esky2det

June 2, 2019

Abstract

Converts sky coordinates to XMM detector coordinates.

1 Instruments/Modes

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOS</td>
<td>imaging</td>
</tr>
<tr>
<td>PN</td>
<td>imaging</td>
</tr>
</tbody>
</table>

2 Use

<table>
<thead>
<tr>
<th>Use</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>pipeline processing</td>
<td>yes</td>
</tr>
<tr>
<td>interactive analysis</td>
<td>yes</td>
</tr>
</tbody>
</table>

3 Description

The task converts sky positions, each specified by an RA/dec pair, to camera-centric coordinates. For the output coordinate system, the user has a choice between DETX/Y or RAWX/Y coordinates. If the latter is specified, the task also calculates and writes the relevant CCDNR. If positional uncertainties are provided by the user, these are also translated into the output coordinate system.

The DETX/Y focal-plane coordinates are the same system as the CAMCOORD2 system described in the Calibration Access and Data Handbook, except that the units of DETX/Y are 0.05 arcseconds rather than the millimetres of CAMCOORD2. RAWX/Y coordinates correspond to the PIXCOORD1 system and thus have units of CCD pixels. The position in this system depends on the readout node, which is specifiable via the parameter mosccdnode. (Note that this parameter is ignored if the instrument is PN, since its CCDs only have the one readout node.) The default setting mosccdnode='primary' reflects the fact that nearly all the XMM EPIC MOS data has been taken with that node setting.

The source positions may either be specified one at a time on the command line (ie, 1 source per invocation of esky2det) or in bulk via a FITS file. These two formats are described in more detail below.
3.1 Command-line position entry.

The sky position of a single source may be entered via the parameters \texttt{ra}, \texttt{dec} and (optionally) \texttt{errorradius}. These are parameters of type angle and can therefore be provided in a variety of formats. Eg:

\begin{verbatim}
ra=32h06m19.6s dec=-07d44m56.7s errorradius=00d00m01.0s
\end{verbatim}

See the task \texttt{param} documentation for further details.

In this mode, the output positions are printed to a single line of standard output. The format of this output line is given in the following examples.

1. Example 1: command line

\begin{verbatim}
esky2det datastyle=user ra=03h47m40.2s dec=24d21m54.62s outunit=raw
\end{verbatim}

gives the output

\begin{verbatim}
# Instrument: EPN
# Coord system of output is RAWXY (PIXCOORD1).
# Source RA = 56.917500 deg.
# Source dec = 24.365171 deg.
# rawX    rawY    ccd  On chip?
32.05  188.69   8   T
\end{verbatim}

2. Example 2: if you want to display the numbers only, add \texttt{withheader='no'} to the command line. In the case above, this would give

\begin{verbatim}
32.05  188.69   8   T
\end{verbatim}

This style is suitable for batch running of \texttt{esky2det} from scripts.

3. Example 3: in this example, the DETX/Y coordinate system is chosen and an uncertainty radius is supplied. The command line example is

\begin{verbatim}
esky2det datastyle=user ra=03h46m25.3s dec=24d18m30.3s outunit=det
witherrorradius=yes errorradius=00d00m05.0s
\end{verbatim}

which gives the output

\begin{verbatim}
# Instrument: EPN
# Coord system of output is DETXY (CAMCOORD2 but in units of 0.05 arcsec).
# Source RA = 56.605415 deg.
# Source dec = 24.308416 deg.
# detX     detY     det   err
-15043.9 -2111.8   100.0
\end{verbatim}

4. Example 4: The same command, but with \texttt{outunit} set now back to ‘raw’, gives
# Instrument: EPN
# Coord system of output is RAWXY (PIXCOORD1).
# Source RA = 56.605415 deg.
# Source dec = 24.308416 deg.

<table>
<thead>
<tr>
<th>rawX</th>
<th>rawY</th>
<th>ccd</th>
<th>On chip?</th>
<th>X err</th>
<th>Y err</th>
</tr>
</thead>
<tbody>
<tr>
<td>40.11</td>
<td>189.72</td>
<td>12</td>
<td>T</td>
<td>0.455E-01</td>
<td>0.455E-01</td>
</tr>
</tbody>
</table>
For the convenience of script authors, I give below the number fields for all four combinations of `outunit` and `witherrorradius`:

- **outunit='raw', witherrorradius='no':**

  # Instrument: <inst>
  # Coord system of output is RAWXY (PIXCOORD1).
  # Source RA = <ra> deg.
  # Source dec = <dec> deg.
  # RAWX   RAWY  ccd  On chip?
  +xxxx.xx +xxxx.xx xx x
  123456789 123456789 12345678

- **outunit='raw', witherrorradius='yes':**

  # Instrument: <inst>
  # Coord system of output is RAWXY (PIXCOORD1).
  # Source RA = <ra> deg.
  # Source dec = <dec> deg.
  # RAWX   RAWY  ccd  On chip?  X err  Y err
  +xxxx.xx +xxxx.xx xx x  x.xxxE+xx  x.xxxE+xx
  123456789 123456789 123456789 123456789 123456789 123456

- **outunit='det', witherrorradius='no':**

  # Instrument: <inst>
  # Coord system of output is DETXY (CAMCOORD2 but in units of 0.05 arcsec).
  # Source RA = <ra> deg.
  # Source dec = <dec> deg.
  # DETX   DETY
  +xxxxx.x +xxxxx.x
  123456789 12345678

- **outunit='det', witherrorradius='yes':**

  # Instrument: <inst>
  # Coord system of output is DETXY (CAMCOORD2 but in units of 0.05 arcsec).
  # Source RA = <ra> deg.
  # Source dec = <dec> deg.
  # DETX   DETY   DET err
  +xxxxx.x +xxxxx.x xxxxx.x
  123456789 123456789 123456789

If `outunit='det'`, and the source is outside the field of view of the instrument, the task halts with an error. This behaviour may be overridden by specifying `checkfov='no'`. If `outunit='raw'` and the source is not found to lie within the bounds of any CCD, a warning message is issued; however the task still prints a RAWX/Y position, this being the position of the source in the RAWX/Y coordinate system of the nearest CCD. In this case the value of the ‘On chip?’ column is ‘F’ rather than ‘T’.
3.2 FITS file position IO.

If positions are supplied in a FITS table, the task attempts to read sky-position information stored in decimal degrees from columns named RA and DEC in the source list specified by the parameter intab. If witherrorcol='yes', esky2det looks by preference for data of the same format in RA_ERR and DEC_ERR columns; if neither column is found, esky2det looks for a common value in arcseconds stored in a column named RADEC_ERR. This last is to cater for the format of source lists output by emldetect.

There are two options for the output: either the output positions can be written to columns in the same file, or a new file can be created to contain them. If the former is desired, set withouttab='no'; if the latter, set withouttab='yes' and provide the name of the new dataset and table in outtab. The transformed position values are written to the appropriate subset of columns RAWX, RAWY, CCDNR, RAWX_ERR, RAWY_ERR, DETX, DETY and DET_ERR (see section 7 for a detailed description of the output format).

If outunit='det', null values are written (with a warning message) for sources which are outside the field of view.

If outunit='raw', a FLAG column is also written. The value of FLAG is set to 0 unless a source does not fall on any CCD, in which case FLAG is set to 1. In these cases the values of RAWX, RAWY and CCDNR represent positions relative to the nearest CCD.

3.3 Uncertainties.

If uncertainties are not requested, the uncertainty fields or columns are not included in the output.

Note that a single uncertainty only is provided for DETX/Y output, whereas separate X and Y uncertainties appear for RAWX/Y. There is no particular need for separate RAWX and RAWY uncertainties while the task is restricted to EPIC application, since pixels on the EPIC cameras are square (except for a small number at the PN chip edges), but an extension to RGS would require separate uncertainty values: hence their retention.

It was decided not to allow the user to specify separate RA and Dec uncertainties because the resulting uncertainty ellipse, although orthogonal in the RA/Dec coordinate system, is no longer necessarily so in a camera-centric system. The possible gains were not considered worth the added complication of specifying a rotated ellipse.

3.4 Task requirements.

The task clearly needs to know the direction in space of the focal axis of the relevant instrument at the time of the observation of interest. There are three necessary pieces of information: the name of the instrument, the date-time of the observation and the spacecraft pointing direction (attitude) at this time. The task processes these as follows: the instrument identification and the observation date-time are used together to calculate (from values stored in the CCF) the instrument boresight correction at the time of the observation; this correction is then applied to the spacecraft pointing.

There are two ways to pass the `cal` information to esky2det, governed by the parameter calinfostyle:

1. calinfostyle = 'set': the information is sought from INSTRUME, DATE-OBS etc keywords in the header of the source list specified by calinfoset. These should be found in the primary headers of all XMM product data sets. It is recommended to use the source list itself as the calinfoset, provided it is an XMM product, and from the same instrument.
2. `calinfostyle = 'user'`: the information must be provided by the user via the parameters `instrument`, `datetime`, `scattra`, `scattdec` and `scattapos`.

In order to be able to apply the correct boresight correction, as well as to have access to the instrument specifications, the task requires access to the CCF components relevant to the dates of observation and analysis. This is achieved in the usual way by constructing a cif file with `cifbuild` and pointing to it with the environment variable `SAS_CCF`.

### 4 Parameters

This section documents the parameters recognized by this task (if any).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Mand</th>
<th>Type</th>
<th>Default</th>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>datastyle</td>
<td>no</td>
<td>string</td>
<td>set</td>
<td>user—set</td>
</tr>
</tbody>
</table>

By use of this parameter, the user can specify the style of input and output desired. If `datastyle='user'`, the task looks for position information for a single source in the parameters `ra` and `dec`. If `datastyle='set'`, the task looks for the source positions stored in columns `RA` and `DEC` of a FITS table specified by the parameter `intab`.

<table>
<thead>
<tr>
<th>intab</th>
<th>no</th>
<th>table</th>
<th>inset.ds:INPUT</th>
</tr>
</thead>
</table>

Name of the FITS table that contains the sky positions. Active only if `datastyle='set'`.

<table>
<thead>
<tr>
<th>witherrorcol</th>
<th>no</th>
<th>boolean</th>
<th>no</th>
</tr>
</thead>
</table>

Active only if `datastyle='set'`. If `witherrorcol='yes'`, `esky2det` looks for position uncertainties in decimal degrees in columns `RA_ERR` and `DEC_ERR` of `intab`. If neither of these columns is found, `esky2det` looks for a common value in arcseconds stored in a column named `RADEC_ERR`. This last is to cater for the format of source lists output by `emldetect`.

<table>
<thead>
<tr>
<th>withouttab</th>
<th>no</th>
<th>boolean</th>
<th>no</th>
</tr>
</thead>
</table>

Active only if `datastyle='set'`. If `withouttab='yes'`, the task looks for `outtab` and writes the output to this file. Otherwise the output is written to columns in the `intab`.

<table>
<thead>
<tr>
<th>outtab</th>
<th>no</th>
<th>table</th>
<th>outset.ds:OUTPUT</th>
</tr>
</thead>
</table>

Name of the FITS table that is to contain the camera-centric positions. Active only if `datastyle='set'` and `withouttab='yes'`.

<table>
<thead>
<tr>
<th>ra</th>
<th>yes</th>
<th>angle</th>
<th>0.0</th>
<th>0 ≤ ra ≤ 360</th>
</tr>
</thead>
</table>

The right ascension (in any of the allowed angle-parameter formats) of the source. Active only if `datastyle='user'`.

<table>
<thead>
<tr>
<th>dec</th>
<th>yes</th>
<th>angle</th>
<th>0.0</th>
<th>−90 ≤ dec ≤ 90</th>
</tr>
</thead>
</table>

The declination (in any of the allowed angle-parameter formats) of the source. Active only if `datastyle='user'`.

<table>
<thead>
<tr>
<th>witherrorradius</th>
<th>no</th>
<th>boolean</th>
<th>no</th>
</tr>
</thead>
</table>

Active only if `datastyle='user'`. If `witherrorradius='yes'`, `esky2det` reads a single angular uncertainty from parameter `errorradius`.

<table>
<thead>
<tr>
<th>errorradius</th>
<th>yes</th>
<th>angle</th>
<th>0.0</th>
<th>0 ≤ errorradius</th>
</tr>
</thead>
</table>

The radius of the uncertainty circle around the source position. Active only if `datastyle='user'` and `witherrorradius='yes'`. 
### withheader

<table>
<thead>
<tr>
<th></th>
<th>no</th>
<th>boolean</th>
<th>yes</th>
</tr>
</thead>
</table>

Active only if `datastyle='user'`. If `withheader='yes'`, the task prints some lines of header information before printing the source position line; if ‘no’, this is omitted. The ‘no’ setting is convenient for those wishing to run `esky2det` from a script.

### outunit

<table>
<thead>
<tr>
<th></th>
<th>no</th>
<th>string</th>
<th>raw</th>
<th>det—raw</th>
</tr>
</thead>
</table>

If `outunit='det'`, the positions are calculated in the DETX/Y system. Otherwise, the RAWX/Y and CCDNR of the position are calculated.

### mosccdnode

<table>
<thead>
<tr>
<th></th>
<th>no</th>
<th>string</th>
<th>primary</th>
<th>primary—redundant</th>
</tr>
</thead>
</table>

This allows the user to specify the readout node for positions on the MOS instruments. It is enabled only if `outunit='raw'`, and ignored for PN.

### calinfostyle

<table>
<thead>
<tr>
<th></th>
<th>no</th>
<th>string</th>
<th>set</th>
<th>set—user</th>
</tr>
</thead>
</table>

If ‘set’ the task obtains information about the instrument and spacecraft pointing from `calinfoiset`; if ‘user’, this information is obtained from parameters `instrument`, `datetime`, `scattra`, `scattdec` and `scattapos`.

### calinfoiset

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>dataset</th>
<th>calinfo.ds</th>
</tr>
</thead>
</table>

The name of the dataset in which information about the instrument and spacecraft pointing etc is stored in keywords.

### instrument

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>string</th>
<th>EMOS1</th>
<th>EMOS1—EMOS2—EPN</th>
</tr>
</thead>
</table>

Active only if `calinfostyle='user'`. The name of the relevant XMM instrument.

### datetime

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>string</th>
<th>0000-00-00T00:00:00</th>
</tr>
</thead>
</table>

If `calinfostyle='user'`, the date and time are expected via this string parameter.

### scattra

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>angle</th>
<th>0.0</th>
<th>$0 \leq \text{attra} \leq 360$</th>
</tr>
</thead>
</table>

The right ascension (in any of the allowed angle-parameter formats) of the spacecraft pointing. Active only if `calinfostyle='user'`.

### scattdec

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>angle</th>
<th>0.0</th>
<th>$-90 \leq \text{attdec} \leq 90$</th>
</tr>
</thead>
</table>

The declination (in any of the allowed angle-parameter formats) of the spacecraft pointing. Active only if `calinfostyle='user'`.

### scattapos

<table>
<thead>
<tr>
<th></th>
<th>yes</th>
<th>angle</th>
<th>0.0</th>
<th>$0 \leq \text{attapos} \leq 360$</th>
</tr>
</thead>
</table>

The position angle (in any of the allowed angle-parameter formats) of the spacecraft pointing. Active only if `calinfostyle='user'`.

### checkfov

<table>
<thead>
<tr>
<th></th>
<th>no</th>
<th>boolean</th>
<th>yes</th>
</tr>
</thead>
</table>

If set true then the task does not return detector coordinates for positions which don’t lie on a CCD or within the field-of-view. Set to false to get a DET value for these cases.

## 5 Errors

This section documents warnings and errors generated by this task (if any). Note that warnings and errors can also be generated in the SAS infrastructure libraries, in which case they would not be documented here. Refer to the index of all errors and warnings available in the HTML version of the SAS documentation.
badCalInfoStyle (error)  
The value of the calinfostyle parameter was not recognised.

badNodeStr (error)  
The value of the mosccdnode parameter was not recognised.

badOutUnit (error)  
The value of the outunit parameter was not recognised.

badDataStyle (error)  
The value of the datastyle parameter was not recognised.

emptyTableName (error)  
Parameter outtab was not given in the form dataSetName:tableName.

notInFov (error)  
The calculated det XY position does not fall within the field of view.

notOnChip (warning)  
The calculated raw XY position does not fall on any ccd

corrective action: The task (i) calculates the RAWX/Y position relative to the nearest chip;  
(ii) either sets the FLAG column to 1 or writes F under ‘On chip?’ in the output, depending
on the value of datastyle.

6 Input Files

1. Required only if datastyle='set': a source list in the form of a FITS table intab containing
the following columns:
   • RA, in decimal degrees.
   • DEC, in decimal degrees.
   • If witherrorcol='yes', either
     – columns RA_ERR and DEC_ERR, in decimal degrees, or:
     – RADEC_ERR, in arcseconds.

   All data types may be either REAL32 or REAL64.

2. Required only if calinfostyle='set': an XMM product file containing keywords INSTRUME,
DATE-OBS, RA_PNT, DEC_PNT and PA_PNT.
7 Output Files

Written only if datastyle='set': a FITS table containing the following columns:

- If outunit='raw':
  - A 32-bit real column RAWX, in units of CCD pixels.
  - A 32-bit real column RAWY, in units of CCD pixels.
  - An 8-bit integer column CCDNR
  - A 32-bit integer column FLAG
  - If witherrorcol='yes', 32-bit real columns RAWY_ERR and RAWY_ERR, in units of CCD pixels.

- If outunit='det':
  - A 32-bit real column DETX, in 0.05 arcsec.
  - A 32-bit real column DETY, in 0.05 arcsec.
  - If witherrorcol='yes', a 32-bit real column DET_ERR, in 0.05 arcsec.

If withouttab='yes', a new dataset outtab is constructed to hold these columns; if 'no', they are written to the intab.

8 Algorithm

Fairly obvious - mostly implemented using the appropriate cal calls.

9 Comments

References