Instrument Kernel
IK

March 2006

Purpose

Navigation and Ancillary Information Facility

- The Instrument kernel serves as a repository for instrument specific information that may be useful within the SPICE context.
  - Specifications for an instrument's field-of-view size, shape, and orientation.
  - Internal instrument timing parameters and other data relating to SPICE computations might also be placed in an I-kernel.

- Note: Instrument mounting alignment data are most often specified in a mission frames kernel (FK).

- The IK is a SPICE text kernel.

I-Kernel Structure

- **The format and structure of a typical I-Kernel is:**

```
KPL/IK

   Comments describing the keywords and values
   to follow, as well as any other pertinent
   information.

      \begindata
         Keyword = Value Assignments
      \begintext

   More descriptive comments.

      \begindata
         More Keyword = Value Assignments
      \begintext

   More comments, followed by more data, etc …
```


I-Kernel Contents (1)

- **The requirements on keywords in an IK are the following:**
  - **Keywords must begin with INS[#], where [#] is replaced with the NAIF instrument ID code (which is a negative number).**
  - **The total length of the keyword must be less than 32 characters.**
  - **Keywords are case-sensitive. (Keyword != KEYWORD)**

- **Some examples of IK keywords:**
  - **MGS MOC NA focal length:** **INS-94031_FOCAL_LENGTH**
  - **MEX HRSC SRC pixel angular size:** **INS-41220_IFOV**
  - **MEX ASPERA NPI number of sectors:** **INS-41130_NUMBER_OF_SECTORS**

- **The SPICE toolkit does not require any specific keywords to be present in IK**
  - **One exception is a set of keywords defining an instrument’s FOV if the GETFOV routine is planned to be used to retrieve the FOV attributes**
    - » **Keywords required by GETFOV will be covered later in this tutorial**

I-Kernel Contents (2)

Navigation and Ancillary Information Facility

- IKs usually contain extensive comments including
  - Instrument overview
  - Reference source(s) for the data included in the IK
  - Names/IDs assigned to the instrument and its parts
  - Explanation of each keyword included in the file
  - Description of the FOV and detector layout
  - Sometimes descriptions of the algorithms in which parameters provided in the IK are used, and even fragments of source code implementing these algorithms
    - » For example optical distortion models or timing algorithms
- This documentation exists primarily to assist users in integrating I-Kernel data into their applications
  - One needs to know the keyword name to get its value from the loaded IK data
  - One needs to know what that value means in order to use it

I-Kernel Interface Routines

Navigation and Ancillary Information Facility

- As with any SPICE kernel, IK files are loaded using FURNSH

```
CALL FURNSH ( 'ik_file_name.ti' )
```

- Knowing the name and type (DP, integer, or character) of a keyword of interest, the value of that keyword can be retrieved using G*POOL routines

```
CALL GDPOOL ( NAME, START, ROOM, N, VALUES, FOUND )
CALL GIPOOL ( NAME, START, ROOM, N, VALUES, FOUND )
CALL GCPOOL ( NAME, START, ROOM, N, VALUES, FOUND )
```

- When an instrument's FOV is defined in the IK using a special set of keywords discussed later in this tutorial, the FOV shape, reference frame, boresight vector, and boundary vectors can be retrieved by calling the GETFOV routine

```
CALL GETFOV ( INSTID, ROOM, SHAPE, FRAME, BSIGHT, N, BOUNDS)
```

FOV Definition Keywords (1)

Navigation and Ancillary Information Facility

- **The following keywords defining FOV attributes for the instrument with NAIF ID (#) must be present in the IK for GETFOV to work:**
  - Keyword defining shape of the FOV

```
INS#_FOV_SHAPE      = 'CIRCLE' or 'ELLIPSE' or
                      'RECTANGLE' or 'POLYGON'
```

  - Keyword defining reference frame with respect to which the boresight vector and FOV boundary vectors are specified

```
INS#_FOV_FRAME      = 'frame name'
```

  - Keyword defining the boresight vector

```
INS#_FOV_BORESIGHT = ( X, Y, Z )
```


FOV Definition Keywords (2)

Navigation and Ancillary Information Facility

- Keyword(s) defining FOV boundary vectors, in either of two ways
  - » By specifying boundary vectors explicitly

```
INS#_FOV_CLASS_SPEC       = 'CORNERS' (optional)
INS#_FOV_BOUNDARY_CORNERS = ( X(1), Y(1), Z(1),
                               …     …     …
                              X(n), Y(n), Z(n) )
```

  - » By providing half angular extents of the FOV (possible only for circular, elliptical or rectangular FOVs)

```
INS#_FOV_CLASS_SPEC       = 'ANGLES'
INS#_FOV_REF_VECTOR       = ( X, Y, Z )
INS#_FOV_REF_ANGLE        = halfangle1
INS#_FOV_CROSS_ANGLE      = halfangle2
INS#_FOV_ANGLE_UNITS      = 'DEGREES' or 'RADIANS' or …
```

FOV Definition Keywords (3)

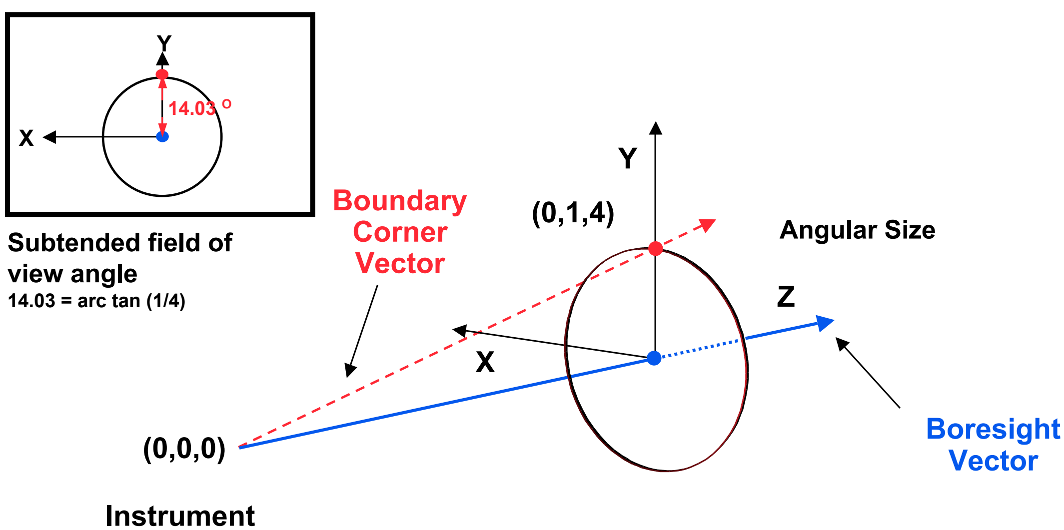
Navigation and Ancillary Information Facility

- **Neither boresight nor reference vectors have to be co-aligned with one of the FOV frame's axes**
  - But for convenience they are frequently defined to be along one of the axes
- **Neither boresight nor corner or reference vectors have to be unit vectors**
  - But they frequently are defined as unit vectors
- **When FOV is specified using the half angular extents method, boresight and reference vectors have to be linearly independent but they don't have to be perpendicular**
  - But for convenience the reference vector is usually picked to be normal to the boresight
- **Half angular extents for a rectangular FOV specify the angles between the boresight and the FOV sides, i.e. they are for the middle of the FOV**

Circular Field of View

Navigation and Ancillary Information Facility

Consider an instrument with a circular field of view.



Circular FOV Definition

The following sets of keywords and values describe this circular field of view:

Specifying boundary vectors explicitly:

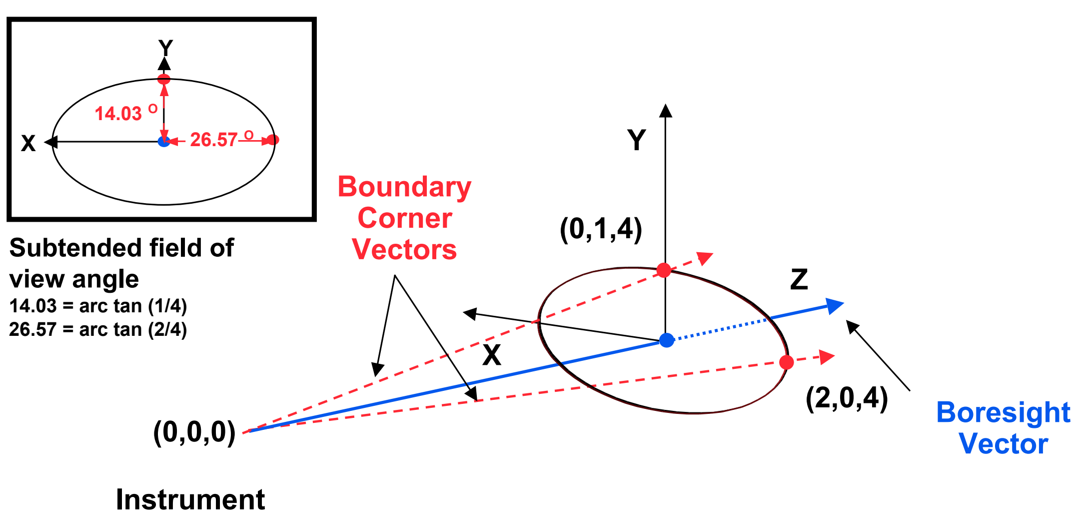
```
INS-11111_FOV_SHAPE            = 'CIRCLE'
INS-11111_FOV_FRAME            = 'FRAME_FOR_INS-11111'
INS-11111_FOV_BORESIGHT        = ( 0.0   0.0   1.0 )
INS-11111_FOV_BOUNDARY_CORNERS = ( 0.0   1.0   4.0 )
```

Specifying half angular extents of the FOV:

```
INS-11111_FOV_SHAPE            = 'CIRCLE'
INS-11111_FOV_FRAME            = 'FRAME_FOR_INS-11111'
INS-11111_FOV_BORESIGHT        = ( 0.0   0.0   1.0 )
INS-11111_FOV_CLASS_SPEC       = 'ANGLES'
INS-11111_FOV_REF_VECTOR       = ( 0.0   1.0   0.0 )
INS-11111_FOV_REF_ANGLE        = 14.03624347
INS-11111_FOV_ANGLE_UNITS      = 'DEGREES'
```


Elliptical Field of View

Consider an instrument with an elliptical field of view.



Elliptical FOV Definition

Navigation and Ancillary Information Facility

The following sets of keywords and values describe this elliptical field of view:

Specifying boundary vectors explicitly:

```
INS-22222_FOV_SHAPE            = 'ELLIPSE'
INS-22222_FOV_FRAME            = 'FRAME_FOR_INS-22222'
INS-22222_FOV_BORESIGHT        = ( 0.0   0.0   1.0 )
INS-22222_FOV_BOUNDARY_CORNERS = ( 0.0   1.0   4.0
                                   2.0   0.0   4.0 )
```

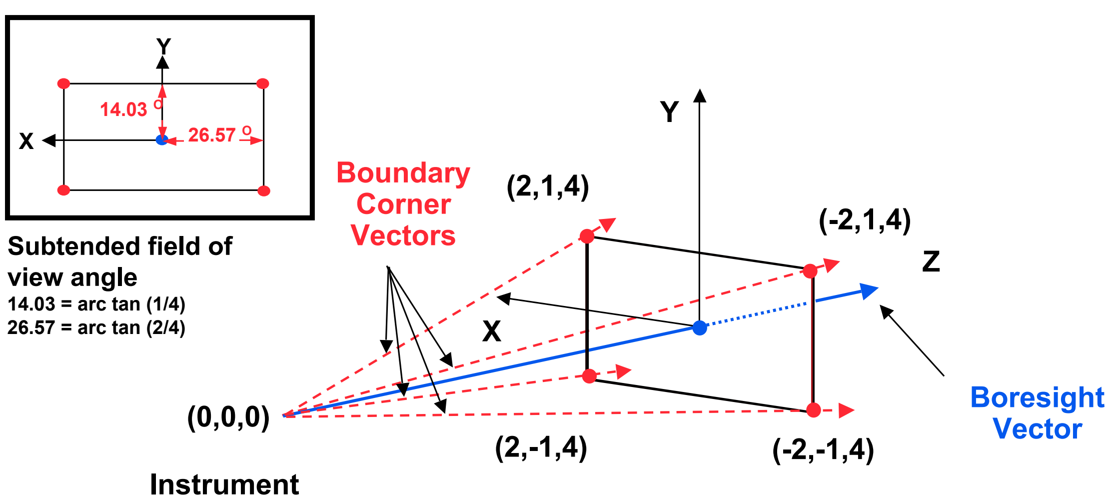
Specifying half angular extents of the FOV:

```
INS-22222_FOV_SHAPE            = 'ELLIPSE'
INS-22222_FOV_FRAME            = 'FRAME_FOR_INS-22222'
INS-22222_FOV_BORESIGHT        = ( 0.0   0.0   1.0 )
INS-22222_FOV_CLASS_SPEC       = 'ANGLES'
INS-22222_FOV_REF_VECTOR       = ( 0.0   1.0   0.0 )
INS-22222_FOV_REF_ANGLE        = 14.03624347
INS-22222_FOV_CROSS_ANGLE      = 26.56505118
INS-22222_FOV_ANGLE_UNITS      = 'DEGREES'
```


Rectangular Field of View

Navigation and Ancillary Information Facility

Consider an instrument with a rectangular field of view.



Rectangular FOV Definition

Navigation and Ancillary Information Facility

The following sets of keywords and values describe this rectangular field of view:

Specifying boundary vectors explicitly:

```
INS-33333_FOV_SHAPE            = 'RECTANGLE'
INS-33333_FOV_FRAME            = 'FRAME_FOR_INS-33333'
INS-33333_FOV_BORESIGHT        = ( 0.0   0.0   1.0 )
INS-33333_FOV_BOUNDARY_CORNERS = ( 2.0   1.0   4.0
                                  -2.0   1.0   4.0
                                  -2.0  -1.0   4.0
                                   2.0  -1.0   4.0 )
```

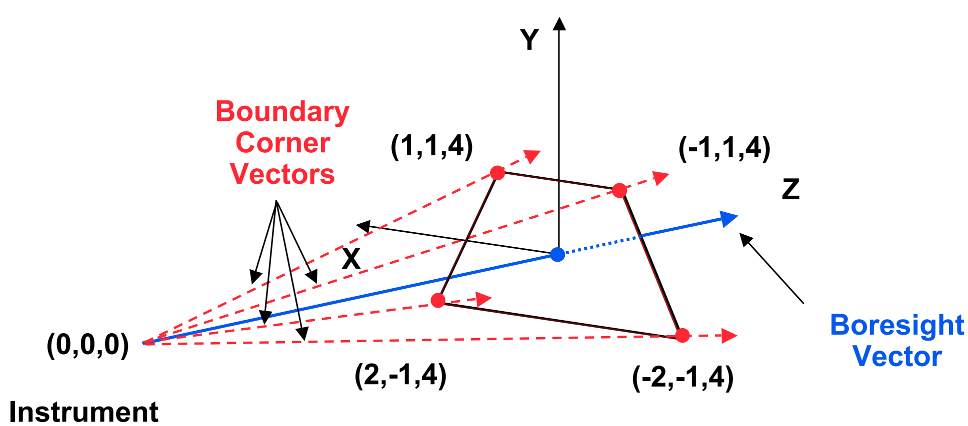
Specifying half angular extents of the FOV:

```
INS-33333_FOV_SHAPE            = 'RECTANGLE'
INS-33333_FOV_FRAME            = 'FRAME_FOR_INS-33333'
INS-33333_FOV_BORESIGHT        = ( 0.0   0.0   1.0 )
INS-33333_FOV_CLASS_SPEC       = 'ANGLES'
INS-33333_FOV_REF_VECTOR       = ( 0.0   1.0   0.0 )
INS-33333_FOV_REF_ANGLE        = 14.03624347
INS-33333_FOV_CROSS_ANGLE      = 26.56505118
INS-33333_FOV_ANGLE_UNITS      = 'DEGREES'
```


Polygonal Fields of View

Navigation and Ancillary Information Facility

Consider an instrument with a trapezoidal field of view.



Polygonal FOV Definition

Navigation and Ancillary Information Facility

The following sets of keywords and values describe this polygonal field of view:

Specifying boundary vectors explicitly:

```
INS-44444_FOV_SHAPE              = 'POLYGON'
INS-44444_FOV_FRAME              = 'FRAME_FOR_INS-44444'
INS-44444_FOV_BORESIGHT          = ( 0.0  0.0  1.0 )
INS-44444_FOV_BOUNDARY_CORNERS = ( 1.0  1.0  4.0
                                  -1.0  1.0  4.0
                                  -2.0 -1.0  4.0
                                   2.0 -1.0  4.0 )
```

Polygonal FOV cannot be specified using half angular extents

Backup

Navigation and Ancillary Information Facility

- **IK file example**
- **Computing angular extents from corner vectors returned by GETFOV**

The following LEMMS1 FOV definition was taken from the Cassini MIMI IK (cas_mimi_v11.ti):

```
Low Energy Magnetospheric Measurements System 1 (LEMMS1)

   Since the MIMI_LEMMS1 detector's FOV is circular and it's diameter is 15.0
   degrees, looking down the X-axis in the CASSINI_MIMI_LEMMS1 frame, we have:
   (Note we are arbitrarily choosing a vector that terminates in the Z=1
   plane.)


                                  ^ Y
                                  |  ins
                                  |
                                  |         /|
                                  |      /   |
                                  |    /     |
                                  |  /     o |
                                  |/  7.50   |
                                  x---------------->
                                X  \          |   Z
                                 ins \        |    ins
                                       \      |
                                         \    |
                                           \  |
                                             \|

                                  |-- 1.0 --|
```

continues

FOV definition from the Cassini MIMI IK (continued):

```
The Y component of one 'boundary corner' vector is:

           Y Component = 1.0 * tan ( 7.50 degrees )
                       = 0.131652498

   The boundary corner vector as displayed below is
   normalized to unit length:
\begindata

 INS-82762_FOV_FRAME   = 'CASSINI_MIMI_LEMMS1'
 INS-82762_FOV_SHAPE   = 'CIRCLE'
 INS-82762_BORESIGHT   = (

  0.0000000000000000   0.0000000000000000  +1.0000000000000000

                      )
 INS-82762_FOV_BOUNDARY_CORNERS = (

  0.0000000000000000  +0.1305261922200500  +0.9914448613738100

                                )

\begintext
```

Circular FOV Angular Size

Navigation and Ancillary Information Facility

The angular separation between the boundary corner vector and the boresight is the angular size.

FORTRAN EXAMPLE

```
C     Retrieve FOV parameters.
      CALL GETFOV(-11111, 1, SHAPE, FRAME, BSGHT, N, BNDS)

C     Compute the angular size.
      ANGSIZ = VSEP( BSGHT, BNDS(1,1) )
```

C EXAMPLE

```
/*  Define the string length parameter. */
    #define STRSIZ         80

/*  Retrieve the field of view parameters. */
    getfov_c(-11111, 1, STRSIZ, STRSIZ, shape, frame,
             bsght, &n, bnds);

/*  Compute the angular separation. */
    angsiz = vsep_c( bsght, &(bnds[0][0]));
```


Elliptical FOV Angular Size - 1

Navigation and Ancillary Information Facility

The angular sizes are the angular separations between the boresight and the boundary vectors.

FORTRAN EXAMPLE

```
C     Retrieve the FOV parameters from the kernel pool.
      CALL GETFOV(-22222, 2, SHAPE, FRAME, BSGHT, N, BNDS)

C     Compute the angular separations.
      ANG1   = VSEP( BSGHT, BNDS(1,1) )
      ANG2   = VSEP( BSGHT, BNDS(1,2) )

C     The angle along the semi-major axis is the larger
C     of the two separations computed.
      LRGANG = MAX( ANG1, ANG2)
      SMLANG = MIN( ANG1, ANG2)
```

Elliptical FOV Angular Size - 2

Navigation and Ancillary Information Facility

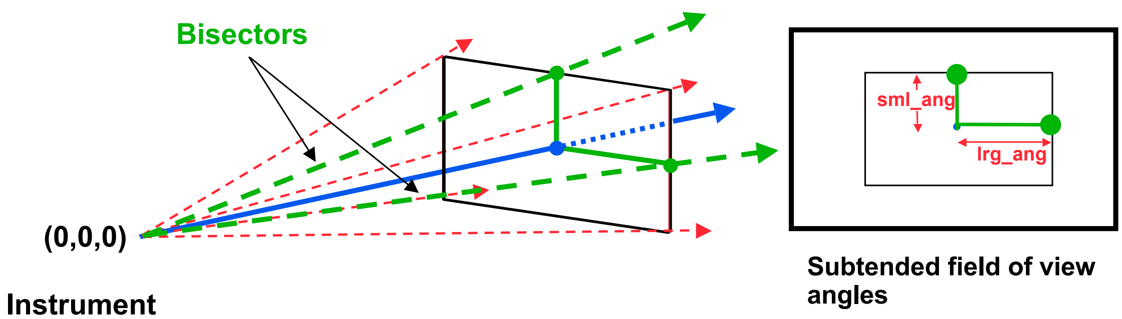
C EXAMPLE

```
/*  Define the string length parameter. */
    #define STRSIZ          80

/*  Retrieve the FOV parameters from the kernel pool. */
    getfov_c(-22222, 2, STRSIZ, STRSIZ, shape, frame,
             bsght, &n, bnds);

/*  Compute the angular separations. */
    ang1 = vsep_c( bsght, &(bnds[0][0]));
    ang2 = vsep_c( bsght, &(bnds[1][0]));

/*  The angle along the semi-major axis is the larger of the
    two separations computed. */
    if ( ang1 > ang2 ) {
       lrgang = ang1; smlang = ang2; }
    else {
       lrgang = ang2; smlang = ang1; }
```


Rectangular FOV Angular Size - 1

Navigation and Ancillary Information Facility

The angular extents of the FOV are computed by calculating the angle between the bisector of adjacent unit boundary vectors and the boresight.



Rectangular FOV Angular Size - 2

FORTRAN EXAMPLE

```
C     Retrieve FOV parameters from the kernel pool.
      CALL GETFOV(-33333, 4, SHAPE, FRAME, BSGHT, N, BNDS)

C     Normalize the 3 boundary vectors
      CALL UNORM(BNDS(1,1), UNTBND(1,1), MAG)
      CALL UNORM(BNDS(1,2), UNTBND(1,2), MAG)
      CALL UNORM(BNDS(1,3), UNTBND(1,3), MAG)

C     Compute the averages.
      CALL VADD(UNTBND(1,1), UNTBND(1,2), VEC1)
      CALL VSCL(0.5, VEC1, VEC1)

      CALL VADD(UNTBND(1,2), UNTBND(1,3), VEC2)
      CALL VSCL(0.5, VEC2, VEC2)

C     Compute the angular separations
      ANG1   = VSEP( BSGHT, VEC1 )
      ANG2   = VSEP( BSGHT, VEC2 )

C     Separate the larger and smaller angles.
      LRGANG = MAX( ANG1, ANG2)
      SMLANG = MIN( ANG1, ANG2)
```


Rectangular FOV Angular Size - 3

C EXAMPLE

```
/*  Define the string length parameter. */
    #define STRSIZ         80

/*  Retrieve the FOV parameters from the kernel pool. */
    getfov_c(-33333, 4, STRSIZ, STRSIZ, shape, frame,
              bsght, &n, bnds);

/*  Normalize the 3 boundary vectors. */
    unorm_c(&(bnds[0][0]), &(untbnd[0][0]), &mag);
    unorm_c(&(bnds[1][0]), &(untbnd[1][0]), &mag);
    unorm_c(&(bnds[2][0]), &(untbnd[2][0]), &mag);

/*  Compute the averages */
    vadd_c(&(untbnd[0][0]), &(untbnd[1][0]), vec1);
    vscl_c(0.5, vec1, vec1);
    vadd_c(&(untbnd[1][0]), &(untbnd[2][0]), vec2);
    vscl_c(0.5, vec2, vec2);

/*  Compute the angular separations. */
    ang1 = vsep_c( bsght, vec1);
    ang2 = vsep_c( bsght, vec2);

/*  Separate the larger and smaller angles. */
    if ( ang1 > ang2 ) {
       lrgang = ang1; smlang = ang2; }
    else {
       lrgang = ang2; smlang = ang1; }
```