	D1	D1, H8
INVR2	D1	D1, H9
LCOUNT	D7	C7, H5
MOVE	C4	B16, C1, H4
NAMASM	B9	2, 6, 8, 12, B10, E2, E3, F4, H11
NAME	B10	2, 6, 8, 12, E1, E2, F4, G1, H12
NAMOBA	B3	B6, H6
NAMOBJ	B2	2, 5, 12, B3, B4, B6, E2, E3, F1, F4, G1, H5
NAMOBR	B4	B6, B7, E1, H6
NAMTRA	B12	4, 8, 12, B13, D5, D6, E1, E2, F1, F4, G4, H9
NMTRA1	B13	E1, E2, H9
NOMOVE	C4	B16, C1, C5, H4
NOPAGE	C7	C1, H4
NOPRNT	C6	B1, C1, D7, H4
NOSCAL	C2	B16, B18, C1, H4
NOSHFT	C3	B16, B18, C1, H4
PAGE	C7	C1, H4
PERSPC	D4	D5, D6, H34
POINTS	B5	2, 5, 12, B2, B3, B7, D1, E1, E2, F1, F4, G1, G4, H7
PRNT	C6	B1, C1, H4
PRTOUT	B20	11, C2, C3, C4, H18
PRTPLT	B19	11, B20, C2, C3, C4, E2, E3, H18
PRTPUT	B21	11, C2, C3, C4, H18
RETREV	D2	E3, H30
ROTX	D4	D5, H33
ROTXD	D4	D5, D6, H33
ROTY	D4	D5, H33
ROTYD	D5	D5, H33
ROTZ	D5	D5, H33

Name	Definition	References
ROTZD	D5	D5, H33
ROT1	D4	D5, H33
ROT2	D4	D5, H33
ROT3	D5	D5, H33
SCAL	C2	B16, B18, C1, C3, C4, H4
SHFT	C3	B16, B18, C1, C4, H4
SHOW	B18	10, B21, C2, C3, E2, E3, H18
TDUMP	D2	E3, H3
TRANSF	B14	4, 10, 12, D6, E2, F1, F4, G1, G4, H15
TRLATE	D4	D5, D6, H32
VIEWXY	C5	B16, C1, H27
VIEWZX	C5	B16, H27
VIEWZY	C5	B16, H27
VISA	D1	D1, F4, H8
VISA2	D1	D1, H8
VISR	D1	D1, H8
VISR2	D1	D1, H9

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

UNIVERSITY OF MICHIGAN
CONCOMP PROJECT

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

DRAWL 70: A COMPUTER GRAPHICS LANGUAGE

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Technical Report

5. AUTHOR(S) (First name, middle initial, last name)

B. HERZOG and FRED SHADKO

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
August 1970	12	0

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DA-49-083 OSA-3050	Technical Report 30
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

Qualified requesters may obtain copies of this report from DDC.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	ADVANCED RESEARCH PROJECTS AGENCY

13. ABSTRACT

The DRAWL language provides a simple means of defining a graphical composition and specifying operations on it. A catalog of parts is kept; any defined item may be re-used any number of times. Changes in viewing angle, scale, absolute location, and projection are easily affected in three dimensions via homogeneous coordinate projective geometry. Graphical output is available on cathode-ray tube-displays, digital-incremental plotters, and on-line computer line-printers and remote printing terminals.

DD FORM 1 NOV 65 1473

Unclassified

Security Classification

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
graphical languages drafting languages coordinate geometry						

16 April 1971

DRAWL 70: A Computer Graphics Language

B. Herzog and F. Shadko

ERRATA*

Page 8, line 18 should read:

ARG(3,1),ARG(3,2),ARG(3,3),ARG(3,4),

Page 11, 8th line from bottom should read:

'NAMEOFOBJ' String too long

Page A-11, replace a' with z in Figure A-7.

Page B-7, replace the COMMENTS with:

INCRPT executes:

Proper execution requires at least one prior call to POINTS. INCRPT obtains the coordinates of the immediately previously-defined point and computes and stores, via a call to POINTS, the absolute coordinates for the point whose data is specified in the call to INCRPT, i.e. in effect there results:

CALL POINTS(ISMT,PX/PW+X/W,PY/PW+Y/W,PZ/PW+Z/W,1.)

where PX,PY,PZ and PW are the homogeneous coordinates of the previously-defined point.

In the event that no previous point has been defined, an error, then a warning is announced and the values of (0.,0.,0.,1.) are assumed and inserted for the missing data. The warning reads:

***WARNING - THERE IS NOTHING TO USE AS BASE FOR RELATIVE OR INCREMENTAL CALL (0.,0.,0.,) ASSUMED.

Page H-11, delete 12th to 16th lines from bottom, i.e. remove:

```
C     NAMSUB WAS USED ... LEFT IN FOR COMPATIBILITY
```

Page H-13, insert "IF(NUP.EQ.0)GO TO 101" between lines 14 and 15:

```
[L. 18]              NUP=DIRPTR-2
                     IF(NUP.EQ.0)GO TO 101
[L. 19]              DO 100 J=2,NUP,2
```

Page H-13, line 31 should read:

```
101    RETURN 1
```

Page H-14, delete line 19, i.e. remove:

```
IF WE KEEP NAMSUB. WE JUST GOTTA HAVE A FINSUB
```

---

*Underlines indicate corrections.