THE UNIVERSITY OF MICHIGAN
Technical Report 30

DRAWL 70: A COMPUTER GRAPHICS LANGUAGE

F. 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

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTRACT</td>
<td>iii</td>
</tr>
<tr>
<td>TABLE OF CONTENTS</td>
<td>v</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>vii</td>
</tr>
<tr>
<td>PREFACE</td>
<td>ix</td>
</tr>
<tr>
<td>PROGRAMMER'S NOTE</td>
<td>xi</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>2</td>
</tr>
<tr>
<td>DEFINITION OF POINTS</td>
<td>4</td>
</tr>
<tr>
<td>NAMING AND DEFINING SEQUENCE FOR OBJECTS</td>
<td>5</td>
</tr>
<tr>
<td>NAMING AND CREATING SEQUENCE FOR ASSEMBLIES</td>
<td>6</td>
</tr>
<tr>
<td>TRANSFORMATIONS IN DRAWL</td>
<td>8</td>
</tr>
<tr>
<td>PERFORMING A TRANSFORMATION</td>
<td>10</td>
</tr>
<tr>
<td>EXECUTING THE DRAWING</td>
<td>10</td>
</tr>
<tr>
<td>COMMENTS ON DRAWL NAMES AND OTHER PROGRAMMING CONVENTIONS</td>
<td>11</td>
</tr>
<tr>
<td>APPENDIX A - HOMOGENEOUS COORDINATE PROJECTIVE GEOMETRY</td>
<td>A1</td>
</tr>
<tr>
<td>APPENDIX B - DRAWL SUBROUTINE DESCRIPTIONS</td>
<td>B1</td>
</tr>
<tr>
<td>APPENDIX C - DRAWL OUTPUT PARAMETERS</td>
<td>C1</td>
</tr>
<tr>
<td></td>
<td>Table C1 - Default settings for DRAWL RUN Parameters</td>
</tr>
<tr>
<td>APPENDIX D - SPECIAL DRAWL ROUTINES</td>
<td>D1</td>
</tr>
<tr>
<td>APPENDIX E - DRAWL ERROR MESSAGES</td>
<td>E1</td>
</tr>
<tr>
<td>APPENDIX F - SAMPLE DRAWL PROGRAMS</td>
<td>F1</td>
</tr>
<tr>
<td>APPENDIX G - THE DRAWL 70 DATA STRUCTURE</td>
<td>G1</td>
</tr>
<tr>
<td>APPENDIX H - DRAWL 70 FORTRAN SOURCE LISTING</td>
<td>H1</td>
</tr>
<tr>
<td>APPENDIX I - ALPHABETICAL LIST OF DRAWL SUBROUTINES</td>
<td>I1</td>
</tr>
</tbody>
</table>

v
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>The DRAWL Sequence of Operations</td>
<td>1</td>
</tr>
<tr>
<td>Figure 2</td>
<td>Specification of a Rectangular Parallelopiped</td>
<td>3</td>
</tr>
<tr>
<td>Figure 3</td>
<td>DRAWL Hierarchy</td>
<td>3</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Specifications of a Triangle</td>
<td>7</td>
</tr>
<tr>
<td>Figure 5</td>
<td>DRAWL Transformation Matrix Breakdown</td>
<td>9</td>
</tr>
<tr>
<td>Figure A-1</td>
<td>DRAWL Transformation Matrix Breakdown</td>
<td>A-5</td>
</tr>
<tr>
<td>Figure A-2</td>
<td>Translation</td>
<td>A-6</td>
</tr>
<tr>
<td>Figure A-3</td>
<td>$\beta_1$-Rotation</td>
<td>A-7</td>
</tr>
<tr>
<td>Figure A-4</td>
<td>$\beta_2$-Rotation</td>
<td>A-8</td>
</tr>
<tr>
<td>Figure A-5</td>
<td>$\beta_3$-Rotation</td>
<td>A-9</td>
</tr>
<tr>
<td>Figure A-6</td>
<td>The Perspective Transformation</td>
<td>A-10</td>
</tr>
<tr>
<td>Figure A-7</td>
<td>Illustration of Perspective Transformation Matrix</td>
<td>A-11</td>
</tr>
<tr>
<td>Figure F-1</td>
<td>DRAWL Output</td>
<td>F-2</td>
</tr>
<tr>
<td>Figure F-2</td>
<td>DRAWL Output</td>
<td>F-3</td>
</tr>
<tr>
<td>Figure F-3</td>
<td>DRAWL Output</td>
<td>F-5</td>
</tr>
<tr>
<td>Figure G-1</td>
<td>DRAWL Data Structure</td>
<td>G-2</td>
</tr>
<tr>
<td>Figure G-2</td>
<td>Internal Operation of TRANSF</td>
<td>G-3</td>
</tr>
</tbody>
</table>
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 DRAWL language. 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.
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 parallelepiped (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:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAMOBJ</td>
<td>NAME</td>
</tr>
<tr>
<td>NAMASM</td>
<td>FNAME</td>
</tr>
<tr>
<td>POINTS</td>
<td>FINASM</td>
</tr>
<tr>
<td>INCRPT</td>
<td></td>
</tr>
<tr>
<td>FINOBJ</td>
<td></td>
</tr>
</tbody>
</table>

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 Parallelepiped
(DRAWL Specification)

Figure 3 DRAWL Hierarchy
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

\[
X = \frac{x}{w}, \quad Y = \frac{y}{w}, \quad Z = \frac{z}{w}.
\]

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

\[ P1 = (\text{SWITCH1}, X1, Y1, Z1, W1) \]

and

\[ P2 = (\text{SWITCH2}, X2, Y2, Z2, W2) \]

are therefore equivalent if and only if

\[
X1/W1 = X2/W2, \\
Y1/W1 = Y2/W2,
\]
and

\[ \frac{Z_1}{W_1} = \frac{Z_2}{W_2} . \]

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

.....

.....

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

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 \( \frac{X}{N} \) 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(3,3),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@'
'NAMFOBJ' No "@" required since length is eight
'N'OBJECT' See the FORTRAN IV rules for imbedding primes
'@' 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:

'NAMFOBJ' 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:

\[ \text{NAMOBJ ( or NAMOBA, or NAMOBR)} \]
\[ \text{NAMASM} \]
\[ \text{NAMTRA ( or NMTRAL, or CMBMAT)} \]
\[ \text{TRANSF ( first argument only).} \]

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:

\[ \text{CALL INIT} \]
\[ \text{CALL NAMOBJ('l@')} \]
\[ \text{CALL POINTS(1, 2., 3., 2., 1.)} \]
\[ \text{CALL FINOBJ} \]
\[ \text{CALL NAMASM('LASM@')} \]
\[ \text{CALL NAME('L@')} \]
\[ \text{CALL FINASM} \]
\[ \text{CALL TRANSF('l@', 'L@', TRLATE(1.,1.,1.))} \]
\[ \text{CALL DRAW('LASM@')} \]

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

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

We achieve a certain amount of notational elegance by using bilateral 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 \( \alpha \)-system, from another, the \( \beta \)-system, we shall call the \( \alpha \)-system axes, \( \alpha_1 \), \( \alpha_2 \) and \( \alpha_3 \), and the \( \beta \)-system axes, \( \beta_1 \), \( \beta_2 \) and \( \beta_3 \).

Further, rather than write out the row vector of the coordinates, we can designate a coordinate in the \( \alpha \)-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 \( \alpha \)-system to a point in the \( \beta \)-system is given by:

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

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

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

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

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

\[ M_{\beta\alpha} = M_{\alpha\beta}^{-1} \]

and

\[ M_{\alpha\beta} = M_{\beta\alpha}^{-1} \]

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 $\alpha$-system has been moved with respect to the $\beta$-system. The corresponding axes remain parallel, but the origin of the $\alpha$-system has been displaced by

$$\begin{bmatrix} \Delta X \\ \Delta Y \\ \Delta Z \end{bmatrix} \text{ in ordinary coordinates,}$$

or

$$\begin{bmatrix} h\Delta X \\ h\Delta Y \\ h\Delta Z \\ h \end{bmatrix} \text{ in homogeneous coordinates.}$$

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 $\alpha$ axes will be rotated with respect to the $\beta$ 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 $\alpha_1$ and $\beta_1$ axes coincide and we rotate the $\alpha$-system with respect to the $\beta$ system through an angle $\theta$ 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 $\beta_2$ axis and rotation about the $\beta_3$ 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 $P_1$ in the $y-z$ plane, Figure A-7. The projected point is $P'_1$.

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,

\[ M_{\alpha\beta} = \begin{bmatrix} h \\ \sim \\ \sim \\ \sim \\ \sim \\ \sim \\ h \Delta X & h \Delta Y & h \Delta Z & h \end{bmatrix} \]

Figure A-2 Translation.
Figure A-3 Rotation about the $\beta_1$ axis.

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

Figure A-4 Rotation about the \( \beta_2 \) axis.
\[ M_{\alpha\beta} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

Figure A-5 Rotation about the \( \beta_3 \) axis.
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
NAME: INIT

FUNCTION:

INIT

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 per-
formed 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 ('QSVNUML')
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 NAMOBR

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.,0.) ASSUMED.

EXAMPLE: CALL NAMOBR ('OBJECT #1')
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.

**FORTRAN IV:**

CALL POINTS (ISWIT, X,Y,Z,W,&L)

**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,&l)

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: &l-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:

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

CALLING SEQUENCE:
PORTTRAN 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 ('@', &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 &l)

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:
&l - 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 discernable. 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: NMTRAL

FUNCTION: See NAMTRA

CALLING SEQUENCE: FORTRAN IV
CALL NMTRAL (TNAME, E, &l)

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 NMTRAL ('TR2@', ARRAY,&39)
NAME: TRANSF

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

CALLING SEQUENCE: 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 transform-
ation 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, &0)
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 DRAW
&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 EYEPONT IN SAME X-Y PLANE AS POINT TO BE DRAWN . THE EYEPONT 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. DSNR 8 is used by DRAW to build a file of CalComp plotter commands. If DRAW is called, DSNR 8 must not be referenced by the user's program. See the MTS FORTRAN Users' Guide for more information on DSNR's ( Data Set
REFERENCE NUMBERS.

EXAMPLES:

CALL DRAW('OBJ@',&l,&r)
CALL DRAW('ASM@',&oo)
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,&8)
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 DRAWL, substituting "PRTOUT" for "DRAWL"

COMMENTS

This is the least expensive entity output method. Also see the comments for PRTILT and DRAWL. The values are scaled to standard CalComp output size, 23" by 49". Also see the comments under DRAWL.

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 5 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@',1)
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

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default value</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drawing Scaling</td>
<td>SCAL</td>
<td>C-2</td>
</tr>
<tr>
<td>Drawing Shifting</td>
<td>SHIFT</td>
<td>C-3</td>
</tr>
<tr>
<td>Paper Advance</td>
<td>MOVE</td>
<td>C-4</td>
</tr>
<tr>
<td>Viewing Plane</td>
<td>VIEWXY</td>
<td>C-5</td>
</tr>
<tr>
<td>Descriptive Printing</td>
<td>Batch: PRNT</td>
<td>C-6</td>
</tr>
<tr>
<td></td>
<td>Interactive:</td>
<td>C-6</td>
</tr>
<tr>
<td></td>
<td>NOPRNT</td>
<td></td>
</tr>
<tr>
<td>Pagination</td>
<td>Batch: PAGE</td>
<td>C-7</td>
</tr>
<tr>
<td></td>
<td>Interactive:</td>
<td>C-7</td>
</tr>
<tr>
<td></td>
<td>NOPAGE</td>
<td></td>
</tr>
<tr>
<td>Error Handling:</td>
<td>RETURN</td>
<td>D-3</td>
</tr>
</tbody>
</table>
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.
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@')
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('B@')
    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 suppressed by NOPRNT.
           Warning comments are suppressed. The default
           is restored by each call to INIT. Setting
           PRNT on turns NOPRNT off. Setting NOPRNT
           on turns PRNT off.
<table>
<thead>
<tr>
<th>NAME:</th>
<th>PAGE, NOPAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FUNCTION:</td>
<td>Turn pagination* on (PAGE) and off (NOPAGE)</td>
</tr>
<tr>
<td>CALLING SEQUENCE:</td>
<td>FORTRAN IV</td>
</tr>
<tr>
<td></td>
<td>CALL PAGE</td>
</tr>
<tr>
<td></td>
<td>CALL NOPAGE</td>
</tr>
<tr>
<td>DEFAULT SETTING:</td>
<td>Batch Users - PAGE is on.</td>
</tr>
<tr>
<td></td>
<td>Terminal Users - NOPAGE is on.</td>
</tr>
<tr>
<td>COMMENTS:</td>
<td>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.</td>
</tr>
</tbody>
</table>

* 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
Several special calls for POINTS and INCRPT are provided.

I. Absolute Coordinates

Special POINTS entry:  

\[ \text{VISA}(X,Y,Z) \]  
\[ \text{INVA}(X,Y,Z) \]  
\[ \text{VISA2}(X,Y) \]  
\[ \text{INVA2}(X,Y) \]

Equivalent to calling:

\[ \text{POINTS}(1,X,Y,Z,1.) \]  
\[ \text{POINTS}(0,X,Y,Z,1.) \]  
\[ \text{POINTS}(1,X,Y,0.,1.) \]  
\[ \text{POINTS}(0,X,Y,0.,1.) \]

II. Incremental Relative Coordinates

Special INCRPT entry:  

\[ \text{VISR}(X,Y,Z) \]  
\[ \text{INVR}(X,Y,Z) \]  
\[ \text{VISR2}(X,Y) \]  
\[ \text{INVR2}(X,Y) \]

Equivalent to calling:

\[ \text{INCRPT}(1,X,Y,Z,1.) \]  
\[ \text{INCRPT}(0,X,Y,Z,1.) \]  
\[ \text{INCRPT}(1,X,Y,0.,1.) \]  
\[ \text{INCRPT}(0,X,Y,0.,1.) \]

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

EXAMPLES:  

\[ \text{CALL VISA}(3.,4.,9.,&6) \]  
\[ \text{CALL VISR2}(A,12.) \]  
\[ \text{CALL INVA2}(6.,7.,&1) \]  
\[ \text{CALL INVR2}(2.15,\text{COS(ALPHA)},&10) \]  
\[ \text{CALL INVA}(10.,A,A) \]
NAME: RETREV

FUNCTION: Print the values of the DRAWL trans-
formation 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.

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

RETURN CODES:

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 com-
plete 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 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 transformation matrix defining functions are:

<table>
<thead>
<tr>
<th>NAME</th>
<th>EFFECT</th>
<th>ARGUMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRILATE(X,Y,Z)</td>
<td>Translation</td>
<td>The X, Y, and Z values (in inches) for the translation row of the transformation matrix.</td>
</tr>
<tr>
<td>CSCALE(W)</td>
<td>Scaling</td>
<td>The sixteenth entry in the transformation matrix.</td>
</tr>
<tr>
<td>PERSPC(S)</td>
<td>Perspective</td>
<td>The location on the Z axis (in inches) of the eyepoint.</td>
</tr>
<tr>
<td>ROTX(R)</td>
<td>Rotation about the X-axis</td>
<td>The amount of rotation (in radians)</td>
</tr>
<tr>
<td>ROT1(R)</td>
<td>Rotation about the X-axis</td>
<td>The amount of rotation (in radians)</td>
</tr>
<tr>
<td>ROTXD(D)</td>
<td>Rotation about the X-axis</td>
<td>The amount of rotation (in degrees)</td>
</tr>
<tr>
<td>ROTY(R)</td>
<td>Rotation about the Y-axis</td>
<td>The amount of rotation (in radians)</td>
</tr>
<tr>
<td>ROT2(R)</td>
<td>Rotation about the Y-axis</td>
<td>The amount of rotation (in radians)</td>
</tr>
</tbody>
</table>
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:

<table>
<thead>
<tr>
<th>TRLATE(X,Y,Z)</th>
<th>CSSCALE(W)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. 0. 0. 0.</td>
<td>1. 0. 0. 0.</td>
</tr>
<tr>
<td>0. 1. 0. 0.</td>
<td>0. 1. 0. 0.</td>
</tr>
<tr>
<td>0. 0. 1. 0.</td>
<td>0. 0. 1. 0.</td>
</tr>
<tr>
<td>X Y Z 1.</td>
<td>0. 0. W</td>
</tr>
</tbody>
</table>

ROT(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. |

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

| COS(R) SIN(R) 0. 0. |
| -SIN(R) COS(R) 0. 0. |
| 0. 0. 1. 0. |
| 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 1
CALL ENDCMB 2
3

CALL NAMTRA("A@",l.,0.,0.,0.,
0.,l.,0.,0.,
0.,0.,l.,0.,
1.,2.,3.,1.)
(B) CALL CMBMAT('B@')
    CALL CSCALE(5.)
    CALL PERSPC(10.) and
    CALL ENDCMB

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@','OBJL@','ZROT@')

is equivalent to:

    CALL TRANSF('OBJ@','OBJL@', 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 NOPRINT is on (whether explicitly set or by default). At most 56 lines are allowed on a page. CALL LCOUNT(56) 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" SPECIFICATION.  
   (POINTS,INCRPT)

2. OVER 200 NAMES WERE SPECIFIED. TERMINATING VIA "ABORT" SPECIFICATION.  
   (NAMOBJ,NAMASM,NAMTRA,NMTRL1,CMBMAT,TRANSF)

3. OVER 5000 LINKAGES WERE SPECIFIED. TERMINATING VIA "ABORT" SPECIFICATION.  
   (NAMTRA,NAMOBJ,TRANSF,POINTS,INCRPT,NMTRL1)

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

5. 'STRING' ALREADY USED AS OBJ/ASM NAME.  
   (NAMTRA,NMTRL1,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=X.
    ([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 DRAW PROGRAM PRODUCES A DRAWING OF A RECTANGULAR
C PARALLELOGRAM (AS IN FIG. 2)
C
C CREATE AN OBJECT
C
CALL NAME('RECPARA')
C
C MOVE THE PEN TO THE ORIGIN
C
CALL POINTS(0,0,0,0,1)
C
C DRAW LINES 1-8 (AS IN FIG. 2)
CALL POINTS(1,0,0,0,1)
CALL POINTS(1,1,0,0,1)
CALL POINTS(1,2,0,0,1)
CALL POINTS(1,3,0,0,1)
CALL POINTS(1,4,0,0,1)
CALL POINTS(1,5,0,0,1)
CALL POINTS(1,6,0,0,1)
CALL POINTS(1,7,0,0,1)
CALL POINTS(1,8,0,0,1)
C
C LINE 9 (FIG. 2) IS "INVISIBLE" - THE SWITCH IS ZERO
CALL POINTS(0,0,0,0,1)
CALL POINTS(1,0,0,0,1)
CALL POINTS(1,1,0,0,1)
CALL POINTS(1,2,0,0,1)
CALL POINTS(1,3,0,0,1)
CALL POINTS(1,4,0,0,1)
CALL POINTS(1,5,0,0,1)
CALL POINTS(1,6,0,0,1)
C
C FINISH THE OBJECT
C
CALL NAME()
C
C DRAW IT. SEE FIG. F-1
C
CALL PAINT('RECPARA')
C
C THAT WAS THE X-Y PROJECTION OF THE OBJECT CREATED IN ITS
C ORIGINAL POSITION. NOW LET'S ROTATE IT AS IN FIG. 2
C
C
C ROTATE IT ABOUT THE Y-AXIS, USING FOUR Calls FOR THE
C FORTRAN CALL STATEMENT MAKES THE MATRIX EASIER TO READ
C
C IF THIS WAS ALL ON ONE CARD, NOTE THAT FORTRAN'S SIN AND COS
C FUNCTIONS USE RADIANS AND NOT DEGREES.
C
CALL NAME('RECPARA',CLS(RADS30),0,-SIN(RADS30),0,1)
CALL NAME('RECPARA',SIN(RADS30),0,0,COS(RADS30),0,1)
CALL TRANSF('RECPARA','RECPARA','RKA')
RADS30=6.2836/360.*30.
C
C NOW ROTATE THE ROTATED OBJECT ABOUT THE X-AXIS
C
CALL NAME('RKA',1,0,0,0,0,0,0,0,0,0,1)
CALL NAME('RKA',0,CLS(RADS45),SIN(RADS45),0,0,0,1)
CALL NAME('RKA',0,-SIN(RADS45),COS(RADS45),0,0,0,1)
CALL TRANSF('RECPARA','RECPARA','RKA')
C
C DRAW THE THREE ROTATED OBJECT (SEE FIG. F-2)
CALL DRAW('RECPARA')
C
STOP
END
C USING THE FCOUNT TV CC STATEMENT, WE CAN DRAW A
C CIRCLE: (X-2)^2 + (Y-4)^2 = 9

X = 2
Y = 4
F = 3

C WE WILL USE 30 LINE SEGMENTS = 31 POINTS.

T = 1.24/30
TS = -T
CALL INIT
CALL NAME('CIRC1')
C MOVE PEN TO X = 5, Y = 7
CALL INV2(5.0, 7.0)
CC 1000 CALL POINTS(1, X + TS * COS(10 * TS), Y + TS * SIN(10 * TS)), 1)
CALL FINISH
C NOW A SPIRAL
CALL NAME('SPIRALA')
CALL POINTS(5, 1.0, 1.0, 1.0)
C LET'S MAKE FOUR LEAVES, TWO INCHES APART.
CELZ = 2 * 120
Z1 = -CELZ
CC 1010 CALL INV2(CELZ, TS)
1S = TS + T
CC 1010 CALL VIS2(X + TS * COS(TS), Y + TS * SIN(TS), Z1 + CELZ, 1)
CALL FINISH
C LET'S PUT CIRCLES ON THE ENDS OF THE SPIRAL AS ENDCAPS.
C THE FIRST CAME IS IN PLACE AT X = 2, Y = 4, Z = 0. LET'S MOVE
C A COPY TO Z = 6.
C CALL NAME('ZMCVEA'), 1.0, C, C, C, C
1
2
C, C, C
2
C, C, C, C
C
C PERFORM THE TRANSFORMATION
CALL TRANSF('NEWCIRC1', 'CIRC1', 'ZMCVE1')
C PUT IT ALL TOGETHER INTO AN ASSEMBLY
CALL NAME('CLTFLTA')
CALL NAME('CIRC1')
CALL NAME('SPIRAL1')
C CALL NAME('NEWCIRC1')
Call FINISH
C SCALE IT TO FIT AN E BY EC FACE, AND ROTATE IT
CALL TRANSF('ULFTLE1', 'CLTFLTA', 'RTCYL(-121)
CALL TRANSF('CLTFLTA', 'ULFTLE1', 'SCALE(2.1)
C (SEE PAGE 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 EYEFLINT AT Z = -20, WE ENTER: -1.200
C (SEE APPENDIX A).
C CALL NAME('PERA'), 1.0, C, C, C, C
1
2
C, C, C
2
C, C, C
C
C CALL TRANSF('CLTFLTE1', 'CLTFLTA', 'PERA')
C AND SO WE ARE READY TO DRAW IT. (SEE FIG. 1-3)
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 Gl. DRAWL Data Structure.
Figure G-2. Internal Operation of TRASE.
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 52B. "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
BLK D4TA
      AT abl CNtain 4, N4M3, AND L1N4G4KE TABL3S
      Curren tly, the linkage table also contains the values
      for transformation matrices and switch settings for points
      C CMMCN /ATAEL/ CI4REC(800), PCINT(5000), TABLE(5000)
C      
C      VAR CNtain 4 PARAMETERS FOR THE TABL3S IN ATABL
C      PTMAT IS THE MA3XIMUM N4M3BER OF PCINTS ALLOW3D
C      PTPTR IS THE INDEX FOR ARRAY PCINT
C      T14BMAX IS THE L4PPER D1MENSION F0R ARRAY TABL3
C      T14BPTR IS THE INDEX INTO TABL3
C      C1RMAX IS THE L4PPER LIMIT F0R ARRAY CI4REC
C      CI4PRTR IS THE INDEX INTO CI4REC
C      P1CTCR IS THE NUMBER OF PCINTS IN AN OBJECT DEFINITION
C      PT IS AN INTERIM COUNTER FOR PCINT DEFINITION
C      IT4P IS ALSO AN INTERIM COUNTER FOR PCINT DEFINITION
C      TP1CTCR C4UNT5 THE T4TAL NUMBER OF PCINTS DEFINED IN A RUN
C      OPEN IS USE C TC CHECK THAT ENTITIES, CNCE OPENED, ARE
C      CLSEC, AND THAT INIT IS CALLED
C      N4SM CN4NT5 THE NUMBER OF ENTITIES INCCLUD3D IN AN
C      A3SEMBLY DEFINITION UN3R N4SM
C      G4N4NAM IS THE CHARACT3R STRING LNSEC F0R GENERAT3D
C      N4M3S F0R USE BY TRANSF
C      N4SCL 3S T3E GL3BAL SWITC3H SET BY SCAL AND N0SCL4
C      N4MOV3 IS T3E GL3BAL SWITC3H SET BY MOVE AND N0MOV3
C      NSHFT 3S T3E GL3BAL SWITC3H SET BY SHFT AND N0SHFT
C      LAM KEEPS TRACK OF THE NUMBER OF DRAWINGS PER RUN
C      NPR3T IS T3E GL3BAL PARAMETER SET BY PRN4T AND N0PRN4T
C      B4TCH IS T3E GL3BAL SWITC3H SET BY PA4GE AND N0PA4GE
C CMMCN /VAR/ PTMAX, PTPTR, T14BMAX, T14BPRTR, CI4PRTR, *P1CTCR, PT, IT4P, TP1CTCR, OPEN, N4SM, G4N4NAM, NS4CL, N4MOV3, NSHFT
C*, LN4, N4PR3T, B4TCH
C      CLS3 H4S TC C0 WITh ERRCR CHECKING IN SUBROUTINE CLOSE
C      AND FINISH3S THE CM4MAT...ENC4MB SEQUENCE BY CALLING
C      N4TR4L 3S TC AS ARGUMENT
C CMMCN /CLS3/TIC(4,4), XN4ME, ISW
C      INTEGR3 2 IP1CINT(10000), IT4BLE(10000), PTMAX, PTPTR, T4BBMAX, T4BBPTR
C      INTEGR3 4 IL14REC, OPNE, CI4MAX, DIRPRT, PTCTR, PT, TP4CTCR
C      REAL 8 DIREC(40), O4JNAM, G4N4NAM, XN4ME
C      DATA PTPTR, TABPTR, DIRPRT/2, 3, 2/, TP1CTCR/1, OPEN/4/, N4SM/0/
C      DATA PTMAX, T14BMAX, CI4MAX/5000, 10000, 40/, G4N4NAM/"%G4NAM000/
C      S4MESICAL N4SCL4, NS4CVE, NSHFT, N4PR3T/FL4SE...TRUE...FL4SE...TRUE/
C      IN TH3E IM4PLEMENTAT3ON, TABL3 AND IT4BLE ARE OVEp4L4D;
C      DIREC AND CI4REC; AND TABL3 AND IT4BLE. CN DIF3REN4T
C      MACHINES THIS IS VARI4BLY EFFICIENT.
C      EQUIVALENCE ((IP1CINT(1), POINT(1)), (IT4BLE(1), TABLE(1)),
C      * (CI4REC(1), IC14REC(1))
C      END
C      INIT
C SUBROUTINE INIT
C      INIT INITIALIZ3S ALL VARI4BLES EXCEPT OPEN (WHICH IS
C      SET AT LAC4 LIME), IT DOES NOT Z3RC OUT ANY TABL3S, BUT
C      MERELY RESETS ALL THEIR PCINT3RS. GL3BAL VARIAB3LES ARE
C      ALSO RESET VIA A CALL TC 'SPEC'.
C CMMCN /ATAEL/ CI4REC(800), PCINT(5000), TABLE(5000)
CCCOMMON /VAR/ PTPMAX, PTPTR, TAMEX, IAEPTR, CIRMAX, DIRPTR,
  *PTCTR, PTT, TPTCTR, CPGN, NASM, GENAM, NSCAL, NMOVE, NSHFT
 *, LAM, NVRT, Batch

C
  DRCWC3 (RCRAV (CMMCA)) IS AN AREA USE FOR COMMUNICATION
C
  BETWEEN THE THREE PIECES OF THE CUPUT ROUTINES. IT PASSES
C
  LIMITS AND HAS THE CUPUT ARRAYS.
C
  COMMON /CRWCCM/CVEC(50), CURMAT(50), EXPIRE(50), IIC, IC,
1 NPTS, LMAX, LMIN, XMAX, XMIN, YMAX, YMIN, TMAX, T(5460), LT(2730), IL
 INTEGER*2 CVEC, CURMAT, EXPIRE, IC, IC, IT, LEN, LINE(48)
 LOGICAL NSCAL, NSVE, NSHFT, NFRT, EATCH
 INTEGER*2 IFCINT(100CC), ITABLE(100CC), PTPMAX, PTPTR, TABMAX, TABPTR
 INTEGER*4 CPEN, CIRMAX, DIRPTR, PTPCTR, PT, TPTCTR, ADROF, ARRAY(5)
 EXTERNAL GETFC, FREEFD
 REAL*8 GENAM, PACE, NGAM, GNAMEC, C, CIREC(400)
 EQUIVALENCE (IFCINT(1), POINT(1)), (ITABLE(1), TABLE(1)),
 * (CIREC(1), CIREC(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 WANT TO IMPLEMENT THESE FEATURES CAN SPLIT
C THEM OFF EASILY.
C
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 SUSPENDED CR BATCH MODE (THE
C BRANCH TO 1198 IS TAKEN)
C
CALL CREPLY(11198)
BATCH=.FALSE.
NVRT=.FALSE.
GC TO 11955
1198 BATCH=.TRUE.
NVRT=.TRUE.

C IF THE USER IS ON A TERMINAL, HE WILL GET THIS ONE HEADER
C ONLY CACE.
1199 IF(BATCH) GC TO 1254

PRINT 1200
1200 FORMAT('** ** ** CRAWL - AUG 10, 197C VERSION ** ** **')
PRINT 1202
1202 FORMAT('** CLSERS WHO HAVE NOT COPIED SAVE:DRAWLNEWS SINCE**')
PRINT 1203
1203 FORMAT('** AUG 10, 1970 FILE ** ** SCLY SAVE: CRAWLNEWS**')
C
C IOTIME RETURNS THE USER I.O. NUMBER IN ARRAY(5), THE TIME
C OF DAY IN ARRAY(2) AND (3), AND THE DATE IN ARRAY(4) AND (5)
C CALL IOTIME(ARRAY)
C
PRINT 1201, ARRAY(5), (ARRAY(1), I=1, 4)
1201 FORMAT('** OLDER ', '4,' INITIATED DRAWL AT ', '24A,' ON ', '24A//')
C
GC TO 1256
C
C CHKFILE 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 CHKFILE('SAVE:NEWS', 61258)
C
C RCALL AND GETFC "GET" THE FILE, WHICH CAN BE READ BY
C SUBROUTINE REAL AND PRINTED BY SUBROUTINE SPRINT.
CALL RCALL(CETFIC, 2, 0, ACREF('SAVE:NEWS'), 1, IFDUB)
C
C LCOUNT(57) FORCES A PAGE EJECT.
CALL LCOUNT(57)

C
1250 CALL READ(LINE, LEN, C, IDLCM, IFCUB, 61255)
CALL SPRINT(LINE, LEN, C)
GC TO 125C

C RCALL AND FREEFD 'FREE UP' THE NEWS FILE
1255 CALL RFREE(1,FREEC,0)
1256 CALL RCEQDEF(1,IFCUP,C)
1257 ANOTHER PAGE EJECT. THE NEWS LETTER IS ON A PRIVATE PAGE
1258 CALL LCOUNT(57)
1259 SET ALL THE DEFAULT VALUES:
1260 PTEMPR = 1
1261 FMX = 4955
1262 TMX = 559C
1263 DMX = 35E
1264 TBPTR = 3
1265 DIRPTR = 1
1266 LM = 0
1267 IFLG = 0
1268 TPTCTR = 1
1269 CPEN = 0
1270 GEN = NA
1271 CALL VIEWXY
1272 T(1) = 0.
1273 T(2) = 0.
1274 IT(1) = 0
1275 NMOVE = .TRUE.
1276 NSHT = .FALSE.
1277 NSCAL = .FALSE.
1278 RETURN

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.
C IF(BATCH) CALL LCOUNT(57)
C IF(BATCH) CALL LCOUNT(2)
C PRINT 760

780 FFORMAT(' ' ' CRAWL TRACEBACK FACILITY' /* ' ' ')
781 ILIM = DIRPTR - 2
782 DC 783 I = 2, IILIM, 2
783 CALL RETREV(CIREC(I1))
784 RETURN
END

C SPECIAL RCLTINES
C
SUBROUTINE SPEC
C SPEC SETS THE DEFAULT GLOBAL SWITCHES WHICH MAKE
C LIF A LITTLE EASIER
C
COMMEN /VATREL/ ICIRC(800), FCINT(55CC), TABLE(5000)
COMMEN /VARI/ FMX, TMPTR, TMax, TAEPR, TMX, DMX, TBPTR,
*PTCTR, PT, IT, PTCTR, CPEN, NAM, GEN, NAM, NSCAL, NMOVE, NSHT
*, LNM, ARP, BACH
C LOGICAL NSCAL, NMOVE, NSHT, NFR, EATCH, LRET, .FALSE.,
* INTEGER = 2 IPCINT(10CC), ITABLE(10CC), FMX, TMPTR, TMax, TBPTR
* INTEGER = 4 CPEN, CPEN, DMX, TBPTR, PTCTR, PT, TPTCTR, ERRCNT(30)/30*0/,
1 ARRAY(5), NUMCNT(3)/LOCC, 2CC, 50CC/, FAGES/0/, NLINE/2/
REAL*8 ERRTP(3)' POINTS ' . NAM ' NAMES ' ' LINKAGES/
REAL*8 ARG, GEN, EATCH, TITLE(4) /* RETURN ' ',
1 ' QUIT ' . ' SYSTEM ' . ' MS ' . ' CIREC(400)

EQUIVALENCE (IPCINT(1), IPCINT(1)), (ITABLE(1), TABLE(1)),
* (DIREC(1), ICIREC(1))
C 'PAGES' COUNTS THE PAGES PRINTED, NLINE IS THE CURRENT
LINECNT FOR THIS PAGE, LEVABT IS THE NUMBER OF ERRORS 
TO TOLERATE BEFORE STOPPING THE JOB, LERCNT IS THE 
NUMBER OF ERRORS (AND WARNINGS) ENCOUNTERED THUS FAR, 
LABORT IS THE KEY SET FOR TYPE OF TERMINATION REQUESTED.

PAGES=0
NLINEx=2
LEVABT=100
LERCNT=0
LABORT=1
GC TO 1

NOSCAL, NCMOVE, AND NCSHFT CONTROL DRAWING SCALING, MOVING, 
AND SHIFTING, IN CONJUNCTION WITH SCAL, MCVE, AND SHFT.
PRNT AND NCPRT CONTROLS WHETHER TRACE COMMENTS ARE TO BE 
PRINTED OR NOT. PAGE AND NPGAGE TURN PAGINATION 
CN AND OFF.
ENTRY NOSCAL
NOSCAL.=.TRUE.
GO TO 1
ENTRY NCMOVE
NMOVE.=.FALSE.
GO TO 1
ENTRY NCSHFT
NSHFT.=.TRUE.
GO TO 1
ENTRY SCAL
NSCAL.=.FALSE.
GO TO 1
ENTRY MCVE
NMOVEx.=.TRUE.
GO TO 1
ENTRY SHFT
NSHFT.=.FALSE.
GO TO 1
ENTRY PRNT
NPRT.=.TRLEx.
GO TO 1
ENTRY PAGE
BATCH.=.TRUE.
GO TO 1
ENTRY NPGAGE
BATCH.=.FALSE.
GO TO 1
ENTRY NCPRT
NPRT.=.FALSE.
GO TO 1

ABORT SETS LABORT TO THE TYPE OF EXIT TO BE TAKEN ON 
ENCOUNTERING THE N-TH ERROR (OR WARNING).
EXIT IS VIA ROUTINES 'SYSTEM' (STCF AND UNLOAD DRAWL), 
'MTS' (STOP BLT DC ACt UNLOAD DRAWL), AND 'QUIT' (STOP 
DRAWL AND IMMEDIATELY SIGN-OFF THE SYSTEM. DO NOT FINISH 
THE JCB.)
ENTRY ABORT(ARG,N)
LERCNT=C
LREx.=.FALSE.
GO 17 I=1,4
IF IPADJ(ARG).EQ. TITLE(I)) GC TO 12
10 CONTINUE
PRINT 11,ARG
11 FLMAT(' ***DRAWL WARNING - "",A8,"") IS AN ILLEGAL ARGUMENT '/
1 FOR ABCRT, "RETURN" ASSUMED.
LABCRT=1
GC TO 13
LABRT=1
LEVABT=A
IF(I.NE.1)LRET=.TRUE.
GC TO 1
C ABRRTZ CUNTS (WITH LERCNT) THE NUMBER OF ERRORS AND
C WARNINGS ENCOUNTERED.
ENTRY ABRRTZ(K)
IF(K.GT.C)CC TO 8
PRINT 7,NUMCAT(K+4),ERFYP(K+4)
7 FORMAT('**CRAWL ERROR - OVER ',(,A,' ',AB,' WERE SPECIFIED.')/
14X,'TERMINATING VIA "ABCRT" SPECIFICATION')
GC TO 6
8 LERCNT=LERCNT+1
ERRCNT(K)=ERFCNT(K)+1
IF(LERCAT.LE.LEVABT) GC TO 1
PRINT 5,ERRCAT
C ULTIMATELY, CRAWL 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('CRAWL ERROR SUMMARY:')/3013)
6 IF(LABRT.NE.1)IF(LABRT-3)12,2,4
GO TO 1
2 CALL QUIT
3 CALL SYSTEM
4 CALL MTS
GC TO 1
C LCCNT ("LINE CUNT") MAINTAINS A CCNT 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 CCNT RESTARTED. "PAGES" KEEPS TRACK OF THE PAGE
C NUMBER WHILE ITIME GETS THE TIME OF DAY FOR EACH PAGE.
ENTRY LCCNT(M)
A=M
IF(INLINE+.GT.5E1)GC TO 58
NLINE=NLINE+A
GO TO 1
58 CALL ITIME(ARRAY)
PAGES=PAGES+1
PRINT 59,ARRAY(5),(ARRAY(I),I=1,4),PAGES
IF(N.EQ.57)N=0
NLINE=2+N
GO TO 56
PRINT 56
56 FORMAT('C')
59 FORMAT('CRAWL',EX,'G-LEVEL PROCESSOR (MODEL 08:01')',6X,
1 USER',A4,EX,'TIME',2A4,6X,'DATE',2A4,12X,'PAGE',I3)
1 RETURN
END
C
C NACBJ
C
C SUBROUTINE NACBJ(UBJNAM,*)
COMMON /ATABL/,ICIREC(1800),PCINT(500),TABLE(5000)
COMMON /VAR/,PMAX,PTPTR,TAEMAX,TABPTR,DIRMAX,DIRETR,
*PCTCR,PT,ITP,IPCTCR,CREA,NSM,GENNAM,NSCAL,NMOVE,NSHFT
*,NND,NPRT,EBATC
LCRGICAL NSCAL,NMOVE,NSHFT,NPRT,EBATC
INTEGER*2 IPCINT(10CCC), ITABLE(I0CC(N), PTOC, PTPTR, TAPMAX, TAPBPR
INTEGER*4 IDIREC, OPEN, D1RMAX, DI1RTF, PTCTR, PI, TPTCTR
REAL*8 D1REIC(4CC), OB1JNAM, GEMNAM, PAC
EQUIVALENCI11(IPCINT(1), PCINT(1)), (ITABLE(1), TABLE(1)),
* (D1REIC(1), IDIREC(1))
C
ISW IS SET TO ZERO FOR ENTRY FROM NAMCBJ.
ISW=0
GO TO 14
ENTRY NAMCBR(0B1JNAM)
C
ISW IS SET TO CNE FOR ENTRY FROM NAMCBR
ISW=1
GO TO 14
ENTRY NAMECBR(0B1JNAM)
C
ISW IS SET TO MINUS CNE FOR ENTRY FROM NAMCBR
ISW=-1
C
CHECK TC SEE IF THIS CALL WAS LEGALLY PLACED (OPEN=O)
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 TC SEE IF NAME OK. IF NOT, RETURN 1
CALL FINDNAM(IFINC, PADD(0B1JNAM), ITYP, 615)
CIREC(DIRPTR)=PAL(0B1JNAM)
PRINT 16, CIREC(DIRPTR)
16 FORMAT('** CRAWL ERROR - ***AB,*** ALREADY USED AS DRAWL NAME')
CALL ABCRTZ(2)
RETURN 1
C
PUT NAME (PACCEC) INTO NAME TABLE
15 CIREC(DIRPTR)=PA1(0B1JNAM)
C
SET OBJECT TYPE IN NAME TABLE
ICIREC(DIRPTR+2+1)=1
C
POINTER TO LINKAGE TABLE
ICIREC(DIRPTR+2+2)=TABPTR+1
C
OBJECT TYPE IN LINKAGE TABLE
ITABLE(TABPTR+1)=1
C
BACK POINTER TO NAME TABLE
ITABLE(TABPTR)=CIREC
C
PRINT ENTRY MESSAGE
IF(BATCH) CALL LCCOUNT(Z)
IF(NPRT) PRINT 1CC, CIREC(DIRPTR)
100 FORMAT('** ENTRY TO NAMCBJ(''***,AB,2H'))
C
UPDATE NAME TABLE INDEX
DIRPTR=CIREC+2
C
IF OVERFLOW IS HERE, GET CLT VIA USERS SPECIFICATION.
IF(DIRPTR.GT.DIRMAX) CALL ABCRTZ(-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 NAMCBR
30 CALL POINTS(C, C..0.., 0..1..)
PTCTR=1
GO TO 40
C
FOR CALL NAMCBR
50 CALL INCFPT(0, C..0.., 0..1..)
C
COUNT THIS FIRST POINT IN THE POINT TOTAL
PTCTR=1
40 RETURN
SUBROUTINE PCINTS(IS,X,Y,Z,*
COMM /ATAEL*, ICIREC(800), PCINT(500), TABLE(5000),
CMMCN /VAR*, FTMAX, PTPTR, TABMAX, TAPTR, DIRMAX, DIRPTR,
*PTCTR, PTPTR, IPTR, PTCTR, COPEN, NASM, GENNAM, NSCALE, NMOVE, NSHFT
*, LNM, NPRT, BATCH
LOGICAL NSCAL, NMCE, NSHFT, NPRT, BATCH
INTEGER*2 IPCINT(1000), ITAELE(1000), PTMAX, PTPTR, TABMAX, TAPTR
INTEGER*4 ICIREC, OPEN, DIRMAX, DIRPTR, PTCTR, PT, PTCTR
REAL*8 CIREC(400), OBNJNAM, GENNAM, PAC
EQUIVALENCE (IPCINT(1), PCINT(1)), (ITABLE(1), TABLE(1)),
* (DIREC(1), ICIREC(1))

IF we are not in object definition mode (i.e., OPEN/=1)
we do not belong here.
IF(OPEN.EQ.1) GC TO 635
IF(BATCH) CALL LCOUNT(2)
PRINT 634

634 FORMAT(*C***CRAWL ERRCR - PCINTS CALLED WITH NO NAMED *,
 1 'OBJECT VIA NAMBJ.')
CALL ABCTZ(1)
RETURN

INCREMENT PCINT COUNTER

PTCTR = PTCTR + 1

PTCTR IS A "NAME" FOR THIS POINT
IPCINT(PTPTR*2-1) = DIRPTR
IPCINT(PTPTR*2) = PTCTR

PUT THE X VALUE INTO PCINTS. NOTE THAT ITS TYPE
IS CHECKED. IT MUST BE A FLOATING PCINT NUMBER.
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

MAKE SURE THERE IS NOT A ZERO SCALE FACTOR (W)
W1 = CHECKF(W)
IF(ABS(W1-C.C) .LT. 0.0001) GC TO 616
PCINT(PT) = W1
GC TO 617

616 IF(BATCH) CALL LCOUNT(1)
IF(NPRT) PRINT 55C

550 FORMAT(*C***WARNING. SCALE FACTOR CF ZERO. CHANGED TO ONE*)
CALL ABCTZ(2)

IF W WAS ZERO, ASSUME HE MEANT LAE, AND PUT IT IN
POINT(PT) = 1.

UPDATE THE TOTAL PCINT COUNTER

PTCTR = PTCTR + 1
ITP = ITP + 2
ITABLE(ITP) = PTPTR

PUT THE PENPL/PENGCHA SWITCH IN THE LINKAGE TABLE, MAKING
SURE THAT IT IS AN INTEGER (VIA ICFKF)
ITABLE(ITP+1) = ICFKF(IS)

UPDATE PCINTS TABLE INDEX
PTPTR = PT + 1
MAKE SURE WE ARE NOT READY TO OVERFLOW THE POINTS TABLE
IF(PTPTPI .GT. PTMAX) CALL ABCH'72(-1)
RETURN
END

INCRPT

SUBROUTINE INCRPT(IS, X, Y, Z, h)
COMMON / ATABL / ICIREC(800), PCINT(5000), TABLE(5000)
COMMON / VAR / PTMAX, PTPTPI, TDIRMAX, TABMAX, DDIRPT, TABPT,
* PTCTPI, PT, TPTPI, TPTCTPI, OPEN, ANAM, SHAM, GEMAN, NSCAL, NMOVE, NSHFT
* LLM, NPRT, BACTH
LOGICAL NSCAL, NARC, NSHFT, NPRT, BACTH
INTEGER*4 PCINT(10000), ITABLE(10000), PTMAX, PTPTPI, TDIRMAX, TABMAX
INTEGER*4 IDIRREC, IDIREC, OPEN, DIRMAX, DDIRPT, PTCTPI, PT, TPTCTPI
REAL*8 IDIREC(400), OBJNAM, GEMAN
EQUIVALENCE (IPINT(1), PCINT(1)), (ITABLE(1), TABLE(1)),
* (DIRREC(1), ICIREC(1))

IF THERE WERE NO PREVIOUSLY DEFINED POINTS, WE CAN'T BE RELATIVE
TO THE PREVIOUS VALUE
IF(TPTCTPI .NE. 1) GC TO 10
IF(BACTH) CALL LCCOUNT(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 ABCH'72(16)

SO WE ASSUME FE MEANT TO BE RELATIVE TO (0.,0.,0.)
CALL POINTS(IS, C, 0., 0., 1.)
RETURN

10 IRROW = TPTCTPI*5-7
AND ADD THEM IN
X1 = TABLE(IRRCW)*X/W
Y1 = TABLE(IRRCW+1)+Y/W
Z1 = TABLE(IRRCW+2)*Z/W

LET POINTS PUT THEM INTO THE POINT TABLE
CALL POINTS(IS, X1, Y1, Z1, 1.)
RETURN
END

SPECIAL ENTRIES FOR POINTS

SLSUBROUTINE VISIA(X, Y, Z)
THESE ARE SELF EXPLANATORY
CALL PCINTS(1, X, Y, Z, 1.)
GC TO 1
ENTRY INVA(X, Y, Z)
CALL PCINTS(0, X, Y, Z, 1.)
GC TO 1
ENTRY VISR(X, Y, Z)
CALL INCPT(1, X, Y, Z, 1.)
GC TO 1
ENTRY INVR(X, Y, Z)
CALL INCPT(0, X, Y, Z, 1.)
GC TO 1
ENTRY VISIA(X, Y)
CALL POINTS(1, X, Y, 0., 1.)
GC TO 1
ENTRY INVA2(X, Y)
CALL PCINTS(C,X,Y,0..1.)
GC TO 1
ENTRY VISR2(X,Y)
CALL INCPT1(X,Y,0..1.)
GC TO 1
ENTRY INVZ2(X,Y)
CALL INCPT(C,X,Y,0..1.)
GC TO 1
RETURN
END

C NAMTRA

SUBROUTINE NAMTRA(GBJNAM,A1,..,A16,*)
* A10,A11,A12,A13,A14,A15,A16,*)
CLMMCN /ATABL/ IIREC(8CO), FCINT(5CC),TAEMAX,TAEPR,CIRMAX,DIRPR,
*PTCTR,PT,ITP,TPCT,CPEN,NASH,GENNAM,ASCAL,NMOVE,NSHFT
*LM,NPRT,BATCH
LOGICAL NSCAL,NMOVE,NSHFT,NFR,T,BATCH
INTEGER*2 IPCINT(10000),ITABLE(10000),PTMAX,PTPTR,TABMAX,TABPR
INTEGER*4 DIREC(OPEN,CIRMAX,DIRPR,PTCTR,PT,TPCT,DIR(2),
*TEST/66CCCCCOO/
REAL*8 DIREC(4C0),GBJNAM,GENNAM,PAC,GBJ Names EQUA (IPCI NT(1),PCINT(1)),(ITAB LE(1),TABLE(1)),
*(DIREC(1),DIREC(1)),(C(1),GBJNAM)
C ARRAY E WILL BE LSEC BY NAMTRA
C DIMENSION E(4,4)
C IFLG = 0 IF ENTRY WAS VIA NAMTRA
IFLG=0
GC TO 335
ENTRY NAMTRA(GBJNAM,E,*)
C IFLG = 1 IF ENTRY WAS VIA NAMTRA
IFLG=1
C DO WE REALLY BELONG HERE?
C 335 IF(OPEN.NE.0)CALL CLOSE
OPEN=0
C PAD OUT THE NAME PASSED AS ARGUMENT
OBJNAM=PAC(GBJNAM)
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)GC TO 115
C FNDDM 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 FNDDM(IFIND,OBJNAM,ITYP,6115)
IF(BATCH)CALL LCCUNT(2)
PRINT 101,OBJNAM
C IF IT WAS PREVIOUSLY USED AS AN ENTITY NAME, THAT'S A FATAL
C ERROR.
101 FORMAT(*'****CRAWL ERROR - ',A8,' ALREADY USED AS DRAWL NAME'*)
CALL ABERTZ(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)=OBJNAM
C OVERFLOW THE NAME TABLE?
IF(DIRPTR.GT.DIRMAX)GC TO 776
PUT IN TYPE 2 FCR TRANSFORMATION TYPE
IDIREC(DIRPTR*2+1) = 2
POINTER INTO LINKAGE TABLE
IDIREC(CIRPTR*2+2) = TABPTR+1
BACK POINTER
ITABLE(TABPTR) = CIRPTR
CHECK ANC SEE WHETHER THE FIRST CHARACTER IN THE NAME IS
A PERCENT SIGN (%). IF SO, IT WAS PROBABLY A CALL BY CMBMAT.
IF(LAND(C(1),TEST).EQ.TEST) GO TO S38
PRINT ACMPAL ENTRY MESSAGE
IF(BATCH)CALL LCCUNT(2)
IF(NPRT)PRINT 103,OBJNM1
103 FORMAT('ENTRY TC NAMTRA('*,A8,2X')')
GO TO 939
S38 IF(BATCH)CALL LCCUNT(2)
INTERNAL CALL MESSAGE
IF(NPRT)PRINT 94C,OBJNM1
940 FORMAT('INTERNAL CALL TC NAMTRA('*,A8,2X')')
UPDATE DIREC INDEX
939 DIRPTR = CIRPTR + 2
CHECK FCR NAME AND LINKAGE OVERFLOW
IF(DIRPTR.GT.CIRMAX)GO TO 776
IF(TABPTR+34.GT.TABMAX)CALL ABCRTZ(-3)
ITABLE(TABPTR+1) = 2
L = (TABPTR+3)/2
IF(FLG.EQ.1)GO TO 106
THIS SECTION HANDLES 'NORMAL' ENTRIES
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)
NMTRAI Handling Section. Just LUCP through array E

DC 772 J=1,4
DC 772 I=1,4
TABLE(L)=CHECKF(E(J, I))

L=L+1
L=L-1
A17=CHECKF(E(4, 4))

CHECK the 16th element to make sure it is not zero (or
small enough to be the next best thing to zero.)

IF (ABS(A17-C.*).LT.0001) GC TO 774

IF NOT ZERO, STORE IT.
TABLE(L)=A17
GC TO 775

IF (BATCH) CALL LCCINT(1)
IF (NPRT) PRINT 55C

FORMAT('***Warning - Scale Factor CF ZER. Changed to One.'*)

IF IT WAS ZERO, ONE IS A PRETTY GOOD GUESS AS TO WHAT HE MEANT.
A17=1.
CALL ABCRTZ(4)
TABLE(L)=A17

UPDATE LINKAGE INDEX
TABPTR=TABPTR+34
RESET IFLGF FCR NEXT TIME
IFLG=0
RETURN

CALL ABCRTZ(-2)
RETURN
END

NAMEASM

SUBROUTINE NAMEASM(OBJNAM, *)
COMMON /ATAEL/, ICIREC(800), PCINT(5000), TABLE(5000)
COMMON /VAK/, PMAX, PTPTR, TABMAX, TAEFTR, DIRMAX, DIRPTR,
*PTCTR, PT, ITP, TPTCTR, CPEN, KASW, GEMAN, NSCAL, NMOVE, NSHFT
*, LAM, NPRT, BATCH
LGCICAL NSCAL, AMCAE, NSHFT, NFR,T, BATCH
INTEGER*2 IPCINT(10000), ITABLE(10000), PTMAX, PTPTR, TABMAX, TABPTR
INTEGER*4 ICIREC, OPEN, CIRMAX, DIRPTR, PTCTR, PT, TPTCTR
REAL*8 CIREC(4CO), OBJNAM, GEMAN, PAC, OBJ
EQUIVALENCE (IPOINT(I),POINT(I)), (ITABLE(I), TABLE(I))

* (CIREC(1) , ICIREC(I))

NAMESUB was used in the earliest versions of DRAW'L, when the only
levels were points, subassemblies, and subassemblies. Assemblie
could contain only subassemblies and points, and subassemblies
could contain only points. We've come a long way........
This was left in for compatibility.

FIRST QUESTION, AS USUAL, IS 'CLCES HE BELONG HERE?'

IF (OPEN.*.E.O.) CALL CLOSE
IF SC, OPEN = 2 ===> MODE IS 'ASSEMBLY DEFINITION'
CPEN=3

SEE IF NAME IS ALREADY DEFINED. IF SC, RETURN 1
CALL FNACLAM(IFINE, PAC(OBJNAM), ITP, &2CO)
DIREC(DIRPTR)=PAC(CBJNAM)
PRINT 201, DIREC(CIRPTR)

FORMAT('O**CRAW'L ERRCR - ***A6, *** ALREADY USED AS DRAW'L NAME')
CALL ABCRTZ(3)
RETURN 1
STORE NAME, TYPE, AND PCINTER TO TABLE IN DIRECT

200 CIREC(DIRPTR)=PAI(UBJNAM)
   [CIREC(DIRPTR*2+1)=3
    IDIREC(DIRPTR*2+2)= TABPTR+1
     BACK PCINTER
     ITABLE(TABPTR)=CIRPTR
     IF(BATCH)CALL LCCUNT(2)
     IF(NPRT)PRINT 301,DIREC(DIRPTR)
     ENTRY MESSAGE

301 FORMAT('CENTRY TC NAMASM(***,A8,2H*)')
     UPDATE NAME INDEX AND CHECK FOR OVERFLOW
     DIREC(DIRPTR)=DIRPTR+2
     IF(DIRPTR.GT.DIRMAX)CALL ABCRTZ(-2)
     IF(TABPTR+3.GT.TABMAX)CALL ABCRTZ(-3)
     STORE TYPE IN LINKAGE TABLE AND ZERCE TWO UNUSED HALF-WORDS
     ITABLE(TABPTR+1)=3
     ITABLE(TABPTR+2)=0
     ITABLE(TABPTR+3)=0
     INITIALIZE THE 'ENTITIES IN THIS ASSEMBLY' COUNTER
     NASM =0
     RETURN

C
C NAME
C
ENTRY NAME(UBJNAM,*,*,*)
     MAKE SURE HE IS IN THE RIGHT PLACE
     IF(OPN*EC.3)GO TO 105
     IF(BATCH)CALL LCCUNT(2)
     PRINT 104

104 FORMAT('C***CRAWL ERROR - NAME CALLED WITH NO *',
     1 'NAMED ASSEMBLY VIA NAMASM')
     CALL ABCRTZ(5)
     RETURN 2

105 OBJ=PAI(UBJNAM)
C
     FIND THE NAME IN THE NAME TABLE
     CALL FNCNAME(I,OE,ITYP,6100)
     IT HAS BETTER NOT BE TRANSFORMATION TYPE (2)
     IF(ITYP.NE.2)GO TO 107
     IF(BATCH)CALL LCCUNT(2)
     PRINT 108,CBJ

108 FORMAT('C***CRAWL ERROR - TRANSFORMATION ***,A8,*** USED',
     1 'AS ARGUMENT TO NAME.')
     CALL ABCRTZ(6)
     RETURN 3

107 NASM=NASM+1
     [INDX=TABPTR+5+NASM
     HAVING INCREMENTED THE ENTITY COUNTER, AND CALCULATED
     AN INDEX INTO THE LINKAGE TABLE, MAKE SURE WE'RE NOT ABOUT TO
     OVERFLOW SAIC IMPORTANT TALE.
     IF(INDX.GT.TABMAX)CALL ABCRTZ(-3)
     I WAS RETURNED BY FNCNAME
     ITABLE(INDX)=I
     RETURN
C
     IF THE NAME WAS NOT FCUNC, COME CCW HERE
     IF(BATCH)CALL LCCUNT(2)
     PRINT 11C,CBJ,OBJ

110 FORMAT('C***CRAWL ERROR ***,A8,2F'(,Z16,' NOT FOUND BY NAME.')
     CALL ABCRTZ(7)
     RETURN 1
END

C FNCAAM

SUBROUTINE FNCAAM(IFIND,NAM,ITYP,*)
COMM CN /ATABL/ ICIREC(800), FCINT(5000), TABLE(5000)
COMM CN /VAR/ PTOX, PTOPT, TAEXPT, TAEPT, DIREX, DIRPT,
* PTOCT, PTO, PTOPT, TOCT, OPN, NAM, GENNAM, NSCAL, NMOVE, NSHTF
*,LNM,NPRT,EBATC
LOGICAL NSCAL,NCEV,NSHTF,NPRT,EBATC
INTEGER I IPCINT(10000), ITABLE(10000), PTOX, PTOPT, TABMAX, TABPRT
INTEGER I DIREC, OPN, DIREX, DIRPT, PCINT, PTOCT, PTO, TOCTCT
REAL 8 DIREC(4CC), OBJNAM, GENNAM, FAL, NAM
EQUIVALENCE (IPCINT(1), PCINT(1)), (ITABLE(1), TABLE(1)),
* (DIREC(1), DIREC(1))
C ESTABLISH THE UPPER LIMIT O F DIREC. WE WILL COUNT BACKWARD THRO UGH THE NAME TABLE.
NUP=DIRPTR-2
DO 100 J=2,NUP,2
I=DIRPTR-J
C IF THIS IS THE NAME, EXIT
IF(NAM.NE.EIREC(1))GTO TC1CC
C SEND BACK THE TYPE
C ( W HERE 1===>OBJECT, 2===>TRANSFORMED, 3===>ASSEMBLY, 4===>TRANSFORMED
C OBJECT OR ASSEMBLY)
ITYP=I DIREC(1)*2+1
C ALSO SEND BACK THE LOCATION WHERE IT CAN BE FOUND
IFIND=I
RETURN
100CONTINUE
RETURN 1
END

C FINCBJ

SUBROUTINE FINCBJ
COMM CN /ATABL/ ICIREC(800), FCINT(5000), TABLE(5000)
COMM CN /VAR/ PTOX, PTOPT, TAEXPT, TAEPT, DIREX, DIRPT,
* PTOCT, PTO, PTOPT, TOCT, OPN, NAM, GENNAM, NSCAL, NMOVE, NSHTF
*,LNM,NPRT,EBATC
LOGICAL NSCAL,NCEV,NSHTF,NPRT,EBATC
INTEGER I IPCINT(10000), ITABLE(10000), PTOX, PTOPT, TABMAX, TABPRT
INTEGER I DIREC, OPN, DIREX, DIRPT, PCINT, PTOCT, PTO, TOCTCT
C ERROR IS USED IN THE PRINT STATEMENT BELOW
REAL 8 DIREC(4CC), OBJNAM, GENNAM, ECRF(3), *FINCBJ *, *ENDCMB *
1 'FINASY' /, XNAME
C IF ANY OF THE VARIABLES IN /CLSE/ ARE USED BY ENDCMB
REAL TF(16)
COMM CN /CLSE/ TC(4,4), XNAME, ISH
EQUIVALENCE (IPCINT(1), PCINT(1)), (ITABLE(1), TABLE(1)),
* (DIREC(1), DIREC(1))
C DO WE EWLCAGE HERE?
C TO HAVE ACCESSED THIS ROUTINE, WE SHOULD HAVE 'OBJECT
C DEFINITION STATE' (CPEN=1) SET ON
IF(OPN.EQ.1)GTO 110
IF(NPRT)CALL DDCLINT(1)
C ERROR(CPEN) IS WHAT SHOULD HAVE BEEN CALLED
IF(NPRT)PRINT 111, ECRF(1), ECRF(CPEN)
C NOW GO AND SEE WHERE HE SHOULD HAVE BEEN
IF (OPEN.EC.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 =IIP+2
IF (BATCH) CALL LCCLNT(1)
IF (NPRT) PRINT 101, DEREIC (IPRTR-2), PTCTR
C
OUTPUT THE NUMBER OF POINTS IN THE OBJECT (A NICE CHECK
C
CN DO LCOPS, LOCLING READS, ETC.)
101 FORMAT(3F* ' ,A8,*' HAS ',I4,*' POINTS.*')
C
RESET COPEN ("I'M READY FOR ANYTHING STATE")
C
OPEN=0
RETURN
C
C
FINASM
C
ENTRY FINASM
C
IF WE KEEP NAMSUB, WE JUST Gotta HAVE A FINSUB
150 IF (OPEN.EC.3) GC TO 160
C
CNCE AGAIN, IF HE DOESN'T BELONG HERE, TELL HIM SO
C
IF (BATCH) CALL LCCLNT(1)
IF (NPRT) PRINT 111, ERRCR(3), ERRCR(COPEN)
IF (OPEN.EC.1) GC TO 110
C
GL TO 210
C
RESET COPEN
160 OPEN=0
C
OUTPUT THE NUMBER OF ENTITIES INCLUDED
C
IF (NPRT) PRINT 170, NASM
170 FORMAT(* ' THERE ARE ',I4,*' COMPONENT ENTITIES.*')
C
STORE AWAY THE NUMBER
C
ITABLE(TABPTR+4)=NASM
C
EVEN ALIGNMENT IS REQUIRED IN LINKAGE TABLE
IF ((NASM/2*2).NE. NASM) TABPTR=TAEBTR+1
C
UPDATE LINKAGE INDEX
C
TABPTR = TABPTR + NASM + 6
C
RESET NAM FOR NEXT TIME
NASM = 0
RETURN
C
C
ENCMB
C
ENTRY ENCMB
C
HE TRULY BELONGS HERE IFF COPEN=2 (CMBMAT WAS CALLED)
205 IF (OPEN.EC.2) GC TO 210
C
IF (BATCH) CALL LCCLNT(1)
IF (NPRT) PRINT 111, ERRCR(2), ERRCR(COPEN)
IF (OPEN.EC.1) GC TO 110
C
GL TO 160
C
111 FORMAT(* ' ***WARNING*** ILLEGAL CALL TC ',A7,1H7;:,A7,
1 'CALLED FOR YCL.*')
C
CALL ABLRFZ(15)
210 CBJNAM=XNAME
C
C
RESET COPEN
C
COPEN=0
C
C
CMBMAT SET A SERIES OF (AT LEAST ONE) MATRIX MULTIPLIES
C
INTO ACTION, THE MATRIX IN TC IS ALL READY FOR TRANSFORMATION
C
DEFINITION. PASS THE NAME GIVEN TO CMBMAT AND THE MULTIPLIED
C
ARRAY IN TC TO NMTR1 (THE ALTERNATE ENTRY TO NMTRA).
CALL NMTRAI(CBJNAM, TC)
RESET ISH (CONTAINED IN /CLSE/)
ISW=0
RETURN
END

TRANSF

SUBROUTINE TRANSF(NEW, CLD, XFCRM, *, *, *, *)
IT SHOULD BE POINTED OUT THAT TRANSF WILL NOT PERFORM
THE MATRIX MULTIPLICATIONS REQUIRED FOR A TRANSFORMATION.
IT MERELY SETS UP THE LINKAGES, AND THE MULTIPLICATIONS
ARE PERFORMED WHEN CRAM, SHCM, PRBPLT, PROUT, ETC., ARE CALLED
CCMON /ATAEL/ ICIREC(1800), PCIINT(5000), TABLE(5000)
CCMON /VAR/ FMAX, PTPIR, TABMAX, TABPTR, DIMAX, DIFTR, PTCTR, PT, IPTCTR, Q(2)

INTEGER*2 IPCINT(10000), ITABLE(12000), FMAX, PTPTR, TABMAX, TABPTR
INTEGER*4 ICIREC, OPEN, DIMAX, DIFTR, PTCTR, PT, IPTCTR, Q(2)
LOGICAL NSCAL, NMCVE, NSHT, NPRT, BATCH
REAL*8 LIREC(400), OBJNAM, GENNAM, PAC, NEW, ULC, XFORM
* * * * *

LAM, NPRT, BATCH
REAL*8 IPCINT(10000), ITABLE(12000), FMAX, PTPIR, TABMAX, TABPTR
INTEGER*4 ICIREC, OPEN, DIMAX, DIFTR, PTCTR, PT, IPTCTR, Q(2)
LOGICAL NSCAL, NMCVE, NSHT, NPRT, BATCH
REAL*8 LIREC(400), OBJNAM, GENNAM, PAC, NEW, ULC, XFORM
ERROR MESSAGE DATA IS IN TYPE
REAL*8 TYPE(4)/* OBJECT ',' XFCRM ',' 'ASSEMBLY',' 'ASSEMBLY'*/
REAL*8 NEWL, XFCRM1, OLC1
EQUIVALENCE (IPCRINT(1), PCIINT(1)), (ITABLE(1), TABLE(1)),
* (DIREC(1), IDIREC(1)), (GENNAM, Q(1))

MASK WILL BE USED TO DETERMINE WHETHER THE CHARACTER
MANIPULATION SUPPORTING NAME-GENERATION WILL BECOME
ULTRA-COMPLEX, OR SIMPLY REMAIN MODERATELY CONFUSING....

DATA MASK/ZOOOOO/ZOOO/

AS USUAL, CAN HE BELIEVE HERE?
IF (OPEN.NE.0) CALL CLOSE
OPEN=0

PAD OUT THE THREE ARGUMENTS
NEWL=PA1C(NEW)
CLUD=PA1C(CLUD)
XFCRM1=PA1C(XFCRM)

CALL LCCINT(2)

IF (BPRT) PRINT 663, NEWL, CLD, XFCRM1
PRINT THE ENTRY MESSAGE

663 FFORMAT('ENTRY TC TRANSF* ','(1H*A8,2H,),1H',A8,2H')
FIND THE TRANSFORMATION MATRIX. IF ITS NOT THERE, BRANCH TO 357
CALL FNCAM(J, XFCRM1, ITYP, 6357)
MAKE SURE THE NAME GIVEN WAS A TRANSFORMATION
BRANCH TO 665 IF IT WAS
IF (ITYP.EQ.2) GC TO 665
IF (BPRT) CALL LCCINT(2)
PRINT 664, XFCRM1
IF IT WASNT, TELL HIM ALL ABOUT IT

664 FFORMAT('****CRAWL ERRORR - ***A8,*** IS NOT A TRANSFORMATION ',
1 'NAME')
CALL ABCRTZ(17)
RETURN

FIND THE 'SOURCE' NAME IN DIRECT. IF IT WASNT THERE,
BRANCH TO 356

CALL FNCAM(1, CLD, ITYP, 6356)
IF IT WAS A TRANSFORMATION, GC TO 634
IF (ITYP.EQ.2) GC TO 834
IF THIS CALL TC TRANSF WAS EQUIVALENT TO THE FORTRAN STATEMENT
A = A + P

(I.E., THE FIRST ARGUMENT WAS THE SAME AS THE SECOND), KEEP GOING

OTHERWISE ERANCE TO 800

IF (OLD1.NE.NEW1) GO TO 800

Q IS INVOLVING THE FORTRAN IV CHARACTER MANIP-

ULATION RESTRICTIONS. THE FUNCTION 'LAND' DOES ITS SHARE

TWO. LAND IS A BITWISE LOGICAL FUNCTION. A BIT IS ON IN

LAND'S RETURNED VALUE ONLY IF IT WAS ON IN THE SAME POSITION

IN BOTH OF THE ARGUMENTS. THIS SEQUENCE MERELY CHECKS TO

SEE IF OVERFLOW IN THE EBCCIC REPRESENTATION OF THE GENER-

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

IN THIS IMPLEMENTATION, THE VALUES ASSOCIATED WITH

THE NAME GIVEN AS FIRST AND SECOND ARGUMENTS ARE PRESERVED

AND GIVEN A NEW (GENERATED) NAME. THE NEW NAME IS PRINTED

HERE. NOTE THAT THIS ALLOWS THE OLD VALUES TO STILL BE

REFERENCED INTERNALLY, BUT REFERENCES TO THE OLD NAME NOW POINT

TO THE TRANSFORMED VALUE

I WAS RETURNED BY FNDNAM. PUT THE GENERATED NAME IN THE

TABLE IN PLACE OF THE OLD NAME.

100 DIREC(1) = GENNAM

PUT THE NAME JUST REPLACED INTO THE NEXT SLOT IN DIREC

DIREC(DIRPTR) = NEW1

PUT IN THE 'XFCRMED-ENTITY' KEY (=4)

DIREC(DIRPTR*2+1) = 4

INDEX INTO LINKAGE TABLE

DIREC(DIRPTR+2) = TABPTR+1

BACK POINTER

ITABLE(TABPTR) = DIREC

TYPE 4 AGAIN

ITABLE(TABPTR+1) = 4

PUT LOCATION OF THE XFCRM INTO LINKAGE TABLE

(ASS PROVIDED BY FNDNAM ABOVE)

ITABLE(TABPTR+2) = J

PUT IN LOCATION OF SOURCE ENTITY

ITABLE(TABPTR+3) = 1

UPDATE INDICES FOR NAME AND LINKAGE TABLES

DIRPTR = DIRPTR+2

TABPTR = TABPTR+6

200 RETURN

THIS SECTION IS FOR 'NORMAL' TRANSFORMATIONS

(I.E., THE FIRST TWO ARGUMENTS ARE NOT THE SAME.)

CHECK AND SEE WHETHER THE NAME HAS BEEN USED BEFORE

IF IT HASN'T, ERANCE TO 801

800 CALL FNDNAM(K, NEW1, ITYP1, &8C1)

IF (BATCH) CALL DCCOUNT(2)

PRINT 803, NEW1, TYPE(1, ITYP1)

TELL HIM THAT HE'S USED THE NAME BEFORE.

ALSO TELL HIM WHAT MODE WAS ASSOCIATED WITH THE NAME.

803 FORMAT ('***CRMWR ERR0R = ***', AB, '*** ALREADY USED AS ',

1, A8, ' NAME,

CALL ABFRTZ1(18)

RETURN 4

PUT AWAY THE NEW NAME (THIS IS AN ENTITY DEFINITION)

8C1 DIREC(DIRPTR) = NEW1

BACK POINTER IN LINKAGE TABLE

ITABLE(TABPTR) = DIREC
SUBROUTINE CRAW(CBJNAM,**,**,**,**)
  THIS SECTION OF CODE IS VIRTUALLY IDENTICAL FOR
  DRAw, SHOW, PRFILT, DISPLA, FRTCUT, PRTPLT. THIS ENTRIES ARE
  PROVIDED IN THIS AREA FOR ALL SUCH ROUTINES.
  IDEV IS SET TO INDICATE WHICH ENTRY WAS TAKEN:
  1==>&SHCW
  C==>&CRAW
  -1==>&CISPLA
  2==>&PRFILT
  3==>&PRFILT
  4==>&PRFILT
  CMMCN /ATAEL/ ICIREC(8CC), PCINT(5CC), TABLE(50CO)
  CMMCN /DRWCCM/CVEC(50), CURMAT(5C), EXPIRE(50), ID, IC,
  NPTS, ICMAX, ICMAX, XMN, XMN, XMN, XMN, XMN, XMN, XMN, XMN, XMN, XMN, XMN,
  (5460), IT(2730), IL
  CMMCN /VAR/ PMAX, PTPR, TMAX, TABPTR, DIRPTR, DIRPTR,
  *PTCTR, PT, IPT, TFCUT, LPEL, NMSK, GENAM, NICAL, NTC, NSHT,
  *, LNM, NPT, BATC
  INTEGER BLK/*, */AC/*NC */IPICTR
  INTEGER*2 INUX, DVEC, CURMAT, EXPIRE, ID, IC, IT
  LOGICAL NCAVL, NVE, NSHT, NPT, BATC
  INTEGER*2 PCTINT(10CCO), ITABLE(10CCO), PMAX, PTPR, TMAX, TABPTR
  INTEGER*4 ICIREC, OPEN, DIRMAX, DIFPTR, PICTR, PT
  REAL*8 LIREC(400), OBJNAM, GENAM, CRKPTR(C)/DISPLA '/', DRAw '/
  + '/SHOW '/, 'PRFILT '/, 'PRTPLT '/, 'PRTPLT '/, PAD, FOUND, XFOUND,
  + CBJNAM
EQUIVLANCE (ICINT(1),PQINT(1)),(ITABLE(1),TABLE(1)),
* (DIREC(1),ICIREC(1))
IDEV=0
GC TO 38
ENTRY SHCN(CBJNA)
IDEV=1
C DFN.8 BLANKS THE DISPLAY SCREEN
IF(NMOVE)CALL CFSD
GO TO 38
ENTRY DISPLAY(CBJNA)
ICEV=-1
GC TO 38
ENTRY PRTCLT(CBJNA)
IDEV=2
GC TO 38
ENTRY PRTPLT(CBJNA)
ICEV=3
GO TO 38
ENTRY PRTPLT(CBJNA,IPUT)
ICEV=4
GO TO 38
C DOES HE BELONG HERE?
38 IF(OPNE .NE. C)CALL CLCSE
OPEN=0
C INITIALIZE THE INDEX FOR THE CLTFLT ARRAY (IL=1)
IL=1
C CALL TIME(C) INITIALIZES THE CPL-TIME CCOUNTER (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=NC
IF(NMOVE)NMO=BLNK
IF(NSHFT)NSH=NC
C THE NEXT FOUR PARAMETERS ARE INITIALIZED FOR DRETRV
ID=1
IG=0
IDM=0
ICM=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 CLTFLT VECTOR. IT WILL BE
C EQUIVALENT TO CALL POINTS(0,0,.C,.C,.1.)
NPTS=1
IF(LNM.EQ.0)GO TO 8
NPTS=0
IL=-1
8 LNM=LNM+1
CBJMAF=FA(F(CBJNA))
C FIND THE ENTITY IN THE NAME TABLE. IF NOT THERE, TO 10
CALL ENDRAM(IFINE,OBJNA1,ITYP,E10)
IF((BATCH)CALL LCCNT(2)
DRWRPT(ICEV+2) WILL SET UP THE APPROPRIATE MESSAGE
IF((NPRT)PRINT 9,DRWRPT(ICEV+2),CBJN M1
9 FORMAT(’CENTRY TO ’,A6,2H(’,A8,2H’))
IF((BATCH)CALL LCCNT(1)
IF((NPRT)PRINT 1,DRWRPT(ICEV+2),NSC,NMC,ASH
C SET UP AND PRINT OUT THE APPROPRIATE PARAMETERS
C MAKE SURE AN ENTITY NAME WAS PASSED. IF SC, GO TO 11
C IF (TYP ’,A6,2H’ ) GO TO 11
C OTHERWISE, PRINT ERRCR MESSAGE
IF((BATCH)CALL LCCNT(2)
PRINT 3,DRWRPT(ICEV+2),CBJN M1
3 FORMAT(’C**CRAWL ERRCR - ATTEMPT TO ’,A6,’ TRANSFORMATION ’’
1 ’,A8,1H’)
CALL ABORTZ(21)
RETURN 2
10 IF((BATCH)CALL LCCNT(2)
PRINT 12,UBJN M1,EBJN M1,DRWRPT(ICEV+2)
C IF THE ARGUMENT WASN’T FCLRD, SET UP MESSAGE AND PRINT IT
12 FORMAT(’C**CRAWL ERRCR - ’,A8,2H(’,Z10,’ NOT FOUND BY ’,A6)
CALL ABORTZ(22)
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 SIMPLER IN ITS OWN, EFFECTIVE WAY.
C
C INITIALIZE VVEC FOR DRETRV
11 CVEC(1)=5-IFIND
13 CCONTINUE
C CALL DRETRV. NOTE IT WILL RETURN TO THE STATEMENT BEFORE
C THE CALL IF IT TAKES EXIT CN AND KEEP CALLING ITSELF
CALL DRETRV(613,619)
C EXIT TC 19 IS TAKEN IF THE USER ATTEMPTED TO PLACE THE
C EYE POINT IN THE SAME X-Y FLANE AS ONE OF THE POINTS TO
C BE DRAWN (THIS WOULD CAUSE DIVISION BY ZERO)
C DRETRV HAS NOW SET UP THE CLTPLT ARRAY. CALL OUTPUT, WHICH
C LIKE CRAWL, IS COMMON TO ALL TYPES OF OUTPUT.
18 CALL OUTPUT(NSCAL,NSHFT,NCVE,IDEV,APRT,INPUT,BATCH,616)
C EXIT TO 16 IS TAKEN IF THE DRAWING GOES OFF THE SCREEN
C WHEN Executing A ‘SCLW’, THIS SCHEDULE COME ABOUT ONLY
C WHEN THE USER IS RUNNING WITH SHFT OR SCAL OFF.
C HAVING CTITEN HERE, THE DRAWING IS COMPLETE.
C GET THE LAPSED C.P.U. TIME IN INT (SINCE CALL TIME(6))
IF((NPRT)CALL TIME(1,0,INT)
C CONVERT IT FROM MILLISECCNCs TO SECCNCs
ZTIME=INT/TICO.
C AND PRINT IT ON
IF((BATCH)CALL LCCNT(1)
IF((NPRT)PRINT 17,ZTIME
17 FORMAT(’F7.3,’ C. P. U. SECCNCs FOR CRAWL.’))
C THE DRAWING IS COMPLETED. ALL IS WELL.
RETURN
C 19 IS ACCESSED ONLY IF DRETRV FAC AN ERRCR AS INDICATED ABOVE
RETURN 3
C 16 IS ACCESSED ONLY IF CLTPLT FAC AN ERRCR AS INDICATED ABOVE
RETURN 4
C END
SUBROUTINE CLTPLT(NSCALE, ASHTF, NMOVE, ICDEV, NPRT, LDN, BATCH)
C
OUTPUT'S MAIN FUNCTION IS TO SCALE DRAWINGS (AND
C SHIFT THEM IF NECESSARY) SO THEY FIT CNTG THE OUTPUT
C DEVICE. IT ALSO SELECTS THE APPROPRIATE OUTPUT DEVICE
C (ACCORDING TO THE INFORMATION PASSED IN ICDEV)
C
INTEGER*2 INLY, CVEC, CLRMAT, EXPIRE, IL, IC, IT
COMMON /CRACCM/CVEC(50), CURMAT(50), EXPRI(50), IO, IC,
1 NAND, ICMAZ, ICMAX, XMAM, XMIN, XMAX, YMIN, YMAX, IT(55460), IT(2730), IL
LOGICAL NSCALE, ASHTF, NMOVE, NPRT, BATCH
INTEGER IMAGE(1570), NSCALE(5)/C, C, 2, G, 3/
C
THE FOLLOWING 4 VARIABLES ARE USED FOR SCALING
XMIN=1.E6
XMAX=-1.E6
YMIN=1.E6
YMAX=-1.E6
C
XLIM AND YLIM ARE THE MAXIMUM CLTPLT SIZES ALLOWED
C FOR THE VARIOUS DEVICES. THIS SECTION SETS THEM.
IF(IDEV.EQ.1.OR.IDEV.EQ.11)CC TO 10
YLIM=28.
XLIM=49.
IF(IDEV.EQ.0)CC TO 12
C
THE UNIVERSITY OF MICHIGAN CRAWL IMPLEMENTATION USES
C THE UNIVERSITY'S CALCMAC ROUTINE PACKAGE. ALL SUCH ROUTINES
C ARE NAMED STARTING WITH A LETTER P
C PFDNUM 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 $RLA MYPROGRAM,SSIC=CRAWL 8=PLCTFILE
C THIS WILL WRITE THE PLCT FILE TO LCN 8 (IF DRAW IS CALLED)
C 8 SHOULD NOT BE USED FOR ANY OTHER PURPOSE
CALL PFCNAM(8)
C
PLTXMX SETS THE MAXIMUM PAPER LENGTH
CALL PLTXMX(49,)
C
GO TO 12
10 IF(IDEV.EQ.-1)GO TO 11
C
THE UNIVERSITY OF MICHIGAN CRAWL IMPLEMENTATION USES THE
C DF ROUTINE (FOR DISPLAY FILE ROUTINE) PACKAGE OF THE
C CLNCOMP PROJECT FOR THE DIGITAL EQUIPMENT CORPORATION
C 33B. 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(C,0)
C
GO TO 12
11 XLIM=30.
YLIM=10.
C
TUPEN INITIALIZES THE ROUTINE FOR THE 10" CN-LINE
C PLCTTHER REFERENCED BY DISPLAY.
C IT HAS BEEN IMPLEMENTED AT THIS POINT
CALL TPCEN(.TRUE.,)
12 IF(BATCH)CALL LCOUNT(3)
IF(NPRT)PRINT 34,XMIN,XMAX,YMIN,YMAX
C
PRINT OUT THE REAL DRAWING LIMITS AS DISCOVERED BY ORETRV
34 FORMAT(* XMIN= ',F12.3,' XMAX= ',F12.3/
1 YMIN= ',F12.3,' YMAX= ',F12.3)
C
START THE SHIFTING AND SCALING PROCEDURE
C FIND CLT IF THE LIMITS AS CISCOVERED FALL WITHIN XLM AND YLM
C MAKE SURE THERE'S NOTHING IN QUADRANTS II, III, OR IV UNLESS
C NOSHFT IS ON.
IF(XMIN+.C.)XMIN=0.
IF(YMIN+.C.)YMIN=0.
XSCALE=(XMAX-XMIN)/XLM
YSCALE=(YMAX-YMIN)/YLM
IF(NOSCAL)YSCALE=1.
IF(NOSCAL)XSCALE=1.
IF(XSCALE.LT.XLM)XSCALE=1.
IF(YSCALE.LT.YLM)YSCALE=1.
IF(NOSHFT)XMIN=0.
IF(NOSHFT)YMIN=0.
C J WILL COUNT THE NUMBER OF POINTS USED (ONE DISC. FILE CAN HCLD ONLY 255 POINTS.)
J=0
C
576 CONTINUE
IF(IDEV.GT.1)GC TO 426
C LOOP THROUGH THE ARRAY WHICH WAS SET UP BY CRETRV
C AND SCALE AND SHIFT ALL POINTS (NOTE THE SCALE FACTOR IS ONE IF NOSCAL IS ON AND THE SHIFT FACTOR IS ZERO IF NOSHFT IS ON.
C THE NEW MAXS AND MINS ARE RECORRED.
DO 126 I=1,NPTS
XR1=(I/(1+2)-1-XMIN)/XSCALE
YR1=(I/(1+2)-YMIN)/YSCALE
IF(XR1.LT.XMIN)XMIN=XR1
IF(XR1.GT.XMAX)XMAX=XR1
IF(YR1.LT.YMIN)YMIN=YR1
IF(YR1.GT.YMAX)YMAX=YR1
C GO TO THE RIGHT OUTPUT DEVICE'S Routines
IF(IDEV).EQ.227,127,227
C THIS IS CALCCMP TERRITORY, THE SWITCH WILL BE IN "IT".
C ALLOWABLE SWITCHES ARE 0=INVISIBLE, 1=VISIBLE, 2=DASHED
C LINE, 3=CENTRE-LINES ARE TO BE DASHED.
C THE SWITCHES HAVE TO BE EMBOSSED UP BY CNE TO BE MADE
C COMPATIBLE WITH THE ROUTINES THEY FLED. C WAS USED FOR
C INVISIBLE BECAUSE OF ITS LOGICAL CONNECTIONS.
127 IS=IT(I)+1
C EACH TYPE OF LINE TO BE DASHED HAS IT'S OWN PLACE
GC TC (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)
GC TO 126
C PENON ALSO GETS (X,Y) IN INCHES
121 CALL PENON(XR1,YR1)
GC TO 126
C PUSHLN (CASHEC LINES) GETS (X,Y) AND 4 OTHER PARAMETERS
C (UNRELATED TO GENERAL IMPLEMENTATIONS)
122 CALL PUSHLN(XR1,YR1,2,1,0,C)
GC TO 126
C PCTRLN(FOR CENTER LINES) GETS (X,Y) IN INCHES. OTHER THREE
C PARAMETERS ARE NOT PRODUCED.
123 CALL PCTRLN(XR1,YR1,2,1,C)
GC TO 126
FOR THE CRT, IF THE SWITCH IS NON-ZEROC, SET IT TO 1

IS=0
IF(IT(I),.NE.C)IS=1

UPDATE NUMBER-CF-POINTS IN THIS DRAWING COUNTER
J=J+1

MAKE THE FIRST VECTOR AN INVISIBLE ONE
IF(J.EQ.1)IS=0

DFXYC BUILDS A DISPLAY FILE. THE ARGUMENTS ARE
X AND Y IN INCHES, THE CAN/CFF SWITCH, EXIT TO 999 IS TAKEN
C
IF THE CRAWING WILL CC OFF THE SCREEN (COULD CRASH
C
THE DISPLAY'S SOFTWARE OTHERWISE.)
CALL DFXYC(XR1,YR1,IS,6999)
IF(J.LE.254)GO TO 126

C IF THE CURRENT FILE IS NULL, SET UP ANOTHER ONE
J=0

DFAVL RETURNS THE NAME CF THE LAST USED
C DISPLAY FILE. THE ASSUMPTION THAT CRAWL IS NOT THE
C ONLY POSSIBLE SOURCE CF DISPLAY FILES ALLOWS GREATER
C FLEXIBILITY.
CALL DFAVL(NAME)

C DFX01 SENDS THE DISPLAY FILE TO THE REMOTE DISPLAY
CALL DFX01(C,NAM)

C DFINAG AGAIN INITIALIZES THE NEW C.F. BUFFER
CALL DFINAG(0,0)
GC TO 126
C
ON-LINE 10" CALCCMP ROUTINE
327 IS=0
IF(IT(I),.NE.C)IS=1

SEND THE PLGHTER THE X, AND Y IN INCHES, WITH PENUP/DOWN Sw.
CALL XYAES(X,Y,IS)

126 CONTINUE
C 126 IS THE END OF THE LLCMP.
C IF WE GET HERE, THE DRAWING IS CCNE. GET THE NAME OF
C THE LAST C.F. FOR TRANSMISSION.
CALL DFAVL(NAME)
GC TO 333
C
THIS SECTION IS FOR PRINTED CUTFIL (PRTPUT,PRTPLT,PRTOUT)

DC 424 1=1,NPTS 426
12=1*2
121=1*2-1
C SCALE THINGS IN THE ARRAY 'T' AND LEAVE THEM THERE
C PASS THE ARRAY TO THE APPROPRIATE ROUTINE
C FOR FINAL CUTFIL
T(I21)=(T(I21)-XMIN)/XSCALE
IF(T(I21).LT.XMIN)XMIN=T(I21)
IF(T(I21).GT.XMAX)XMAX=T(I21)
T(I2)=T(I2)-YMIN/YSCALE
IF(T(I2).LT.YMIN)YMIN=T(I2)
IF(T(I2).GT.YMAX)YMAX=T(I2)
424 CONTINUE
IF(DEV=2)425,525,625
C 525 IS FOR PRINTER PLCT ROUTINE
C 625 IS FOR WRITING ON A DEVICE (LCD C - S)
C 425 IS FOR PRINTING OUT THE VALUES IN TABULAR FORM
425 IF(BATCH)CALL LCCOUNT(57)
DC 427 1=1,NPTS 426
IF(BATCH)CALL LCCOUNT(1)
C PRINT OUT THE VALUES IN THE ARRT 'T'
427 PRINT 110,T(I1),IT(I2-1),T(I2)
FORMAT(3X,12,4(2X,F7.2))
IF(BATCH)CALL LCCOUNT(57)
GO TO 333

525 IF(BATCH)CALL LCCOUNT(57)
IF(BATCH)ILEN=1CC
IF(.NOT.EATC)ILEN=71
IF(XMAX1.LT.(1.5*YMAX1))XMAX1=1.5*YMAX1
IF(YMAX1.LT.(.666*XMAX1))YMAX1=XMAX1*.666

C PLOT14 WILL PLCT THE ENDPOINTS OF ALL LINES ON THE PRINTER
C IN ORDER, THE ARGUMENTS ARE:
C SCALING INFORMATION (O 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 LS), LABEL
C SWITCH (O MEANS NO LABEL), AND LCCATLN CF LABEL.
C THIS RCLTNE IS IN THE MTS SYSTEM LIBRARY. DIFFERENT
C INSTALLATIONS WILL HAVE DIFFERENT RCLTINES AND ARGUMENTS.
CALL PLOT14(NSCALE,2,93,2,ILEN,IMAGE,XMAX1,XMIN1,YMAX1,YMIN1,
* 'a',T(1),T(2),NPTS,8,C,N)
IF(BATCH)CALL LCCOUNT(57)

333 CONTINUE
C IF THE DRAWING WAS SCALED, TELL HIM ABOUT IT
IF(ABS(XSCALE-1.)*.LT.0.05)CC TC 37
IF(BATCH)CALL LCCOUNT(1)
IF(NPRT)PRINT 36,XSCALE

36 FORMAT(1 THIS DRAWING WAS SCALED BY ',F8.3)
37 IF(BATCH)CALL LCCOUNT(3)
38 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 CLSES THE ON-LINE PLOTTER'S BUFFER
IF(IDEV.EQ.0.AND.NMOVE)CALL PLTEND
IF(ICEV.EQ.1)CALL DF201(C,NAM)
IF(NMOVE.AND.ICEV.EQ.-1)CALL TCLOSE
RETURN

999 RETURN 1
C
C INTERNAL RETRIEVAL SUBROUTINE
C
C DRETRV IS A PSEUDO-RECURSIVE RCLTNE. IT 'PUSHES'
C AS MANY TIMES AS THERE ARE LEVELS OF NESTED ASSEMBLIES OR
C APPLICATIONS OF TRANSFORMATION MATRICES. IT MAINTAINS THREE
C PUSH-CMP 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 CNTC DVEC, ALL ITS CONSTITUENT
C ENTITIES ARE ALSO Pushed CNTC DVEC. THE ENTITIES
WHICH ARE CONTAINED IN THE LAST ENTITY (IF ANY) ARE THEN
PUSHED ON THE END OF THE STACK, WITH THE OLD END BEING
GIVEN THE NEGATIVE OF ITS VALUE. IC IS DVEC'S INDEX.
CURMAT - CONTAINS POINTERS TO THE TRANSFORMATION MATRICES
WHICH ARE TO BE APPLIED TO ALL POINTS PROCESSED. THIS KEEPS
TRACK OF THE LEVELS OF TRANSFORMATION. IC IS THE INDEX
FOR THIS STACK. TMAT CONTAINS THE PRODUCT OF ALL MATRICES
FROM CURMAT(1) ... CURMAT(IC) AT ANY GIVEN TIME.
EXPRIE - KEEPS THE VALUES OF IC SUCH THAT WHEN EXPRIE(IC)
IS LESS THAN ID, CURMAT(IC) IS TAKEN OFF THE LIST, AND
TMAT IS RE-EVALUATED. IC IS LSEC BY EXPRIE AS INDEX.
SUBROUTINE CRETRV($(*,*
C CMMCN /TAEL/ ICIRED(80C), FCINT(50C), TABLE(5000)
C CMMCN /CRwCM/CVEC(50C), CURMAT(50C), EXPRIE(50C), ID, IC,
C NPTS, IMAX, IMAX, XMIN, XMAX, YMIN, YMAX, T(5460), IT(2730), IL
C CMMCN /VAR/ PTMAX, PTFTR, TAEMAX, TAEFR, DIPMAX, DIPFTR,
C *PCTCR, PT, TIP, TIPCTR, OPEN, NAMX, GENVAM, KSCAL, NMOVE, NSFHT
C *LNM, NPTF, BATCH
C INTEGER*2 INLM, DVEC, CURMAT, EXPRIE, IC, IC, IT
C INTEGER*2 IPCINT(100C), ITABLE(100C), PTMAX, PTFTR, TABMAX, TABPTR
C INTEGER*4 IDIREC, OPEN, DIREC, DIPFTR, PCTCR, PT, TIPCTR, IX/1/, IY/2/
C LOGICAL BATCH
C REAL*8 CIRED(4C0), PAD, GENVAM
C EQUIVALENCE (IPCIINT(1), IPCINT(1)), (ITABLE(1), TABLE(1)),
C * (DIRC(1), ICIRRED(1))
C OUTPTS WILL CONTAIN THE VALUES ULTIMATELY PASSED TO THE
C OUTPUT ARRAY 'T'
C TMAT, TMA1, AND IPCINT ARE SCRATCH MATRICES.
C REAL OUTPTS(4), TIPCTR(4), TMA1(4,4), TMAT(4,4)
C ID WAS SET TO 1 BY CRAW (STMT 11) FOR THE FIRST TIME THRU DRETRV
C INUM=ID
C WHEN ID=C WE'RE DCON.
C IF(INUM.EQ.C)RETURN
C TO 4 IF THIS IS THE FIRST TIME THRU DCON
C IF(INUM.EQ.1)G0 TO 4
C THIS SECTION *FCPS* ALL STACKS.
C GO 100 I=1, IC
C 11=1D-1+1
C THE FIRST NEGATIVE DVEC(K) MEANS HEAD OF SOMETHING TO DO.
C IF(DVEC(I)).LT.0)GO TC 101
C CONTINUE
C DOES THIS ENTITY HAVE SUB-ELEMENTS?
C IF NOT, TC 102
C 101 IF(DVEC(I+1).EQ.C)G0 TC 102
C DVEC(I+1)=DVEC(I)+1-1
C INDICATE 'ON-THIS-ENTITY'.
C DVEC(ID)=-DVEC(ID)
C G0 4
C ZERO IT IN CVEC
C 102 CVEC(I)=0
C ID=1D-1
C IF DVEC FINALLY EMPTY, WE'RE DCON.
C IF(ID.EQ.0)RETURN
C NO MATRIX TO USE? IF NOT, RETURN.
C IF(IC.LE.0)RETURN 1
C SHOULD CURRENT MATRIX BE EXPIRED? IF NOT, RETURN
C IF(IDC.EQ.EXPRIE(IC)) RETURN 1
C IF GC, ZERO IT OUT
C 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
DO 112 K=1,4
TMAT(J,K)=0.
IF(J.EQ.K)TMAT(J,K)=1.
112 CONTINUE
C
IN THIS LOOP, BUILD TMAT WITH THE NECESSARY MULTIPLICATION.
DO 115 J=1,IC
IAKG=IDIREC(CURMAT(J)*2+2)/2+1
CALL MATPLT(TMAT,TABLE(IAKG),TMAT1)
IF(IC.EQ.J)RETURN 1
DO 115 K=1,4
CC 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 CURMAT OF ENTITY TO BE DRAWN.
C
ISTART IS THE LOCATION IN TABLE OF THIS ENTITY.
C
IGO IS THE TYPE OF ENTITY.
C
ISTART=IDIREC(-DVEC(INUM)*2+2)
IGO=IDIREC(-DVEC(INUM)*2+1)
CC TO(1CC0,2CC0,3CC0,4CC0), IGO
CC
OBJECT RETRIEVAL - SET UP LIMITS IN 'POINT' FOR THIS OBJECT
1000 ILCWR=ISTART+3
IUUPR=ILCWR+(ITAELE(ISTART+1)-1)*2
C
CC 10C9 J=ILCWR,IUUPR,2
NDX=ITABLE(J)+1
KUUPR=NDX+3
K1=0
C
COPY THE NEXT PCINT IN THE OBJECT INTO TCPINT
DO 1011 K=NDX,KUUPR
K1=K1+1
1011 TCPINT(K1)=PCINT(K)
C
INCREMENT PCINTS-TO-BE-DRAWN CCLATER
NPTS=NPTS+1
C
IF IC < 1, PCINT IS TTRANSFORMEC
C
IF IC = 1, PCINT IS TO BE TRANSFORMED BY ONLY ONE MATRIX
C
IF IC > 1, PCINT IS TO BE TRANSFORMED BY THE MATRIX PRODUCT IN T
C
C
UNTRANSFORMEC PCINT
IF IC=1)1015,1014,1012
1019 CC 1013 K1=1,4
1013 CLTPTS(K1)=TCPINT(K1)
CL TO 1010
C
POINT TO BE TRANFORMEC BY TMAT
1012 CALL PTMLT(CLTPTS,TCPINT,TMAT)
GC TO 1010
C
POINT TO BE TRANFORMEC BY ONE MATRIX ONLY - TAKEN STRAIGHT
C
FROM TABLE.
1014 IARG=IDIREC(CURMAT(1)*2+2)/2+1
CALL PTMLT(CLTPTS,TCPINT,TABLE(IARG))
1016 IF(ABS(CLTPTS(41)).LT..001)GC TC 2CC1
C
INCREMENT CLTMLT VECTOR INDEX, CHECKING FOR OVERFLOW.
IL=IL+2
IF(IL.GT.235)GC TO 1235
HERE'S THE SCALE FACTOR CALLING ITS THING.

IX AND IY ARE SET BY VIEWXYZ, VIEWZ, AND VIEWZY. DEFAULT SETTING IS X-Y PROJECTION.

CHECK THE DRAWING LIMITS - USED BY OUTPUT FOR SCALING.

T(IL)=OUTPTS(1X)/OUTPTS(4)
T(IL+1)=CUTPTS(IY)/CUTPTS(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)

TYPE-OF-LINE SWITCH (FIRST ARGUMENT TO "CALL POINTS")

1CC9 IT(1L/2+1)=ITABLE(IJ+1)

ZERO OUT THE LAST ENTRY IN DVEC.
DVEC(ID)=0

DECREMENT CVEC INDEX
ID=ID-1
RETURN 1

1235 IF(BATCH) CALL LCCINT(2)
PRINT 1236
1236 FORMAT('0***DRAW LERKCR - ONLY 273C PCINTS/DRAWING.')
RETURN

ASSEMBLY RETRIEVAL
3000 NUM=ITABLE(ISSTART+3)

GET THE # OF ENTITIES IN THIS ENTITY.
DVEC(ID+1)=NUM

SET THE LCCP LIMITS TO DIG THE ENTITIES OUT OF TABLE
NLWR=ISSTART+5
NUPPR=NLWR+NUM-1

WE'LL PUT THE PFENTER TO EACH ENTITY INTO DVEC
IC=ID+NUPRP+1

KEEP TRACK OF HOW BIG IC GETS. THIS WAS A DESIGN PARAMETER.

BY PRINTING IT OUT (ALONG WITH ICMAX), FIND OUT HOW BIG DIMENSION LIMITS FOR DVEC AND CURMAT SHOULDC BE
IF(IC.GT.ICMAX) ICMAX=IC
IC=IC+1
3200 I=NLWR,NUPPR
3100 I=I-1

PICK UP FCINTER
3200 DVEC(I1)=ITABLE(I)
RETURN 1

RETRIEVAL OF TRANSFORMED ENTITIES
4000 IF(IC.GT.ICMAX) ICMAX=IC
IC=IC+1

PUT POINTER TO THE XFORM IN CURMAT
CURMAT(IC)=ITABLE(ISSTART+1)
DVEC(ID+1)=1

SET UP DVEC WITH PTR TO 'SOURCE' ENTITY
DVEC(ID+2)=ITABLE(ISSTART+2)

INDICATE WHEN THE MATRIX JUST SPECIFIED IS TO 'EXPIRE' - THAT IS SHOULD NOT BE MULTIPLIED INTO PCINTS ANY MORE IN THIS DRAWING.
EXPIRED(IC)=IC
ID=ID+2
IF(ID.GT.ICMAX) ICMAX=IC

IF = 1, THIS IS FIRST MULTIPLICATION LEVEL.
IF(IC.GT.1) GC TO 4106
IF(IC.LE.1) RETURN 1

GET LOCATION OF VALUES IN TABLE.
K=ID*IREC(CURMAT(1)*2+2)/2
CC 4107 J=1,4
CC 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 CCWA, MULTIPLY THE NEW MATRIX
C  INTO WHAT CAME BEFORE.
4106 CC 4105 J=1,4
  DO 4105 I=1,4
4105 TMAT(I,J)=TMAT(I,J)
  IARG=IDIREC(CURMAT(IC)*2+2)/2+1
  CALL MATPL(I, TMAT, TABLE(IARG), TMAT)
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 LCCOUNT(3)
  PRINT 2002,CLTPS(3)
2002 FORMAT('C***CRAWL ERRCR - ATTEMPT TO PLACE EYEPONT IN SAME X-Y* /
  * Z PLANE AS POINT TO BE DRAWN. EYEPONT WAS AT Z = ',F10.3)
  CALL ABRFTZ(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 PAD

C THIS FUNCTION SEARCHES THE CHARACTER STRING PASSED AS ARGUMENT
C FOR AN 'a' (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'T AS CONTAIN MISC. LEFT-OVER TRASH - VERY
C UNDESIRABLE ON COMPUTERS. 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 KEYPRESSED BY THE USER.
REAL FUNCTION PAD*8 ( NAM1)
REAL*8 NAM,NAM1
INTEGER GET1,VAL/2CC0CC0CC7C/,TNAM,PLT1,ACRCF
NAM=NAM1
CC 10 I=1,6
TNAM=GET1(ACRCF(NAM),I-1)
IF(TNAM.EQ.VAL)GO TO 20
10 CONTINUE
  PAD=NAM
  RETURN
20 CC 40 J=1,8
CALL PUT1(ACROF(NAM),J-1,' ')  
PAU=NAM  
RETURN  
END  

C C ARGUMENT CHECKING FUNCTIONS - INTEGER  
C  
C   FORTRAN IV (G LEVEL) RUNNING UNDER PTS DSES 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  
C   FUNCTION ICHKF(IANS)  
C   ICHKF CHECKS INTEGER VALUES FOR INTEGERNESS.  
C COMMON /VAR/ FTMX,PTPTR,TAEMAX,TAEPTR,CIRMAX,DIRPTR,  
C *PTCTR,PT,ITP,PTPTR,CT,OPEN,IRMAX,DIRPTR,PTCTR,PT  
C *,LNK,NPRT,BATCH  
C LOGICAL NSCAL,AMOVE,NSFFT,NAFRT,BATCH  
C INTEGER*2 PFTMAX,PTP PTR,TAEMAX,TABFT  
C INTEGER*4 PTCTR,OPEN,CIRMAX,DIRPTR,PTCTR,PT  
C REAL*8 GENAN  
C CATA MASK /ZCCGCOOO/  
C EQUIVALENCE (IANS1,AANS1),(IA,AANS2)  
C C USERS OF 360'S WCH ARE HAVING TRULELE IMPLEMENTING THESE  
C C FUNCTIONS ARE WELCOME TO ADDRESS THEIR QUESTIONS TO US.  
C IAAS1=IANS  
C IF(IANS1)=5,2  
C LAND IS A 'BIT-WISE' LOGICAL FUNCTION. A BIT IS SET IN  
C THE RETURNED VALUE OF LAND IFF THE CORRESPONDING BIT  
C IS ON IN BOTH ARGUMENTS.  
C 1 IF(MASK-LAND(AANS1,MASK))4,3,4  
C 2 IF(LAND(AANS1,MASK))4,3,4  
C 5 ICHKF=0  
C RETURN  
C 3 ICHKF=IANS1  
C RETURN  
C 4 ICHKF=AANS1  
C IF(BATCH)CALL LCCSNT(1)  
C IF(NPRT)PRINT 500,AANS1,ICHKF  
C 500 FORMAT(' ***WARNING: UNEXPECTED DECIMAL POINT FOUND.','F8.1  
C 1,' CHANGE TO ',I6)  
C CALL ABCTRZ(8)  
C RETURN  
C C ARGUMENT CHECKING FUNCTIONS - REAL  
C  
C C ENTRY CHECKF(AANS)  
C C CHECKF CHECKS FLOATING POINT ARGUMENTS.  
C AANS2=AANS  
C IF(IA)=0,5C,2C  
C 10 IF(MASK-LAND(IA,MASK))30,4C,3C  
C 2C IF(LAND(IA,MASK))30,4C,3C  
C 5C CHECKF=0  
C RETURN  
C 3C CHECKF=AANS  
C RETURN  
C 40 CHECKF=IA  
C IF(BATCH)CALL LCCSNT(1)  
C IF(NPRT)PRINT 510,IA,CHECKF
C

C AUTOMATIC OBJECT-ASSEMBLY CLSER

C CLOSE MERELY CHECKS "OPEN" TO DETERMINE WHAT "MODE" IS IN EFFECT
C TYPE 1 ==> OBJECT DEFINITION UNDER WAY
C TYPE 2 ==> MATRX COMBINATION (VIA CMXMAT) UNDER WAY
C TYPE 3 ==> ASSEMBLY DEFINITION UNDER WAY
C TYPE 4 ==> INIT WAS NOT CALLED PRIOR TO THIS
C TYPE 0 ==> CLEAR SAILING - SHOULD NEVER BE ZERO IF GUT HRE

SUBROUTINE CLSER
COMMON /ATAEL/, ICIREC(600), PCINT(15000), TABLE(57000)
COMMON /VAR/, FMAX, PTPTR, SMAX, TAEPR, DIRMAX, DIRPTR,
*PTCTR, PT, TIP, TPTCTR, OPEH, NAMN, NCAIN, NMVE, NSHFT
*, LNM, NPT, BATCH
COMMON /CLEE/ TC(4), XNAME, I8H
LOGICAL ASCAL, NMCVE, NSHFT, NFRT, BATH
INTEGER*2 IPCINT(10000), ITAELE(10000), PMAX, PIPTR, TABMAX, TABPTR
INTEGER*4 IDIREC, OPEH, DIRMAT, DIRPTR, PTCTR, PT, TPTCTR
REAL*8 CIREC(400), UBJNAM, GNach, XNAME
EQUIVALENCE (1POINT(1), PCINT(1)), (1TAEL(1), TABLE(1)),
* (DIREC(1), I1IREC(1))
GC TO(10000, 2000, 3000, 4000), OPEN

100 IF(BATCH)CALL LCCLAT(2)
IF(NPT)PRINT 101, DIREC(DIRPTR-2)
101 FORMAT(* **WARNING: FAILURE TC FINISH OBJECT ***, AB, 1H/)*
I ' FINCBJ CALLED FCR YCL.*')
CALL FINCBJ
CALL ABCRTZ(11)
RETURN

300 IF(BATCH)CALL LCCCAT(2)
IF(NPT)PRINT 301, DIREC(DIRPTR-2)
301 FORMAT(* **WARNING: FAILURE TC FINISH ASSEMBLY ***, AB, 1H/)*
I ' FINASM CALLED FCR YCL.*')
CALL FINASM
CALL ABCRTZ(12)
RETURN

200 IF(BATCH)CALL LCCLAT(2)
IF(NPT)PRINT 200, XNAME

500 FORMAT(* **WARNING: FAILURE TC ENC CMXMAT(***, AB, 2H/)*
I ' ENCMCB CALLED FCR YCL.*')
CALL ENCMCB
CALL ABCRTZ(13)
RETURN

400 IF(BATCH)CALL LCCCLAT(2)
IF(NPT)PRINT 401

4C1 FORMAT(* **WARNING - FAILURE TC CALL INIT.*')
CALL INIT
CALL ABORTZ(14)
RETURN
END

C MATMLT
C
C MATMLT MULTIPLIES TWO FOUR-BY-FOUR MATRICES TOGETHER
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 PCINT MULTIPLICATION ROUTINE
C
PTMLT MULTIPLIES A CNE-BY-FCUR BY A FCUR-BY-FOUR
C
C = P * F
ENTRY PTMLT(C,P,F)
REAL*4 C(4),P(4),F(4,4),TA(4,4)
DO 10 I=1,4
10 C(I)=0.
DO 20 I=1,4
DO 20 J=1,4
20 C(I)=D(I,J)+F(J)*F(I,J)
RETURN
ENTRY ICENT(TA)
IDENT SETS UP A 4-BY-4 IENTITY MATRIX
DO 3 I=1,4
DO 3 J=1,4
3 IF(I.EQ.J)TA(I,J)=1.
CONTINUE
RETURN
END
C
C RETRIEVAL SUBROUTINE
C
RETREV GICS CLT OF THE TABLES AND DISPLAYS THE VALUE ASSOCIATED
WITH THE NAME GIVEN AS ARGUMENT. FCUR TYPES ARE POSSIBLE:
A) OBJECT (TYPE 1)
B) ASSEMBLY (TYPE 3)
C) TRANSFORMATION MATRIX (TYPE 2)
D) TRANSFORMED OBJECT OR ASSEMBLY (TYPE 4)
SUBROUTINE RETREV(TSTNAM,*)
COMMON /ATAEL/IICR(8C0), PCINT(5C0), TABLE(5000)
COMMON /VAR/PMAX, PTPTIR, IAEFTR, DIRMX, DIRPTR,
*PTCTR, PT, ITP, TPTCTR, OPEP, NASP, GEMNAM, NSCAL, NMOVE, NSHFT
*, LAM, NPT, BATC
LOGICAL NSCAL, NMOVE, NSHFT, NFR, BATC
INTEGER*4 IPICT1(10000), ITABLE1(10000), PTMAX, PTPTIR, TABMAX, TABPTR
INTEGER*4 IICR, OPEN, DIRMX, DIRPTR, PTCTR, PT, TPTCTR
REAL*8 LIREC(4C0), OBJNAM, GEMNAM
EQUIVALENCE (IPICT1(I), PCINT(I)), (ITABLE1(I), TABLE(I)),
* (DIREC(I), IICR(I))
REAL*8 TSTNAM,FCLND, XFCRAC, NAM,PAC
C DUE THE BELONG HERE (SHOULD WE HAVE FINISHED SOMETHING UP?)
IF(OPEN.NE.C) CALL CLOSE
OPEN=0
NAM=PAD(TSTNAM)
C FIND THE NAME. TYPE IN ITYP, LG.COMAT IN I-FIND
C IF NOT FOUND, GO TO 558
CALL FNCAM(IFINC,NAM,ITYP,ES98)
C INDEX INTO THE NAMES TABLE
ISTART=ICIREC(IFINC*2+2)
IGO=ITYP
C GO TO THE RIGHT PLACE.
GO TO(100C,200C,300C,400C),IGC
558 IF(BATCH)CALL LCCOUNT(2)
PRINT 10,NAM,NAM
1C FORMAT(*C***CRAWL ERRCR=***,AE,2H(*,Z16,*) NOT FOUND BY RETREV*)
CALL ABCTZ(10)
RETURN 1
C GET THE NAME FFCM STORAGE
1000 FOUND=DIREC(ITALLE(START-1))
C SET UPPER AND LOWER BOUNDS CN CC LCCP TC DIG OUT POINT VALUES
ILCWR=ISTART+3
IUPPR=ILCWR+(ITALLE(START+1)-1)*2
IF(BATCH)CALL LCCOUNT(3)
PRINT 11C1,FCCND,ITALLE(START+1)
1101 FORMAT(***OBJE **'***A8,'**' HAS 'I4,' FCINTS:**)
' Switch x'x','y'x','z'x','w')
C INDEX THROUGH FCINTS TABLE SETTING UP PRINT LINES
DO 1010 J=ILCWR,IUPPR,2
NCX=ITALLE(J+1)
KUPPR=NDX+2
IF(BATCH)CALL LCCOUNT(1)
1010 PRINT 11C2,ITALLE(J+1),(FCINT(K),K=NCX,KUPPR)
1102 FORMAT(3X,12,4(2X,F7.2))
RETURN
C GET THE NAME FFCM THE TABLE
2000 FOUND=DIREC(ITALLE(START-1))
C IF(BATCH)CALL LCCOUNT(6)
PRINT 2102,FCLAC
21C2 FORMAT(***TRANS FORMATION MATRIX ***A8,**' IS:**)
C SET THE UPPER AND LOWER BOUNDS CN PRINT STATEMENT TO
C DISPLAY THE VALUES IN THE MATRIX
ILCWR=(ISTART+2)/2
IUPPR=ILCWR+15
PRINT 21C3,(TABLE(K),K=ILCWR,IUPPR)
2103 FORMAT(2X,F7.3,2X,F7.3,2X,F7.3,2X,F7.3,2X,F7.3)
RETURN
C GET THE NAME FFCM THE TABLE
3000 FOUND=DIREC(ITALLE(START-1))
C NUM IS THE NUMBER OF ELEMENTS IN THE ASSEMBLY
NUM=ITALLE(START+3)
IF(BATCH)CALL LCCOUNT(2)
PRINT 31C1,FCCND,NUM
31C1 FORMAT(*CSIMPLE ASSEMBLY ***A8,**' HAS 'I3,' ELEMENTS:**)
C SET DC LCCP LIMITS
NL=$2348+5$
NUPPR=NL+NUM-1
DC 3103 K=NL,WALPPR
IF(BATCH)CALL LCCOUNT(1)
C USE ITABLE(K) AS SUBSCRIPT FCR NAME TABLE
3103 PRINT 3102,DIREC(ITALLE(K))
3102 FORMAT(2X,A8)
RETURN
C GET THE NAME OF THE ENTITY IN CLESTICA
4000 FOUND=DIREC(ITALLE(START-1))
GET THE NAME OF THE TRANSFORMATION MATRIX ASSOCIATED WITH IT
XFOUND=DIRECT(IITAEL(ISTART+1))
GET THE NAME OF THE SOURCE ENTITY
NAME=DIRECT(IITABLE(ISTART+2))
PRINT THE WHOLE MESS OUT
IF (BATCH) CALL LCCOUNT
PRINT 4101, FCUNCD, NAME, XFUND
41C1 FORMAT(('Q*', 'A8', '= ' 'A8', '*' 'A8'))
RETURN
END

SUBROUTINE CMEMAT(OBJNAM)
COMMON /VAR*/PTMAX, PTPT, TMAX, TAEPT, DIRMAT, DIRPTR,
* PTCTR, PT, IPT, IFTCTR, CPEN, NASP, GEMNAM, NSCAL, NMOVE, NSHFT
*, LNM, N PTR, BATCH
COMMON /CLETC(4,4), XNAME, ISW
INTEGER*2 PTMAX, PTPT, TMAX, TAEPT, IFTCTR
INTEGER*4 DIRMAT, DIRPTR, PTCTR, PT, IPTCTR, CPEN, Q12, '80061234'/
1 MASK/Z00000CC00/, INCR/Z00CC06CC/
LICAL NMSCAL, NMOVE, NSHFT, NPTR, BATCH
REAL*8 TRTRATE, RCTX, RCTY, RCTZ, CSCALE, PERSPC, NEWNAM, GEMNAM, OBJNAM,
1 ROTXD, ROTYD, ROTZD, CTX, CTY, CTZ, RCT, RCT2, RCT3
EQUIVALENCE (G1, NEWNAM)
REAL TC(4,4), TB(4,4), TA(4,4)
IF (OPEN, NE, C) CALL CLOSE
C DOES HE BELONG HERE? SET CPEN TO CMEMAT STATUS
CPEN=2
XNAME=OBJNAM
C ISW=1 ==> THAT THE SINGLE-TRANSFORMATION-MATRIX-DEFINING
C FUNCTIONS WHEN USED WILL BE A PART OF A CMEMAT SEQUENCE, AS
C OPPOSED TO BEING A PART OF A CALL TO TRANSF.
ISW=1
C SET UP AN IENTITY MATRIX
DC 1 I=1,4
DC 1 J=1,4
TC(I,J)=C.
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 COLLECTED.
C TC WILL RECEIVE THE PRODUCT OF TA AND TB (WHERE TB carries
C THE CUMULATIVE PRODUCT FOR THE ENTIRE SEQUENCE.)
C TD IS A CCFY OF TC, WHICH WILL BE THE ARGUMENT TO NMTRAJ,
C ALONG WITH OBJNAM, WHEN ENCMEM IS FINALLY CALLED.
2 DC 3 J=1,4
DC 3 I=1,4
TC(I,J)=TC(I,J)
TB(I,J)=TC(I,J)
TA(I,J)=C.
IF (I.EQ.J) TA(I,J)=1.
3 CONTINUE
CMEMAT=NEWNAM
RETURN
C ALL THE SINGLE FUNCTIONS PERFORM IN THE SAME MANNER AS
C TRLATE, AN EXPRESSION IS PROVIDED FOR TRLATE ONLY.
ENTRY TRLATE(CX, CY, CZ)
C IF TRANSF IS USING IT, INITIALLY TA
IF(ISw.NE.1) CALL IDENT(TA)
  INSERT THE VALUES IN THE APPROPRIATE MATRIX LOCATIONS
  TA(4,1)=CHECKF(CX)
  TA(4,2)=CHECKF(CY)
  TA(4,3)=CHECKF(DZ)
C IF PART OF A CBMAT SEQUENCE, IC 55 TO MULTIPLY IT ALL TOGETHER
IF(ISw.EQ.1) GC TC 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 NMTRA (ACTUALLY AN ALTERNATE ENTRY)
CALL NMTRA1(NewNam,TA,&999)
TRLATE=NewNam
RETURN
ENTRY RCTXC(B)
A=3.14159*CHECKF(B)/18C.
GO TO 50
ENTRY RCT1(A)
ENTRY RCTX(A)
50 IF(ISw.NE.1) CALL IDENT(TA)
  TA(2,2)=CCS(CHECKF(A))
  TA(2,3)=SIN(CHECKF(A))
  TA(3,2)=-SIN(CHECKF(A))
  TA(3,3)=CCS(CHECKF(A))
IF(ISw.EQ.1) GC TC 99
  Q(1)=Q(1)+256
  IF(LAND(MASK,Q(1))).EQ.MASK) Q(1)=Q(1)+INCR
  CALL NMTRA1(NewNam,TA,&999)
  RUTx=NewNam
  RCTXD=NewNam
  RCT1=NewNam
RETURN
ENTRY ROTOE(B)
A=3.14159*CHECKF(B)/18D.
GO TO 51
ENTRY ROTO2(A)
ENTRY RUTY(A)
51 IF(ISw.NE.1) CALL IDENT(TA)
  TA(1,1)=CCS(CHECKF(A))
  TA(1,3)=-SIN(CHECKF(A))
  TA(3,1)=SIN(CHECKF(A))
  TA(3,3)=CCS(CHECKF(A))
IF(ISw.EQ.1) GC TC 99
  Q(1)=Q(1)+256
  IF(LAND(MASK,Q(1))).EQ.MASK) Q(1)=Q(1)+INCR
  CALL NMTRA1(NewNam,TA,&999)
  RCTY=NewNam
  RCTyD=NewNam
  RCT2=NewNam
RETURN
ENTRY RCTZC(E)
A=CHECKF(B)*3.14159/18G.
GO TO 52
ENTRY RTO3(A)
ENTRY RCTZ(A)
52 IF(ISw.NE.1) CALL IDENT(TA)
  TA(1,1)=CCS(CHECKF(A))
  TA(1,2)=SIN(CHECKF(A))
  TA(2,1)=-SIN(CHECKF(A))
TA(2,2)=COS(CHECKF(A))
IF(ISW.EQ.1) GC TO 99
Q(1)=Q(1)+256
IF(LAND(MASK,Q(1)).EQ.MASK)C(1)=C(1)+INCR
CALL NMTRAI(NEWNAM,TA,EC99)
RCTZ=NEWNAM
RCT3=NEWNAM
RCTZD=NEWNAM
RETURN
ENTRY CSCALE(S)
IF(ISW.EQ.1) CALL IDENT(TA)
TA(4,4)=CHECKF(S)
IF(ISW.EQ.1) GC TO 99
Q(1)=Q(1)+256
IF(LAND(MASK,Q(1)).EQ.MASK)C(1)=C(1)+INCR
CALL NMTRAI(NEWNAM,TA,EC99)
CSCALE=NEWNAM
RETURN
ENTRY PERSPC(CZ)
IF(ISW.EQ.1) CALL IDENT(TA)
C
CHECK AND SEE IF HE WANTS TO DIVIDE BY ZERO
IF(ISW.EQ.1. AND. ABS(CHECKF(CZ)).LT..CO1) GC TO 99
IF(ISW.EQ.0. AND. ABS(CHECKF(CZ)).LT..CC1) GC TO 98
TA(3,4)=-1./CZ
S8 IF(ISW.EQ.1) GC TO 99
Q(1)=Q(1)+256
IF(LAND(MASK,Q(1)).EQ.MASK)C(1)=C(1)+INCR
CALL NMTRAI(NEWNAM,TA,EC99)
IF(LAND(MASK,Q(1)).EQ.MASK)C(1)=C(1)+INCR
PERSPC=NEWNAM
RETURN
C
MULTIPLY IT ALL TOGETHER, AND GC BACK TO 2 TO MAKE COPIES
C
AND INITIALIZE THE ARRAYS WHICH NEED INITIALIZING.
S9 CALL MATMLT(TC,TA,TB)
GC TO 2
S99 RETURN
END
APPENDIX I

Alphabetical List of DRAWL Subroutines
<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABORT</td>
<td>D3</td>
<td>E1, E2, H4</td>
</tr>
<tr>
<td>CMIBMAT</td>
<td>D4</td>
<td>D5, D6, E1, E2, E3, H32</td>
</tr>
<tr>
<td>CSBRT</td>
<td>D4</td>
<td>D5, D6, H34</td>
</tr>
<tr>
<td>DISPLA</td>
<td>B22</td>
<td>C2, H18</td>
</tr>
<tr>
<td>DRAW</td>
<td>B16</td>
<td>10, 12, B18, B19, B20, B21, B22, C2, C4, C5, E2, E3, F1, F4, H4, H17</td>
</tr>
<tr>
<td>ENDCMB</td>
<td>D4</td>
<td>D5, D6, E1, H14</td>
</tr>
<tr>
<td>FINASM</td>
<td>B11</td>
<td>2, 6, 8, E1, F4, H14</td>
</tr>
<tr>
<td>FINOBJ</td>
<td>B8</td>
<td>2, 5, 12, E1, F1, F4, H4, H13</td>
</tr>
<tr>
<td>INCRPT</td>
<td>B7</td>
<td>2, 5, B2, B4, B6, D1, E1, E2, H8</td>
</tr>
<tr>
<td>INIT</td>
<td>B1</td>
<td>12, C2, C3, C4, C5, C6, C7, E1, F1, F4, H1</td>
</tr>
<tr>
<td>INVA</td>
<td>D1</td>
<td>D1, H3</td>
</tr>
<tr>
<td>INVA2</td>
<td>D1</td>
<td>D1, F4, H8</td>
</tr>
<tr>
<td>INVR</td>
<td>D1</td>
<td>D1, H8</td>
</tr>
<tr>
<td>INVR2</td>
<td>D1</td>
<td>D1, H9</td>
</tr>
<tr>
<td>LOCCT</td>
<td>D7</td>
<td>C7, H5</td>
</tr>
<tr>
<td>MOVE</td>
<td>C4</td>
<td>B16, C1, H4</td>
</tr>
<tr>
<td>NAMASM</td>
<td>B9</td>
<td>2, 6, 8, 12, B10, E2, E3, F4, H11</td>
</tr>
<tr>
<td>NAME</td>
<td>B10</td>
<td>2, 6, 8, 12, E1, E2, F4, C1, H12</td>
</tr>
<tr>
<td>NAMOBJ</td>
<td>B3</td>
<td>B6, H6</td>
</tr>
<tr>
<td>NAMOBJB</td>
<td>B2</td>
<td>2, 5, 12, B3, B4, B6, E2, E3, F1, F4, G1, H5</td>
</tr>
<tr>
<td>NAMOBX</td>
<td>B4</td>
<td>B6, B7, E1, H6</td>
</tr>
<tr>
<td>NAMTRA</td>
<td>B12</td>
<td>4, 8, 12, B13, D5, D6, E1, E2, F1, F4, C4, H9</td>
</tr>
<tr>
<td>NMTRAL</td>
<td>B13</td>
<td>E1, E2, H9</td>
</tr>
<tr>
<td>NOMOVE</td>
<td>C4</td>
<td>B16, C1, C5, H4</td>
</tr>
<tr>
<td>NOMPAG</td>
<td>C7</td>
<td>C1, H4</td>
</tr>
<tr>
<td>NOPRINT</td>
<td>C6</td>
<td>B1, C1, D7, H4</td>
</tr>
<tr>
<td>NOSCAL</td>
<td>C2</td>
<td>B16, B18, C1, H4</td>
</tr>
<tr>
<td>NOSHFT</td>
<td>C3</td>
<td>B16, B18, C1, H4</td>
</tr>
<tr>
<td>PAGE</td>
<td>C7</td>
<td>C1, H4</td>
</tr>
<tr>
<td>PERSPC</td>
<td>D4</td>
<td>D5, D6, H34</td>
</tr>
<tr>
<td>POINTS</td>
<td>B5</td>
<td>2, 5, 12, B2, B3, B7, D1, E1, E2, F1, F4, G1, C4, H7</td>
</tr>
<tr>
<td>PRNT</td>
<td>C6</td>
<td>B1, C1, H4</td>
</tr>
<tr>
<td>PRPNTOUT</td>
<td>B20</td>
<td>11, C2, C3, C4, H18</td>
</tr>
<tr>
<td>PRPTLT</td>
<td>B19</td>
<td>11, B20, C2, C3, C4, E2, E3, H18</td>
</tr>
<tr>
<td>PRPTPUT</td>
<td>B21</td>
<td>11, C2, C3, C4, H18</td>
</tr>
<tr>
<td>RETREV</td>
<td>D2</td>
<td>E3, H30</td>
</tr>
<tr>
<td>ROTX</td>
<td>D4</td>
<td>D5, H33</td>
</tr>
<tr>
<td>ROTXD</td>
<td>D4</td>
<td>D5, D6, H33</td>
</tr>
<tr>
<td>ROTY</td>
<td>D4</td>
<td>D5, H33</td>
</tr>
<tr>
<td>ROTYD</td>
<td>D5</td>
<td>D5, H33</td>
</tr>
<tr>
<td>ROTZ</td>
<td>D5</td>
<td>D5, H33</td>
</tr>
</tbody>
</table>
### APPENDIX I - INDEX OF DRAWL SUBROUTINES (cont'd.)

<table>
<thead>
<tr>
<th>Name</th>
<th>Definition</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROTZD</td>
<td>D5</td>
<td>D5, H33</td>
</tr>
<tr>
<td>ROT1</td>
<td>D4</td>
<td>D5, H33</td>
</tr>
<tr>
<td>ROT2</td>
<td>D4</td>
<td>D5, H33</td>
</tr>
<tr>
<td>ROT3</td>
<td>D5</td>
<td>D5, H33</td>
</tr>
<tr>
<td>SCAL</td>
<td>C2</td>
<td>B16, B18, C1, C3, C4, H4</td>
</tr>
<tr>
<td>SHFT</td>
<td>C3</td>
<td>B16, B18, C1, C4, H4</td>
</tr>
<tr>
<td>SHOW</td>
<td>B18</td>
<td>10, P21, C2, C3, E2, E3, H18</td>
</tr>
<tr>
<td>TDUMP</td>
<td>D2</td>
<td>E3, H3</td>
</tr>
<tr>
<td>TRANSF</td>
<td>B14</td>
<td>4, 10, 12, D6, E2, F1, F4, C1, C4, H15</td>
</tr>
<tr>
<td>TRLATE</td>
<td>D4</td>
<td>D5, D6, H32</td>
</tr>
<tr>
<td>VIEWXY</td>
<td>C5</td>
<td>B16, C1, H27</td>
</tr>
<tr>
<td>VIEWZX</td>
<td>C5</td>
<td>B16, H27</td>
</tr>
<tr>
<td>VIEWZY</td>
<td>C5</td>
<td>B16, H27</td>
</tr>
<tr>
<td>VISA</td>
<td>D1</td>
<td>D1, F4, H8</td>
</tr>
<tr>
<td>VISA2</td>
<td>D1</td>
<td>D1, H8</td>
</tr>
<tr>
<td>VISR</td>
<td>D1</td>
<td>D1, H8</td>
</tr>
<tr>
<td>VISR2</td>
<td>D1</td>
<td>D1, H9</td>
</tr>
</tbody>
</table>
### 1. ORIGINATING ACTIVITY (Corporate author)

UNIVERSITY OF MICHIGAN
CONCOMP PROJECT

### 2a. REPORT SECURITY CLASSIFICATION

Unclassified

### 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

August 1970

<table>
<thead>
<tr>
<th>7a. TOTAL NO. OF PAGES</th>
<th>7b. NO. OF REFs</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>

### 8a. CONTRACT OR GRANT NO.

DA-49-083 OSA-3050

<table>
<thead>
<tr>
<th>8b. ORIGINATOR'S REPORT NUMBER(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical Report 30</td>
</tr>
</tbody>
</table>

### 10. DISTRIBUTION STATEMENT

Qualified requesters may obtain copies of this report from DDC.

### 11. SUPPLEMENTARY NOTES

<table>
<thead>
<tr>
<th>12. SPONSORING MILITARY ACTIVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADVANCED RESEARCH PROJECTS AGENCY</td>
</tr>
</tbody>
</table>

### 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.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th>LINK B</th>
<th>LINK C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role</td>
<td>Role</td>
<td>Role</td>
<td>Role</td>
</tr>
<tr>
<td>WT</td>
<td>WT</td>
<td>WT</td>
<td>WT</td>
</tr>
</tbody>
</table>

- graphical languages
- drafting languages
- coordinate geometry
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 co-
ordinates 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 co-
ordinates 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=DIREPTR-2
[L. 19] 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.