raygen

Diab Jerius

Table of Contents

1 Copying

The software described by this manual is copyright © 2006 Smithsonian Institution. All rights reserved.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

2 Introduction

`raygen` generates rays from one or more user specified sources, filling one or more user specified entrance apertures. The output rays are in the `bpipe` format.

The source specification is done via scripts written in the `lua` language. `raygen` provides a basic set of angular generators (image, point, rect) which may be combined with spectrum generators to create fairly complicated sources. The use of an embedded scripting language to combine these generators provides a high degree of flexibility.

The area to be illuminated (i.e. the telescope's entrance aperture) is constructed from a set of primitive geometric shapes (circles, annuli, annular wedges) which are combined via a script (again, written in `lua`).

3 Parameters

`raygen` uses an IRAF compatible parameter file.

It takes the following parameters

source

> The `lua` script which specifies the sources.

source_override

> `lua` statements to be passed to the source script. These are placed in a function called `override`, which may be called by the source script.

ea

> A `lua` script which defines the entrance aperture to illuminate.

ea_override

> `lua` statements to be passed to the entrance_aperture script. These are placed in a function called `override`, which may be called by the script.

output

> The output file. If it is the string '`stdout`', the rays are written to the UNIX standard output stream.

logfile

> A file to which logging output should be written. If it is the string '`stderr`' the rays are written to the UNIX standard error stream.

limit_type

> This determines how `raygen` knows when to stop outputting rays. It determines what the `limit` parameter specifies. It can one of the following values:
>
> > '`rays`'
> >
> > > the number of rays
> >
> > '`krays`'
> >
> > > the number of rays, in units of one thousand rays
> >
> > '`Mrays`'
> >
> > > the number of rays, in units of one million rays
> >
> > '`sec`'
> >
> > > the time to run, in seconds
> >
> > '`ksec`'
> >
> > > the time to run, in units of one thousand seconds
> >
> > '`r/cm2`'
> >
> > > rays per square centimeter at the entrance aperture
> >
> > '`r/mm2`'
> >
> > > rays per square millimeter at the entrance aperture

limit

   The quantity of whatever `limit_type` specifies that `raygen` must generate.

node

   The node Z position (mm) from which to measure the off-axis source angle
   (I<theta>).

seed1

   The first seed for the random number generator. It must be in the range
   [1,2147483562].

seed2

   The second seed for the random number generator. It must be in the range
   [1,214748339]

block

   The random number block to start at. It must be in the range [0,1048575].

ray_dist

   How the rays should be distributed at the entrance aperture. Currently, only
   '`random`' is supported.

debug

   A list of debug flags; none are presently available.

version

   Output the version to the UNIX standard error stream and exit.

help

   Output a help message and exit.

4 Coordinate Systems

Source sky positions are specified in cylindrical coordinates, with the Z axis along the optical axis. The angular distance from the optical axis is described by *theta*, the azimuth about the axis by *phi*, where $phi = 0$ is along the +X axis. The distance to the source is actually specified by its Z coordinate, which is generally zero at the telescope's node, except in the case of the AXAF HRMA, where it is zero at CAP datum A. (The ray generator actually doesn't set the zero point; it just cares about where the node is.)

5 Units

Angular and linear quantities may be specified in a variety of units. Linear quantities
without unit specifications are assumed to be in millimeters. Angular quantities without
unit specifications are assumed to be in radians. To specify a unit, pass the entire quantity
as a string, appending the unit to the value. For example,

```
theta = 2.0    --> radians
theta = '2.0 deg' --> degrees
```

The available angular units are 'deg', 'degree', 'arcmin', 'arcminute', 'arcsec',
'arcsecond', 'milliarcsecond', 'rad', 'radian', 'milliradian', 'mrad', 'microradian',
'urad'.

The available linear units are 'parsec', 'm', 'meter', 'decimeter', 'dm', 'centimeter',
'cm', 'millimeter', 'mm', 'micrometer', 'micron', 'um', 'nanometer', 'nm', 'Angstrom',
'angstrom', 'A'.

6 Source specification

Currently, sources are specified as combinations of *position* and *spectrum* generators. There may be multiple spectrum generators associated with each position generator. There is only one position generator per source.

You must write a `lua` function entitled `source` which creates the sources you wish to model. Each source is begun with the `begin_source` function, and ended with the `end_source` function. Both take a single parameter, namely a string specifying the name of the source, which is used to identify it in any output:

```
function source()

  begin_source( 'M87 - knot1' )
    mono( 'knot1 spectrum', 2.5, 1 )      -- the spectrum generator
    point( 'knot1 point' )                -- the position generator
  end_source( 'M87 - knot1' )

end
```

Note that the identifying string given to `end_source` must be identical to that passed to `begin_source`.

The file containing your source script is what is given to the `raygen source` parameter. The source override code provided by the `source_override` parameter is placed in a function called `override`. You should place a call to `override` early in your script (it can be done outside of the `source` function).

The provided position generators are described in the following sections.

6.1 Position Generators

Position generators define the angular extent and position of a source. Each function has both mandatory and optional arguments. Optional arguments are usually set to a reasonable default. Mandatory arguments are explicitly included in the argument list; optional arguments are passed via a `lua` table (an associative array, or hash in Perl lingo). For example, here are two equivalent methods of creating a rectangular source at a specific off-axis location:

```
optargs = { }
optargs.theta = '2 arcmin'
optargs.phi = '3 arcmin'
optargs.z = '-2 parsec'
rect( 'square in the sky', '1 arcsec', '1 arcsec', optargs )

rect( 'square in the sky', '1 arcsec', '1 arcsec',
      {
          theta = '2 arcmin',
          phi = '3 arcmin',
```

```
            z = '-2 parsec'
        } )
```

Note that z is *negative*; the Z coordinate increases *towards* the focal plane.

6.1.1 Intensity Distributions

Several intensity distribution models are available (well, two). The parameters for these are usually passed as a `Lua` table in the `optargs` optional arguments parameter.

The intensity distribution may be specified independently in the X and Y directions, or may be specified as a common distribution. If no distribution is specified it will be uniform. Common parameters are provided via the the `dist` table, while parameters specific to an axis are provided via the `x` and `y` tables. This code

```
disk( "disk", '40 arcminutes',
      {
        dist = {
                type = 'gaussian',
                sigma = '3 arcsec'
              },
        x = { center = '1 arcsec' },
        y = { center = '1 arcsec' }
      }  )
```

sets up a disk source with a radially symmetric Gaussian offset from the source center. Note that the `center` parameter is only valid in the axis specific tables. To specify independent distributions, don't use the `dist` parameter:

```
disk( "disk", '40 arcminutes',
      {
        x = {
                type = 'gaussian',
                center = '1 arcsec',
                sigma = '3 arcsec'
            },
        y = {
                type = uniform'
            },
      }  )
```

The following distributions are available. The type of distribution is specified by the `type` parameter, as shown in the above examples.

`uniform`

>   This provides a constant intensity across the source. It has no parameters.

gaussian

> This is a Gaussian distribution. The center is nominally at the center of the source, but may be changed. The width of the Gaussian is specified by one of the following two parameters:
>
> fwhm
>
>> The Full Width Half Maximum of the Gaussian.
>
> stddev
>
>> The standard deviation of the Gaussian.
>
> The center of the Gaussian is specified via the `center` parameter, which must be specified independently for each axis.

6.1.2 rect(name, width, height [, optargs])

A rectangle with the given angular *width* and *height* dimensions. *name* is a string used for identification purposes. *optargs* is a `lua` table containing optional parameters.

The position of the source is given by these parameters:

theta

> The distance from the optical axis to the center of the rectangle, in radians. If not specified, it defaults to '0'.

phi

> The azimuth about the optical axis, in radians. If not specified, it defaults to '0'.

z

> The Z coordinate of the source. This defaults to an infinitely distant source.

For instructions on specifying the source distribution, Section 6.1.1 [Intensity Distributions], page 12.

6.1.3 disk(name, radius [, optargs])

A circular disk with the given angular *radius*. *name* is a string used for identification purposes. *optargs* is a `lua` table containing optional parameters.

The position of the source is given by these parameters:

theta

> The distance from the optical axis to the center of the rectangle, in radians. If not specified, it defaults to '0'.

phi

> The azimuth about the optical axis, in radians. If not specified, it defaults to '0'.

z

> The Z coordinate of the source. This defaults to an infinitely distant source.

For instructions on specifying the source distribution,

6.1.4 image(name [, optargs])

This specifies that the angular distribution of the source is derived from an image, either read from file or one previously created (the latter is currently not supported). *name* is a string used for identification purposes. Note that the 'file' optional argument is for now not optional; it *must* be specified.

The following optional arguments are recognized:

file

> The name of the image file

format

> The format of the file. If not specified, B<raygen> will attempt to guess at the format. It may be one of the following:
>
> 'ascii'
>
> > The pixel values are in ASCII. Values are separated by whitespace (including newlines). The file does not contain any information about the pixel size or dimensions; you must specify these using the `pixsz`, `pixsz_x`, `pixsz_y`, `nx`, `ny` arguments.
>
> 'double'
>
> > The pixel values are stored as binary double precision floating point numbers. The file does not contain any information about the pixel size or dimensions; you must specify these using the `pixsz`, `pixsz_x`, `pixsz_y`, `nx`, `ny` arguments.
>
> 'float'
>
> > The pixel values are stored as binary single precision floating point numbers. The file does not contain any information about the pixel size or dimensions; you must specify these using the `pixsz`, `pixsz_x`, `pixsz_y`, `nx`, `ny` arguments.
>
> 'fits'
>
> > The image is a standard FITS image. The pixel size may be specified by the 'CDELT'n FITS keywords, or by the optional `pixsz`, `pixsz_x`, `pixsz_y` arguments. If they are not specified, the default is 0.5 seconds of arc. If the pixel size is specified in the FITS file, the units for the size may be specified with the 'CUNIT'n FITS keywords. If not, degrees are assumed. The FITS keywords 'CRPIX'n and 'CRVAL'n are supported.
> >
> > If the FITS image has a coordinate with a 'CTYPE' whose value begins with 'RA', it is taken to be in the standard Astronomical Right Ascension coordinate system, and the X axis `flip` flag is

turned on automatically (as in that system the X pixel increment is negative, which leads to incorrectly flipped images otherwise).

'pbm'

The image is in the Portable Bitmap format. The dimensions of the image are encoded in the file. The user must specify the size of the pixels using the optional `pixsz`, `pixsz_x`, `pixsz_y` arguments

'pgm'

The image is in the Portable Grayscale Map format. The dimensions of the image are encoded in the file. The user must specify the size of the pixels using the optional `pixsz`, `pixsz_x`, `pixsz_y` arguments

`nx`
`ny`

The dimensions of the image. Depending upon the image format, this may be specified in the image file.

`pixsz`
`pixsz_x`
`pixsz_y`

The size of the pixels. Pixels need not be square. Depending upon the image format, this may be specified in the image file.

`theta`

The distance from the optical axis to the center of the rectangle, in radians. If not specified, it defaults to '0'.

`phi`

The azimuth about the optical axis, in radians. If not specified, it defaults to '0'.

`z`

The Z coordinate of the source. It defaults to an infinitely distant source.

`clip`

The minimum value of pixel to raytrace. Pixels with values less than this are ignored.

`flip`

This takes one of the following values:

`x`

The image is flipped along the X axis.

`y`

The image is flipped along the Y axis.

`xy`

The image is flipped along both the X and Y axes.

```
point_idx_x
point_idx_y
```
>   The pixel index (possibly fractional) of the position in the image which corre-
>   sponds to the angular source position specified by `theta` and `phi`. You may
>   also specify this indirectly via `point_wcs_x`/`point_wcs_y` and `ref_idx_x`/`ref_`
>   `idx_y` and `ref_wcs_x`/`ref_wcs_y`.

```
point_wcs_x
point_wcs_y
```
>   The WCS position which corresponds to the angular source position specified
>   by `theta` and `phi`. This is used in conjunction with `ref_idx_x`/`ref_idx_y` and
>   `ref_wcs_x`/`ref_wcs_y`.

```
ref_wcs_x
ref_wcs_y
```
>   The WCS position of a reference point in the image. This is used in conjunction
>   with `ref_idx_x`/`ref_idx_y` and `point_wcs_x`/`point_wcs_y`.

```
ref_idx_x
ref_idx_y
```
>   The pixel index (possibly fractional) of the reference point in the image. This
>   is used in conjunction with `ref_wcs_x`/`ref_wcs_y` and `point_wcs_x`/`point_`
>   `wcs_y`.

6.1.5 point(name [, optargs])

A point source. *name* is a string used for identification purposes. *optargs* is a `lua` table
containing optional parameters. The following optional arguments are recognized:

theta

>   The distance from the optical axis to the point, in radians. If not specified, it
>   defaults to '0'.

phi

>   The azimuth about the optical axis, in radians. If not specified, it defaults to
>   '0'.

z

>   The Z coordinate of the source. It defaults to an infinitely distant source.

6.2 Spectrum Generators

The spectrum generators are:

6.2.1 mono(name, energy, flux)

This models a monoenergetic spectrum. *name* is a string used for identification purposes.
energy is the energy in keV. Flux is in photons/s/cm^2 at the telescope.

6.2.2 flat(name, energy_min, energy_max, flux)

This models a flat photon spectrum (i.e, constant number of photons as a function of energy). *name* is a string used for identification purposes. *energy_min* and *energy_max* specify the energy range, in keV. The bounds are *not* included in the range. Flux is in photons/s/cm^2 at the telescope.

6.2.3 spectrum(name [, optargs])

This function specifies an arbitrary, binned spectrum to simulate. *name* is a string used for identification purposes. Note that the 'file' optional argument is for now not optional; it *must* be specified.

The following optional arguments are recognized:

file

> The name of the file containing the spectrum.

format

> The format of the file. It defaults to 'rdb'. It may be one of:

> 'rdb'

> > The file is in an rdb table. There must be at least three columns, which denote the limits of the bins and the flux within each bin. The names of the columns may be specified by the emin, emax, and flux optional arguments.

emin

> The name of the column containing the minimum edge of a bin, in keV, for those formats for which it is appropriate. It defaults to 'emin'.

emax

> The name of the column containing the maximum edge of a bin, in keV, for those formats for which it is appropriate. It defaults to 'emax'.

flux

> The name of the column containing the flux (for those formats for which it is appropriate). It defaults to 'flux'.

units

> The units in which the spectrum is specified. It may be one of 'ergs/s/cm2/kev', 'kev/s/cm2/kev', or 'photons/s/cm2'.

scale

> An arbitrary scale factor to apply to the spectrum. It defaults to '1.0'.

Variable and Parameter index

U

V

X

Y

Z