Chapter 5

Catalogs

5.1 Overview

AIPS keeps a catalog with a directory which contains an entry for each data file and its associated extension files. The catalog header record is used to keep various pieces of information about the data in the main data file and keeps track of the number and types of extension files associated with the main data file. These catalog header records are kept in individual files. The intent of this chapter is to describe the contents of the catalog header and to describe the use of the routines that access the catalog header record.

The information in the catalog header record is patterned after the FITS format tape header, although it is not as flexible. The catalog header describes the order and amount of data and maximum and minimum values, etc.

AIPS data files have a structure very similar to the structure of data of FITS format tapes. An image consists of a rectangular array of up to 7 dimensions. Pixel locations must be evenly spaced along each axis, although a proper redefinition of the axis can usually make this possible. The header record contains the number of pixels along each axis, a label for each axis, the number of the reference pixel (may be a fractional pixel and need not be in the portion of the axis covered), the coordinate at the reference pixel, the coordinate increment between pixels and the coordinate rotation. The axes of images may be in any order.

The AIPS format for uv data is also similar to the FITS convention. Each data point has a number of "random parameters", usually "u", "v", time, baseline number etc., followed by a rectangular array similar to, but usually smaller than, an image data array. Up to 14 random parameters have labels kept in the catalog header. More than 14 random parameters can be used, but the labels for the fifteenth and following are lost.

Most tasks read an old data file, do some operation on the data and write a new data file. In this case, the task simply takes the old catalog header record and modifies it to describe the data in the new file.

AIPS also keeps a catalog of the images displayed on all display devices. This image catalog allows AIPS interactive verbs to use the display devices without having to find and read the original catalog header record.

5.2 Public and Private catalogs

AIPS catalogs may be either public, i.e., all files on a given disk are in the same catalog, or private, i.e., each user has a separate catalog on each disk. The stand-alone utility program, SETPAR, is used to specify which type is currently in use. The distinction is completely transparent to the programmer; all distinctions between the two types are hidden in ZPHFIL and the catalog routines.

5.3 File Names

AIPS data files, especially cataloged files, are referenced in a number of different ways. The following list summarizes the three basic ways of specifying AIPS data files:

1. AIPS logical names. The full AIPS logical file specification is given by disk number, file name, file class, file sequence number, file physical type, user number, and, for extension files, the version number. This is the fundamental way an AIPS user specifies a file; although some of these, such as physical type and user number, may not have to be specified directly. In a task, these values are used by CATDIR (which may be called by a higher level routine such as MAPOPN) to locate the desired file in the AIPS catalog using various default and wild-card conventions.

2. Disk and catalog number. Just as the AIPS user frequently uses the disk and catalog numbers to specify files using the verb GETNAME, programs usually keep track of cataloged files by means of the disk and catalog numbers, file types, and version numbers for extension files.

3. Physical name. The host operating system needs a name for the file for its own catalog. The allowed physical file specifications depends on the host operating system, so AIPS tasks use the Z routine ZPHFIL to create the physical name from the disk and catalog numbers, the file type and version, and the user number for systems with private catalogs. These physical names may be up to 48 characters long.

   An example from a VAX system with private catalogs is "DA0n:ttrcccvv.uuu", where n is the zero-relative disk drive number, DA0n: is a logical variable which is assigned to a directory, tt is a two character file type (e.g., "MA"), r is a data format version number (see Appendix A), ccc is the catalog slot number, vv is the version (01 for "MA", "SC" and "UV" files), and uuu is the user's number. All numbers are expressed in hexadecimal notation.

5.4 Data Catalog

The data catalog actually consists of many separate files. There is one directory file (type "CA") per user for private catalogs per disk drive. Each cataloged file has its own catalog header file ("CB").

5.4.1 Catalog Directory

Each catalog directory contains a one block (256-word) header and a number of catalog directory blocks. The header block contains principally the number of catalog blocks in the file; this is set when the file is initialized or expanded. The directory blocks contain a 32-byte reference to each catalog header record. The directory is used to speed catalog searches and also contains the map status words that register map file activity.

5.4.2 Header block

The format of the Header Block is as follows:

```
OFFSET  TYPE         DESCRIPTION
   0     I   Volume number of disk containing this catalog
   1     I   Unused
   2     I   Number of catalog blocks in this file
```

5.4.3 Directory Section

The Mth directory block contains NLPR entries, each NWPL words, indexing the NLPR∗(M-1)+1 to the NLPR∗M-1 catalog records. The first directory block is the 2nd block in the file. The parameters are given by NWPL = 11 and NLPR = 256/NWPL.

The description of a directory entry is as follows:

```
OFFSET LENGTH TYPE         DESCRIPTION
   0      1    I    User ID number; or -1 if slot is empty
   1      1    I    Map file activity status
```

```
   2      2    H(2) Date/Time file was cataloged
   4      1    I    User defined sequence number 1 to 9999
   5      3    H(3) User defined map name, 12 characters
   8      2    H(2) User defined map class, 6 characters
  10      1    H    Map type, 2 characters
    ...
```

5.4.4 Directory Usage

Map name and class are user defined character strings of 12 and 6 characters that can be used to identify and locate a specific map. The strings are stored as HOLLERITH strings together with the 2-character HOLLERITH string which identifies the "physical" map type, in their slots in the directory. The sequence number is similarly an arbitrary integer reference number.

The Map Status is an integer registering the activity of the map file itself.

```
STATUS = 0    => no programs are accessing the map file
       = n>0  => n programs are reading the map
       = -1   => one program is writing into the file
       = n<0  => 1 + n programs are reading the map, one
                 program is writing into the file.
```

Maintaining the integrity of the catalog entries is essential to ensure reliable access to the cataloged files. Thus certain rules should be followed when using the catalog. These rules are coded in to the utility routines described below; these routines should be used when at all possible to access the catalog.

Rules:

1. Take exclusive use of the catalog whenever you access it. The required operation should be done quickly and then the catalog file should be closed and released.

2. The status word must be monitored to see if an intended catalog or map operation will disturb an (asynchronous) operation already in progress.

Specifically: Do not modify a catalog block, nor write into a map file which is not in a rest state (STATUS = 0).

If you intend to write into a map and STATUS = 0, change the status to "WRITE" (STATUS = -1) before releasing exclusive use of the catalog.

If you intend to read a map file or catalog block, check to see if someone else is writing on it (STATUS < 0). If so, decide whether this is acceptable to your program. If so, modify the status to "READ":

```
STATUS = 1 + STATUS if STATUS > 0
STATUS =-1 + STATUS if STATUS < 0.
```

Clear status when you have finished your operation. If you were reading, reverse the process just described. If you were writing, STATUS = - (1 + STATUS).

5.4.5 Structure of The Catalog Header Record

The catalog header block is a fixed format data structure 512 bytes long (one byte is defined in AIPS as half an integer). The catalog header block contains double and single precision floating point numbers, integers, and hollerith strings. The catalog header record is accessed by equivalencing integer, hollerith, real and double precision arrays, and obtaining the information from the array of the appropriate data type. Since the structure of the catalog header is subject to change we use pointers for the different arrays that are computed by VHDRIN. These pointers are kept in a common invoked with the INCLUDE DHDR.INC.

The uses of the pointers are given in the following table. In this table, the term "random parameters" refers to the portion of a uv data record that contain u, v, w, time, baseline etc.; the term "indeterminate" pixel means a pixel whose value is not given.

```
TYPE   POINTER      DESCRIPTION

H(2)    KHOBJ       Source name
H(2)    KHTEL       Telescope, i.e., 'VLA'
H(2)    KHINS       e.g., receiver or correlator
H(2)    KHOBS       Observer name
H(2)    KHDOB       Observation date in format 'DD/MM/YY'
H(2)    KHDMP       Date map created in format 'DD/MM/YY'
H(2)    KHBUN       Map units, i.e., 'JY/BEAM '
H(2)(14)KHPTP       Random Parameter types
        KIPTPN= 14  Max. number of labeled random paramaters
H(2)(7) KHCTP       Coordinate type, i.e., 'RA---SIN'
        KICTPN= 7   Max. number of axes
D(7)    KDCRV       Coordinate value at reference pixel
R(7)    KRCIC       Coordinate value increment along axis
R(7)    KRCRP       Coordinate Reference Pixel
R(7)    KRCRT       Coordinate Rotation Angles
R       KREPO       Epoch of coordinates (years)
R       KRDMX       Real value of data maximum
R       KRDMN       Real value of data minimum
R       KRBLK       Value of indeterminate pixel (real
                     maps only)
I       KIGCN       Number of random par. groups.
                    This is the number of uv data
                    records.
I       KIPCN       Number of random parameters
I       KIDIM       Number of coordinate axes
I(7)    KINAX       Number of pixels on each axis
I       KIIMS       Image sequence no.
H(3)    KHIMN       Image name (12 characters)
        KHIMNO= 1   Character offset in HOLLERITH string
H(2)    KHIMC       Image class (6 characters)
        KHIMCO= 13  Character offset in HOLLERITH string
H       KHPTY       Map physical type (i.e., 'MA','UV') (2 char)
        KHPTYO= 19  Character offset in HOLLERITH
I       KIIMU       Image user ID number
I       KINIT       # clean iterations
R       KRBMJ       Beam major axis in degrees
R       KRBMN       Beam minor axis in degrees
R       KRBPA       Beam position angle in degrees
I       KITYP       Clean map type: 1-4 => normal,
                    components, residual, points.
                    For uv data this word contains a
                    two character sort order code.
I       KIALT       Velocity reference frame: 1-3
                    => LSR, Helio, Observer +
                    256 if radio definition.
D       KDORA       Antenna pointing Right Ascension
D       KDODE       Antenna pointing Declination
D       KDRST       Rest frequency of line (Hz)
```

```
D       KDARV       Alternate ref pixel value
                    (frequency or velocity)
R       KRARP       Alternate ref pixel location
                    (frequency or velocity)
R       KRXSH       Offset in X (rotated RA) of phase center
R       KRYSH       Offset in Y (rotated Dec) from tangent pt.
H(20)   KHEXT       Names of extension file types (2 char)
        KHEXTN= 20  Max number of extension files
I(20)   KIVER       Number of versions of corresponding
                    extension file.
```

The actual values of the pointers depend on the size of the various data types and are computed in the routine VHDRIN. Note that VHDRIN should be called *after* ZDCHIN is called because it uses values set by ZDCHIN. VHDRIN has no call arguments.

The name of the pointer tells which data type array the data is to be read from: KInnn indicates the integer array, KHnnn indicates the hollerith array, KRnnn indicates the real array, and KDnnn indicates the double precision array. Conversion of HOLLERITH data to and from CHARACTER variables is done using routines H2CHR and CHR2H. The Name, class, and physical type are contained in HOLLERITH strings as are the labels of the regular and random axes. This is best explained by an example:

```
      INTEGER   NDIM1, INDEX
      REAL      CRPIX2
      CHARACTER CLASS*6, ALABE2*8
      DOUBLE PRECISION CRVAL3
C                                     Include for header pointers
      INCLUDE 'INCS:DHDR.INC'
C                                     Include for catalog header
C                                     common
      INCLUDE 'INCS:DCAT.INC'
          .
          .
          .
C                                     Get the dimension of
C                                     the first axis (I)
      NDIM1 = CATBLK(KINAX)
C                                     Get reference pixel
C                                     of second axis (R)
      CRPIX2 = CATR(KRCRP+1)
C                                     Get coordinate at reference
C                                     pixel on third axis. (R)
      CRVAL3 = CATD(KDCRV+2)
C                                     Copy axis label for third
C                                     axis (H array).
      INDEX = KHPTP + 2 * 2
      CALL H2CHR (8, 1, CATH(INDEX), ALABE2)
C                                     Copy image class.
      CALL H2CHR (6, KHIMCO, CATH(KHIMC), CLASS)
```

In the example above the catalog header block is obtained from a common defined in INCLUDE DCAT.INC. Many AIPS utility routines get the catalog header record from this common, so it is a good place to store it.

Keyword-Value Pairs

Arbitrary sets of keyword value pairs can be stored in an extension of the catalog header using routine CATKEY. The values may be of the following types: DOUBLE PRECISION, REAL, HOLLERITH (up to 8 char), INTEGER and LOGICAL. A description of CATKEY is given at the end of this chapter.

Image Files

An image consists of a single multidimensional (up to 7), rectangular array of pixel values. The structure of this array is defined by the catalog header record, which contains the number of dimensions (KIDIM) and the number of pixels on each axis (KINAX). All images are stored as REAL values.

The label for each axis is in a HOLLERITH string array pointed to by KHCTP. The coordinate increment between pixels must be a constant on each axis, and the array of axis increments is obtained using the pointer KRCIC. The array of coordinate reference pixels (the pixel at which the coordinate value is that pointed to by KDCRV) is pointed to by KRCRP; the reference pixel need not be either an integral pixel or in the range covered by the data. The coordinate values at the reference pixels are pointed to by KDCRV.

Each axis also has an associated rotation angle, but the only rotation currently supported is that on the plane of the sky. This rotation value is kept on the declination/Galactic latitude/Ecliptic latitude/Y axis and is the rotation of the coordinate system from north toward east.

Since there is no explicit provision made in the catalog header for such important parameters as position, frequency, and polarization, these are always declared as axes even if that axis contains only one pixel. This allows a place in the header record for these parameters.

Since the Stokes' axis is not inherently an ordered set, we use the following definitions for the values along the stokes' axis.

```
0  => beam           5  => Percent polarization
1  => I              6  => Fractional polarization
2  => Q              7  => Polarization position angle
3  => U              8  => Spectral index
4  => V              9  => Optical depth
```

Pixel values may be blanked using "magic value" blanking. The magic (stored) value for images is given by KRBLK (always 'INDE').

Each row of an image (first dimension) starts on a disk sector boundary (as defined on the local system) unless several rows may fit in a sector. In the latter case, as many rows as possible are put in a sector, but a row is not allowed to cross a sector boundary. Each plane in the image (dimension 3 and higher) starts on a sector boundary.

All angles in the header record are in degrees.

Uv Data Files

Uv data files consist of a sequence of interferometer visibility records each of which contains all data measured on a given baseline (pair of antennas) in a given integration period. The number of visibility records is given in the catalog header record by the integer value pointed to by KIGCN. The order of the visibility records are given by the two character code pointed to by KITYP. (More details of the sort order can be found in the chapter on disk I/O). All values are in floating point (except for compressed data).

Each visibility record consists of a number (KIPCN) of "random" parameters, followed by a data array similar to a miniature image. Any number of random parameters are allowed, but only the labels of 14 (KIPTPN) can be kept in the header. These labels are kept in Hollerith strings pointed to by KHPTP. The random parameters are used for values which vary "randomly" from visibility to visibility (i.e., u, v, w, time, baseline). The data array is described by the catalog header record in the same ways as for an image file.

The tangent point of the data (position for which the u, v, and w are computed) is kept as the RA and Dec axis in the data array. The offset in x and y (RA and dec after rotation) are pointed to by KRXSH and KRYSH. All angles in the catalog header record are in degrees.

Uv data may contain correlator based polarization or true Stokes' parameters. In the former case, the following Stokes' values are defined:

```
-1  => RR
-2  => LL
-3  => RL
-4  => LR
-5  => XX  Orientation of X and Y are defined in the
-6  => YY  AN table
-7  => XY
-8  => YX
```

Visibility records are allowed to span disk sector boundaries. More details about the uv data file format are given in the chapter on disk I/O.

Single Dish Data Files

Randomly sampled sky brightness measurments may be stored in data files which are similar to uv data files (the file "type" of the files is "UV"). The random parameters use for this type of data give the celestial position and beam or feed number rather than the location in the uv plane or a baseline. This type of data is described in more detail in next chapter.

5.4.6 Routines to Access the Data Catalog

MAPOPN and MAPCLS

There are a number of utility routines to access the catalog header record. In many cases, most of the catalog operations can be taken care of by the pair of routines MAPOPN and MAPCLS. MAPOPN will locate the correct catalog entry from a given name, class, disk, sequence and physical type following all default and wild-card conventions. MAPOPN then reads the catalog header record, opens the main data file and marks the catalog status word. Following a call to an initialization routine, the file can be read from or written to. After all I/O to the file is complete, MAPCLS will close the file, update the catalog header record if requested and clear the catalog status word for the file. A description of the call sequence of MAPOPN and MAPCLS is given at the end of this chapter.

CATDIR and CATIO

If MAPOPN and MAPCLS are not appropriate, then the use of more specialized routines is necessary. First the desired file must be located in the catalog directory. The routine CATDIR is the basic method of accessing the catalog directory. This routine will find the desired file given the name, class, etc. following the usual default and wild-card conventions. CATDIR returns the disk number and catalog slot number. Given a disk number and catalog slot number, CATIO can read or write a catalog header record and/or change the status word. Detailed descriptions of CATDIR and CATIO can be found at the end of this chapter.

5.4.7 Routines to Interpret the Catalog Header

There are a number of specialized routines which obtain information from the catalog header record. The following list gives a short description of each and detailed descriptions of the call sequence are found at the end of this chapter.

- AXEFND will return the axis number of a given type of random or regular axis.
- ROTFND returns the angle of rotation on the sky of either an image or uv data file.
- UVPGET obtains a number of pointers and other pieces of information which simplify accessing uv data.

5.4.8 Catalog Status

The AIPS catalog directory keeps a status word for each cataloged file. This status word is used to help prevent conflicting use of the file. The status may be marked as either "READ" or "WRIT"; the status of each file can be seen in AIPS by listing the catalog. A file can be marked "READ" multiple times, but a file marked "WRIT" cannot be marked "READ" or "WRIT" again, and a file marked "READ" cannot be marked "WRIT".

The use of the status word can complicate updating of the catalog header with CATIO. If the status of a file has been marked as "WRIT", then the opcode in the call to CATIO must be "UPDT". If the status is not marked, the opcode must be "WRIT" to update the catalog header block.

5.5 Image Catalog

5.5.1 Overview

The image catalog contains data for images stored on the TV device that identify the images, refer them back to their original map files, and specify scaling of the X-Y and intensity coordinates. There is a separate image catalog which performs the same functions for graphics devices (e.g., TEK4012 storage screens).

There is one image catalog file for each television device, whose physical name corresponds to ICv0000n, where v = version code and n = the device number (0 for graphics, 1 to n for TVs). They reside on disk 1 and must be created at AIPS installation, usually by FILAIP.

5.5.2 Data Structures

General: For each gray-scale image plane of the TV device, the IC contains N 1-block (256-word) records for cataloging up to N subimages, plus a (N-1)/51+1 block directory. The directory immediately precedes the catalog blocks for each image plane. For each TV graphics overlay plane there is one catalog block with no directory. These blocks follow immediately after the last gray-scale block.

The IC for pure graphics devices (called TK devices) has one image catalog block for each device in the system including all "local" TK devices followed by all remote-entry devices. Record number n in this file is associated with TK device number n (NTKDEV in /DCHCOM/ from include DDCH.INC).

The image catalog blocks themselves are essentially duplicates of the map catalog blocks except that scaling information replaces the extension file index of the map catalog.

The following is a description of the format of the directory block and the portions of the image catalog block which is different from the normal catalog header block.

```
   Directory Block (Gray-scale image)

OFFSET  TYPE    DESCRIPTION
     0   I    Sequence number of last sub-image cataloged
              on this plane
     1   I    Seq. no. of sub-image in slot 1; 0 if slot empty
     2   I(4) TV pixel positions of corners of 1st sub-image,
              x1,y1,x2,y2
     6   I    Seq. no. of sub-image in slot 2; 0 if empty
     7   I(4) TV pixel positions of corners of 2nd sub-image
     .
     .
     .
```

Catalog Block for each image or subimage:

Most of the Image Catalog block is identical to the map CAtalog block of the source of the image. (See section on CB files.) The information on antenna pointing, alternate frequency/velocity axis descriptions, and extension files (KIALT, KDORA, KDODE, KDRST, KDARV, KRARP, KHEXT and KIVER) is replaced in the IC by:

```
 TYPE POINTER DESCRIPTION

 R(2) IRRAN Map values displayed as min & max brightness.  I IIVOL
Disk volume from which map came I IICNO Catalog slot number of orig.
map I(4) IIWIN Map pixel positions of corners of displayed image (rel.
to orig. map) I(5) IIDEP Depth of displayed image in 7 - dimensional
map (axes 3 - 7) I(4) IICOR TV pixel positions of corners of image on
screen I IHTRA 2-char code for transfer function used to compute TV
brightness from map intensity values.  I IIPLT Code for type of plot.
I(31)IIOTH Misc. plot type dependent info.  (at the moment no more
than 20 used)
```

The standard pointer values are computed by VHDRIN and are available through the common /HDRVAL/ via include DHDR.INC. They are machine-dependent and are used in the same way as the normal catalog pointers.

5.5.3 Usage notes

We assume that single images only are stored on graphics planes; there is no directory.

When a gray-image plane is cleared, its directory is zeroed. As images are added to the plane, their coordinates are written into an open directory slot for that plane, along with the current value of the plane sequence number. The sequence number is then incremented. If an old image is completely overwritten by a new one, its directory slot is cleared. For partially overlapping images, the sequence # allows the user to select the one most recently loaded into a given part of the plane.

5.5.4 Subroutines

There are a number of routines to manipulate the image catalog. The following is a short description of each; detailed descriptions of the call sequences is given at the end of this chapter.

- YCINIT clears the Image Catalog for a given plane.
- YCOVER asks if there are any overlapped images in each quadrant visible.
- YCWRIT adds a new block to the catalog.
- YCREAD returns the block corresponding to a given TV pixel.
- TVFIND determines desired image, asks user if > 1 visible.

These routines expect the “plane number” as an argument. TV gray scale planes are numbered 1–NGRAY, TV graphics overlay planes are numbered (NGRAY+1)–(NGRAY+NGRAPH), and TK devices are referenced by any plane number greater than NGRAY+NGRAPH.

5.5.5 Image Catalog Commons

The COMMON /TVCHAR/ referenced by INCLUDE 'DTVC.INC' contains TV device characteristics such as:

```
NGRAY      # of gray-scale planes on this device
NGRAPH     # of graphics planes
MAXXTV(2)  Maximum number of pixels in x,y directions in image
```

The listing of DTVC.INC is given at the end of this chapter.

The common /DCHCOM/ (from DDCH.INC) contains two important parameters in this regard: NTVDE-V and NTKDEV. The subroutine ZDCHIN sets these to the actual number of such devices present locally. Then, the routines ZWHOMI (in AIPS only) and GTPARM (in all tasks) reset them to the device number assigned to the current user. ZWHOMI determines these assignments.

5.6 Coordinate Systems

Astronomical images are usually represented as projections onto a plane causing the true position on the sky of a pixel to be a nonlinear function of the pixel location. In a similar fashion, most spectral observations are done with evenly spaced frequency channels which results in a nonlinear relation between the velocity of a channel and the channel number. AIPS Memos Nos. 27 and 46 describe in great detail the approach AIPS uses to these problems. Much of the following sections is taken from these memos.

5.6.1 Velocity and Frequency

The physically meaningful measure in a spectrum is the radial velocity of a feature; unfortunately, observations are normally made using a uniform spacing in frequency (and may contain Doppler tracking to remove the effects of the earth's motion). Thus it is necessary to convert between frequency and velocity. The details of the conversion are in AIPS Memo No. 27 and will not be reproduced here. Conversion can be done using the routines described in the section on celestial positions. The following sections describe the naming conventions and the way in which the necessary information is stored in the catalog header block.

Axis Labels

The AIPS convention is to use the axis label to denote the axis type with the first four characters and the inertial reference system with the last four characters. The axis types currently supported are 'FREQ...' which is regularly gridded in frequency, 'VELO...' which is regularly gridded in velocity, and 'FELO...' which is regularly gridded in frequency, but expressed in velocity units in the optical convention.

The inertial reference systems currently supported are '-LSR', '-HEL', and '-OBS' indicating Local Standard of Rest, heliocentric, and geocentric. Others may be added if necessary.

Catalog Information

In addition to the normal axis coordinate information carried in the catalog header, described previously in this chapter, the catalog header record has provision for storing an alternate frequency axis type. The AIPS verb ALTDEF allows the user to switch the two axis definitions. The pointers for these values are given in the following:

```
KDRST    Rest frequency (Hz)
KRARP    Alternate reference pixel
KDARV    Alternate reference value
KIALT    axis type code. 1=>LSR, 2=>HEL, 3=>OBS
         (plus 256 if radio convention).
         0 implies no alternate axis.
```

5.6.2 Celestial Positions

The following sections will describe the AIPS conventions and routines for determining positions from images with different projections.

Axis Labels

The AIPS convention is to use the first four characters of the axis type and the second four characters to denote the projection. The standard nonlinear axis types are given in the following:

- RA−− denotes Right ascension
- DEC− denotes declination
- GLON denotes galactic longitude
- GLAT denotes galactic latitude
- ELON denotes Ecliptic longitude
- ELAT denotes Ecliptic latitude

The geometry used for the projection is given in the axis label using the codes given in the following list:

- −TAN denotes tangent projection. This projection is commonly used in optical astronomy.
- −SIN denotes sine projection. This projection is commonly used in radio aperture synthesis images.
- −ARC denotes arc projection. In this geometry, angular distances are preserved and it is commonly used for Schmidt telescopes and for single dish radio telescopes.
- −NCP denotes a projection to a plane perpendicular to the North Celestial Pole. This geometry is used by Westerbork.
- −STG denotes stereographic projection. This is the tangent projection from the opposite side of the celestial sphere.
- −AIT denotes Aitoff projection. This is used for very large fields.
- −GLS denotes Global sinusoidal projection. This is also used for very large fields.
- −MER denotes Mercator projection.

Determining Positions

There are a number of AIPS utility routines which help determine the position of a given location in an image. These routines use values in theINCLUDE DLOC.INC. A listing of this include can be found at the end of this chapter.

Position Routines The upper level position determination routines are briefly described in the following; details of the call sequences are given at the end of this chapter.

- SETLOC initializes the DLOC.INC INCLUDE based on the current catalog header block in the D-CAT.INC (CATBLK) common.
- XYPIX determines the pixel location corresponding to a specified coordinate value.
- XYVAL determines the coordinate value (X,Y,Z) corresponding to a given pixel location.
- FNDX returns the X axis coordinate value of a point given the Y axis coordinate value and the X axis pixel position of a point. Does rotations and non linear axes.
- FNDY returns the Y axis coordinate value of a point given the X axis coordinate value and the Y axis pixel position of a point. Does rotations and non linear axes.
- COORDT converts between celestial, galactic and ecliptic coordinates.

Include DLOC.INC The commons in INCLUDE DLOC.INC are used by the position routines and the plot labeling routines to keep constants needed for the coordinate transformation. The contents of these commons is described in the following:

```
RPVAL   D(4)    Reference pixel values
COND2R  D       Degrees to radians multiplier = pi/180
AXDENU  D       delta(nu) / nu(x) when a FELO axis is
                present.
GEOMD1  D       Storage for parameter needed for geometry.
GEOMD2  D                         "
GEOMD3  D                         "
GEOMD4  D                         "
RPLOC   R(4)    Reference pixel locations
AXINC   R(4)    Axis increments
CTYP    C(4)*20 Axis types
CPREF   C(2)*5  x,y axis prefixes for labeling
ROT     R       Rotation angle of position axes
SAXLAB  C(2)*20 Labels for axes 3 and 4 values
ZDEPTH  I(5)    Value of Idepth from SETLOC call
ZAXIS   I       1 relative number of z axis
AXTYP   I       Position axis code
CORTYP  I       Which position is which
LABTYP  I       Special x,y label request
SGNROT  I       Extra sign to apply to rotation
AXFUNC  I(7)    Kind of axis code
KLOCL   I       0-rel axis number-longitude axis
KLOCM   I       0-rel axis number-latitude axis
KLOCF   I       0-rel axis number-frequency axis
KLOCS   I       0-rel axis number-stokes axis
KLOCA   I       0-rel axis number-"primary axis" 3
KLOCB   I       0-rel axis number-"primary axis" 4
NCHLAB  I(2)    Number of characters in SAXLAB
```

Several of the above values need further explanation:

```
AXTYP   value = 0  no position-axis pair
              = 1  x-y are position pair
              = 2  x-z are position pair
              = 3  y-z are position pair
              = 4  2 z axes form a pair
CORTYP  value = 0  linear x,y axes
              = 1  x is longitude, y is latitude
              = 2  y is longitude, x is latitude
              = 3  x is longitude, z is latitude
              = 4  z is longitude, x is latitude
              = 5  y is longitude, z is latitude
              = 6  z is longitude, y is latitude
 LABTYP value = 10 * ycode + xcode
        code  = 0   use CPREF, CTYP
              = 1   use Ecliptic longitude
              = 2   use Ecliptic latitude
              = 3   use Galactic longitude
              = 4   use Galactic latitude
              = 5   use Right Ascension
```

```
                 = 6   use declination
   AXFUNC value = -1   no axis
                 = 0   linear axis
                 = 1   FELO axis
                 = 2   SIN projection
                 = 3   TAN projection
                 = 4   ARC projection
                 = 5   NCP projection
                 = 6   GLS projection
                 = 7   MER projection
                 = 8   AIT projection
                 = 9   STG projection
```

The KLOCx parameters have a value of -1 if the corresponding axis does not exist. If AXTYP is 2 or 3, the pointer KLOCA will always point at the z axis. In this case, SETLOC does not have enough information to prepare SAXLAB. The string must be computed later when an appropriate x,y position is specified.

5.6.3 Rotations

The use of one rotation angle per axis, as provided in the AIPS catalog header, is obviously not enough to completely describe an arbitrary rotation of the coordinate system. In practice, the only rotation currently used in AIPS is the rotation in the sky plane (projected RA and dec, galactic latitude and longitude, or ecliptic latitude and longitude). The rotation angle in this plane of the actual coordinate system of the image, in the usual astronomical north through east convention, is given on the axis corresponding to the declination, galactic latitude, or ecliptic latitude as appropriate.

Another convention followed in AIPS involving rotations is related to precession. As the earth precesses, the north-south line in a field will rotate; this causes a rotation in an image made of a given field on the sky. This "differential precession" will cause problems determining positions away from the field center and comparing images made at different epochs. To avoid this problem, the coordinate system used for the u-v data is rotated to the orientation as of the mean epoch (1950 or 2000).

5.7 Text of INCLUDE files

5.7.1 DCAT.INC

```
C                                                           Include DCAT.
C                                       catalog header common
      INTEGER   CATBLK(256)
      REAL      CATR(256)
      HOLLERITH CATH(256)
      DOUBLE PRECISION CATD(128)
      COMMON /MAPHDR/ CATBLK
      EQUIVALENCE (CATBLK, CATR, CATH, CATD)
C                                                           End DCAT.
```

5.7.2 DHDR.INC

```
C                                                           Include DHDR.
      INTEGER   KHOBJ, KHTEL, KHINS, KHOBS, KHDOB, KHDMP, KHBUN,
     *   KHPTP, KHCTP, KRCIC, KRCRP, KRCRT, KREPO, KRDMX, KRDMN, KRBLK,
     *   KHIMN, KHIMC, KHPTY, KRBMJ, KRBMN, KRBPA, KRARP, KRXSH, KRYSH,
```

```
     *   KHIMNO, KHIMCO, KHPTYO,
     *   KDCRV, KDORA, KDODE, KDRST, KDARV,
     *   KIGCN, KINIT,
     *   KIPTPN, KICTPN, KIEXTN,
     *   KIPCN, KIDIM, KINAX, KIIMS, KIIMU, KITYP, KIALT, KHEXT, KIVER,
     *   IRRAN, IIVOL, IICNO, IIWIN, IIDEP, IICOR, IITRA, IIPLT, IIOTH,
     *   KIRES, KIRESN
      COMMON /HDRVAL/  KHOBJ, KHTEL, KHINS, KHOBS, KHDOB, KHDMP, KHBUN,
     *   KHPTP, KHCTP, KRCIC, KRCRP, KRCRT, KREPO, KRDMX, KRDMN, KRBLK,
     *   KHIMN, KHIMC, KHPTY, KRBMJ, KRBMN, KRBPA, KRARP, KRXSH, KRYSH,
     *   KHIMNO, KHIMCO, KHPTYO,
     *   KDCRV, KDORA, KDODE, KDRST, KDARV,
     *   KIGCN, KINIT,
     *   KIPTPN, KICTPN, KIEXTN,
     *   KIPCN, KIDIM, KINAX, KIIMS, KIIMU, KITYP, KIALT, KHEXT, KIVER,
     *   IRRAN, IIVOL, IICNO, IIWIN, IIDEP, IICOR, IITRA, IIPLT, IIOTH,
     *   KIRES, KIRESN
C                                                            End DHDR.
```

5.7.3 DLOC.INC

```
C                                                            Include DLOC.
C                                              Position labeling common
      DOUBLE PRECISION RPVAL(4), COND2R, AXDENU, GEOMD1, GEOMD2, GEOMD3,
     *   GEOMD4
      CHARACTER CTYP(4)*20, CPREF(2)*5, SAXLAB(2)*20
      REAL      RPLOC(4), AXINC(4), ROT
      INTEGER   ZDEPTH(5), ZAXIS, AXTYP, CORTYP, LABTYP, SGNROT,
     *   AXFUNC(7), KLOCL, KLOCM, KLOCF, KLOCS, KLOCA, KLOCB,
     *   NCHLAB(2)
      COMMON /LOCATC/ CTYP, CPREF, SAXLAB
      COMMON /LOCATI/ RPVAL, COND2R, AXDENU, GEOMD1, GEOMD2, GEOMD3,
     *   GEOMD4, RPLOC, AXINC, ROT, ZDEPTH,
     *   ZAXIS, AXTYP, CORTYP, LABTYP, SGNROT, AXFUNC, KLOCL, KLOCM,
     *   KLOCF, KLOCS, KLOCA, KLOCB, NCHLAB
C                                                            End DLOC.
```

5.7.4 DTVC.INC

```
C                                                            Include DTVC.
      INTEGER   NGRAY, NGRAPH, NIMAGE, MAXXTV(2), MAXINT, LUTOUT,
     *   OFMINP, OFMOUT, SCXINC, SCYINC, MXZOOM, CSIZTV(2), TYPSPL,
     *   TVALUS, TVXMOD, TVYMOD, ISUNUM,
     *   TVDUMS(10),
     *   TVZOOM(3), TVSCRX(16), TVSCRY(16), TVLIMG(4), TVSPLT(2),
     *   TVSPLM, TVSPLC, TYPMOV(16), YBUFF(168)
      COMMON /TVCHAR/ NGRAY, NGRAPH, NIMAGE, MAXXTV, MAXINT, LUTOUT,
     *   OFMINP, OFMOUT, SCXINC, SCYINC, MXZOOM, CSIZTV, TYPSPL,
     *   TVALUS, TVXMOD, TVYMOD, ISUNUM,                TVDUMS,
     *   TVZOOM, TVSCRX, TVSCRY, TVLIMG, TVSPLT, TVSPLM, TVSPLC,
     *   TYPMOV, YBUFF
C                                                            End DTVC.
```

5.8 Routines

5.8.1 AXEFND

AXEFND determines the order number of an axis whose name is in the character string TYPE. It will work for either regular or random axes.

```
AXEFND (NCHC, TYPE, NAXIS, AXES, IOFF, IERR)
Inputs:
   NCHC      I    Compare only first NCHC characters of axis type
   TYPE      C*8  Axis type.
   NAXIS     I    the number of axes to search,
                     for uniform axes use: CATBLK(KIDIM) or KICTPN
                     for random axes use:  CATBLK(KIPCN) or KIPTPN
   AXES(*)   H    Catalog axis name list,
                     for uniform axes use: CATH(KHCTP)
                     for random axes use : CATH(KHPTP)
Output:
   IOFF      I    Axis offset ( zero relative axis number)
   IERR      I    Return error code, 0=>OK, 1=>could not find.
```

5.8.2 CATDIR

CATDIR manipulates catalog directory.

```
CATDIR (OP, IVOL, CNO, CNAME, CCLASS, SEQ, PTYPE, USID,
  *   STAT, BUFF, IERR)
Inputs:
   OP     C*4   searches find entry with specified data:
                'SRCH' high seq # (if SEQ 0), return things
                'SRNH' high seq # (if SEQ 0), NOT return things
                'SRCN' next match, return things
                'SRNN' next match, NOT return things
                'OPEN' = create a new slot (and init header file)
                'CLOS' = destroy a slot
                'INFO' = return contents of a slot
                'CSTA' = modify status of a slot
   IVOL   I     Disk volume containing catalog
                0 => all on searches, OPEN
   CNO    I     Slot number to begin: SRCN, SRNN, OPEN
                Ignored if IVOL = 0 : searches, OPEN
                Slot number to examine (solely): CLOS, INFO, CSTA
   CNAME  C*12  Map name: searches, OPEN, CLOS
   CCLASS C*6   Map type: searches, OPEN, CLOS
   SEQ    I     Map sequence number: searches, OPEN, CLOS
   PTYPE  C*2   Map physical type (2 chars): searches, OPEN, CLOS
   USID   I     User identification #: searches, OPEN, CLOS
   STAT   C*4   Status (OP=CSTA): READ, WRIT, CLRD, or CLWR
Outputs:
   CNO    I     Slot number found: searches, OPEN
   IVOL   I     If 0 on input, value actually used: searches, OPEN
```

```
CNAME  C*12  Map name: SRCH, SRCN, INFO
CCLASS C*6   Map type: SRCH, SRCN, INFO
SEQ    I     Map sequence number: SRCH, SRCN, INFO
PTYPE  C*2   Map physical file type: SRCH, SRCN, INFO
USID   I     User identification #: SRCH, SRCN, INFO
STAT   C*4   Status: INFO
BUFF   I(256)    Working buffer
IERR   I     Error return
             1 =>  can't open cat file or header file
             2 =>  input error
             3 =>  can't read catalog or header file
             4 =>  CLOSE blocked by non-REST status
             5 =>  end of catalog on OPEN or SRCH i.e.
                   no open slots or slot not found
             6 =>  on INFO requested slot not open
             7 =>  can't use WRIT status because now READ
             8 =>  on CLOSE the ID's don't match
             9 =>  Warning: read status added on a file
                   being written
            10 =>  Clear read/write when didn't exist warning
```

5.8.3 CATKEY

Reads or writes KEYWORDs from or to an AIPS image (or uv) header. The order of the keywords is arbitrary. Uses LUN 15, so any CA or CB files must be closed before calling this routine.

```
CATKEY (OPCODE, IVOL, CNO, KEYWRD, NUMKEY, LOCS, VALUES, KEYTYP,
  *   BUFFER, IERR)
Inputs:
   OPCODE   C*4        Operation desired, 'READ', 'WRIT',
                          'ALL ' => Read all.
                          'REED' => no error msg if some missing
   IVOL     I          File disk number
   CNO      I          File catalog block number
In/out:
   KEYWRD   C(*)*8     Keywords to read/write: output on ALL
   NUMKEY   I          Number of keywords to read/write.
                          Input on OPCODE='ALL' = max. to read.
                          Output on OPCODE='ALL' = no. read.
   LOCS     I(NUMKEY)  The word offset of first short integer
                       word of keyword value in array VALUES.
                       Output on READ, input on WRIT.
                       On READ this value will be -1 for keywords
                       not found.
   VALUES   I          The array of keyword values; due to word
                       alignment problems on some machines values
                       longer than a short integer should be copied,
                       eg. if the  5th keyword (XXX) is a R*8:
                            IPOINT = LOCS(5)
                            CALL COPY (NWDPDP, VALUES(IPOINT), XXX)
                       Output on READ, input on WRIT
   KEYTYP   I(NUMKEY)  The type code of the keywords:
                          1 = Double precision floating
                          2 = Single precision floating
```

```
                        3 = Character string (8 HOLLERITH chars)
                        4 = integer
                        5 = Logical
Output:
   BUFFER   I(256)     Scratch buffer
   IERR     I          Return code, 0=>OK,
                          1-10 => ZFIO error
                          19   => unrecognized data type.
                          20   => bad OPCODE
                          20+n => n keywords not found on READ.
                                  This produces messages at level 6
                                  suppress them w MSGSUP if needed
```

5.8.4 CATIO

CATIO reads or writes blocks in the map catalog header file.

```
CATIO (OP, IVOL, CNO, CATBLK, STAT, BUFF, IERR)
Inputs:
   OP       C*4      'READ' => get block into CATBLK
                     'WRIT' => put CATBLK onto disk catalog
                     'UPDT' => as WRIT but for use when the calling
                               program has previously set the
                               status to WRITE
   IVOL     I        Disk volume containing catalog (1 rel)
   CNO      I        Slot number of interest
   CATBLK   I(256)   Array to be written on disk: WRIT, UPDT
   STAT     C*4      Status desired for slot after operation
                         'READ','WRIT','REST' where REST => no
                         change of status is desired
Outputs:
   CATBLK   I(256)   Array read from disk: READ
   BUFF     I(256)   Working buffer
   IERR     I        Error code: 0 => ok
                        1 => cannot open catalog file
                        2 => input parameter error
                        3 => cannot read catalog file
                        4 => cannot WRIT/UPDT: file is busy
                        5 => did READ/UPDT, cannot add STAT = WRIT
                        6 => Warning on READ, file writing
                        7 => As 6, also added STAT=READ
                        8 => As 6, STAT inconsistent or wrong
                        9 => Warning: STAT inconsistent/wrong
The requested OP is performed unless IERR = 1 through 4.  The
final status requested is not set if IERR = 1 - 5, 8 - 9.  The
latter are probably unimportant.
```

5.8.5 COORDT

COORDT translates between types of coordinates:

```
COORDT (ITI, ITO, LONGI, LATI, EPOK, LONGO, LATO, IERR)
Inputs:
   ITI    I     Input type (1 Ra, Dec; 2 gal, 3 ecliptic)
```

```
   ITO    I     Output type
   LONGI  D     Input longitude
   LATI   D     Input latitude
   EPOK   R     Epoch of positions (used very simply with
                ecliptic coords only)
                1950 assumed in Galactic conversions!!!!!!!!!
Output:
   LONGO  D     Output longitude
   LATI   D     Output latitude
   IERR   I     error: 0 ok, 1 input error, 2 conversion fails
```

5.8.6 FNDX

FNDX returns the X axis coordinate value of a point given the Y axis coordinate value and the X axis pixel position of the point. Needed for rotations and non-linear axes (L-M).

```
FNDX (XPIX, YVAL, XVAL, IERR)
Inputs:
   XPIX   R     X pixel position
   YVAL   D     Y coordinate value
Output:
   XVAL   D     X coordinate value
   IERR   I     0 ok, 1 out of range, 2 bad type, 3 undefined
Common:
   Pos. parms in DLOC.INC must have been set up by SETLOC
```

5.8.7 FNDY

FNDY returns the Y axis coordinate value of a point given the X axis coordinate value and the Y axis pixel position of the point. Needed for rotations and non-linear axes (L-M).

```
FNDY (YPIX, XVAL, YVAL, IERR)
Inputs:
   YPIX   R     Y pixel position
   XVAL   D     X coordinate value
Output:
   YVAL   D     Y coordinate value
   IERR   I     0 ok, 1 out of range, 2 bad type, 3 undefined
Common:
   Pos. parms in DLOC.INC must have been set up by SETLOC
```

5.8.8 MAPCLS

closes a cataloged file, updates header on disk, clears catalog status.

```
MAPCLS (OP, IVOL, CNO, LUN, IND, CATBLK, CATUP, WBUFF, IERR)
Inputs:
   OP      C*4     OPcode used by MAPOPN to open this file
   IVOL    I       Disk volume containing map file
   CNO     I       Catalog slot number of file
   LUN     I       Logical unit # used for file
   IND     I       FTAB pointer for LUN
   CATBLK  I(256)  New catalog header which can optionally
                   be written into header if OP=WRIT or INIT
```

```
                    Dummy arguement if OP=READ
   CATUP   L        If TRUE, write CATBLK into catalog,
                    ignored if OP = READ
Outputs:
   IERR    I        0 = O.K.
                    1 = CATDIR couldn't access catalog
                    5 = illegal OP code
```

5.8.9 MAPOPN

MAPOPN opens a map file marking the catalog entry for the desired type of operation.

```
MAPOPN (OP, IVOL, NAMEIN, CLASIN, SEQIN, TYPIN, USID,
  *   LUN, IND, CNO, CATBLK, WBUFF, IERR)
Inputs:
   OP      C*4      Operation: READ, WRIT, or INIT where INIT is for
                    known creation processes (it ignores current file
                    status & leaves it unchanged).  Also: HDWR for
                    use when the header is being changed, but the
                    data are to be read only.
   LUN     I        Logical unit # to use
In/out:
   NAMEIN  C*12     Image name (name)
   CLASIN  C*6      Image name (class)
   SEQIN   I        Image name (seq.#)
   USID    I        User identification #
   IVOL    I        Input disk unit
   TYPIN   C*2      Physical type of file
Outputs:
   IND     I        FTAB pointer
   CNO     I        Catalog slot containing map
   CATBLK  I(256)   Buffer containing current catalog block
   WBUFF   I(256)   Working buffer for CATIO and CATDIR
   IERR    I        Error output: 0 = OK
                       2 = Can't open WRIT because file busy
                           or can't READ because file marked WRITE
                       3 = File not found
                       4 = Catalog i/o error
                       5 = Illegal OP code
                       6 = Can't open file
```

5.8.10 ROTFND

ROTFND finds the map rotation angle from a given catalog block.

```
ROTFND (CATR, ROT, IERR)
Inputs:
    CATR(*)   R    Map catalog header
Outputs:
    ROT       R    Map rotation angle (degrees)
    IERR      I    Error code. 0=>OK, 1=>couldn't find axis.
```

5.8.11 SETLOC

SETLOC uses the catalog header to build the values of the position commons in INCLUDE DLOC.INC for use by position finding and axis labeling routines (at least).

```
SETLOC (DEPTH, SWAPOK)
Inputs:
   DEPTH   I(5)     Position of map plane axes 3 - 7
   SWAPOK  L        T => okay to swap axes if rotation near 90
Common:
   DCAT.INC catalog block (not modified)
   DLOC.INC position parms - filled in here
```

5.8.12 TVFIND

TVFIND determines which of the visible TV images the user wishes to select. If there is more than one visible image, it requires the user to point at it with the cursor. The TV must already be open.

```
TVFIND (MAXPL, TYPE, IPL, UNIQUE, CATBLK, SCRTCH, IERR)
Inputs:
   MAXPL   I        Highest plane number allowed (i.e. do graphics
                    planes count?)
   TYPE    C*2      2-char image type to restrict search
Output:
   IPL     I        Plane number found
   UNIQUE  L        T => only one image visible now
                    (all types except zeroed ones ('ZZ'))
   CATBLK  I(256)   Image catalog block found
   SCRTCH  I(256)   Scratch buffer
   IERR    I        Error code: 0 => ok
                       1 => no image
                       2 => IO error in image catalog
                       3 => TV error
```

5.8.13 UVPGET

UVPGET determines pointers and other information from a UV CATBLK in the common in INCLUDE DCAT.INC. The address relative to the start of a vis record for the real part for a given spectral channel (CHAN) and stokes parameter (ICOR) is given by NRPARM+(CHAN-1)*INCF+ABS(ICOR-ICOR0)*INCS + (IF-1)*INCIF

Single dish data, i.e. randomly sampled data in the image plane, is also recognized and ILOCU and ILOCV point to the longitude like and latitude like random parameters. Also a "BEAM" random parameter may be substitued for the "BASELINE" random parameter. The data type present may be determined from the common variable TYPUVD. Two types of single dish data are recognized:

TYPUVD=1 = unprojected RA and Dec and

TYPUVD=2 = projected RA and Dec (ready for GRIDR)

```
UVPGET (IERR)
Inputs: From common /MAPHDR/ (DCAT.INC)
   CATBLK   I(256)   Catalog block
   CATH     H(256)   same as CATBLK
   CATR     R(256)   same as CATBLK
   CATD     D(128)   same as CATBLK
Output: In common /UVHDR/ (DUVH.INC)
   SOURCE   C*8      Source name.
```

```
ILOCU    I       Offset from beginning of vis record of U
                 or longitude for single dish format data.
ILOCV    I       Offset from beginning of vis record of V
                 or longitude for single dish format data.
ILOCW    I       Offset from beginning of vis record of W.
ILOCT    I              "                       Time
ILOCB    I              "                     Baseline
                                              (or beam)
ILOCSU   I              "                   Source id.
ILOCFQ   I              "                   Freq id.
JLOCC    I       0-rel. order in data of complex values
JLOCS    I       Order in data of Stokes' parameters.
JLOCF    I       Order in data of Frequency.
JLOCR    I       Order in data of RA
JLOCD    I       Order in data of dec.
JLOCIF   I       Order in data of IF.
INCS     I       Increment in data for stokes (see above)
INCF     I       Increment in data for freq. (see above)
INCIF    I       Increment in data for IF.
ICOR0    I       Stokes value of first value.
NRPARM   I       Number of random parameters
LREC     I       Length in values of a vis record.
NVIS     I       Number of visibilities
FREQ     D       Frequency (Hz)
RA       D       Right ascension (1950) deg.
DEC      D       Declination (1950) deg.
NCOR     I       Number of correlators (Stokes' parm.)
ISORT    C*2     Sort order 1st 2 char meaningful.
TYPUVD   I       UV data type, 0=interferometer,
                    1=single dish unprojected,
                    2=single dish projected RA and Dec.
IERR     I       Return error code: 0=>OK,
                    1, 2, 5, 7 : not all normal rand parms
                    2, 3, 6, 7 : not all normal axes
```

5.8.14 XYPIX

XYPIX determines the pixel location corresponding to a specified coordinate value. The pixel location is not necessarily an integer. The position parms are provided by the commons in DLOC.INC which requires a previous call to SETLOC.

```
XYPIX (X, Y, XPIX, YPIX, IERR)
Inputs:
   X      D     X-coordinate value (header units)
   Y      D     Y-coordinate value (header units)
Output:
   XPIX   R     x-coordinate pixel location
   YPIX   R     y-coordinate pixel location
   IERR   I     0 ok, 1 out of range, 2 bad type, 3 undefined
```

5.8.15 XYVAL

XYVAL determines the coordinate value (X,Y,Z) corresponding to the pixel location (XPIX,YPIX). The pixel values need not be integers. The necessary map header data is passed via commons in DLOC.INC requiring a previous call to SETLOC. This program is the inverse of XYPIX.

```
XYVAL (XPIX, YPIX, X, Y, Z, IERR)
Inputs:
   XPIX    R    Pixel location, x-coordinate
   YPIX    R    Pixel location, y-coordinate
Outputs:
   X       D    X-coordinate value at pixel location
   Y       D    Y-coordinate value at pixel location
   Z       D    Z-coordinate value (if part of a position
                pair with either X or Y)
   IERR    I    0 ok, 1 out of range, 2 bad type, 3 undefined
Common inputs:
   DLOC.INC position parms deduced from the map header by
            subroutine SETLOC.
Units are as in the mapheader: degrees for position coords
```

5.8.16 YCINIT

Initialize image catalog for plane IPLANE - TK now done with TKCATL

```
YCINIT (IPLANE, BUFF)
Input:
   IPLANE   I        Image plane to initialize
Output:
   BUFF     I(256)   Working buffer
```

5.8.17 YCOVER

YCOVER checks to see if there are partially replaced images in any of the TV planes currently visible by quadrant.

```
YCOVER (OVER, BUF, IERR)
Outputs:
   OVER   L(4)     T => there are in quadr. I
   BUF    I(512)   scratch
   IERR   I        Error code: 0 => ok, other catlg IO error
```

5.8.18 YCWRIT

Write image catalog block in CATBLK into image catalog.

```
YCWRIT (IPLANE, IMAWIN, CATBLK, BUFF, IERR)
Inputs:
   IPLANE   I        image plane involved
   IMAWIN   I(4)     Corners of image on screen
   CATBLK   I(256)   Image catalog block
 Outputs:
   BUFF     I(256)   working buffer
   IERR     I        error code: 0 => ok
                        1 => no room in catalog
                        2 => IO problems
```

5.8.19 YCREAD

Read image catalog block into CATBLK - TV only (TK in TKCATL).

```
YCREAD (IPLANE, IX, IY, CATBLK ,IERR)
Inputs:
   IPLANE  I       plane containing image whose block is wanted
   IX      I       X pixel coordinate of a point within image
   IY      I       Y pixel coordinate of point within image
Outputs:
   CATBLK  I(256)  Image catalog block
   IERR    I       error codes: 0 => ok
                      1 => IX, IY lies outside image
                      2 => Catalog i/o errors
                      3 => refers to TK device
```

THIS PAGE DELIBERATELY BLANK