NAME

transform_ray - rotate and translate rays

PARAMETERS

transform_ray uses an IRAF compatible parameter interface

input *file*

Input ray stream. If the filename is the string `stdin`, it reads from the standard input stream.

output *file*

Output ray stream. If the filename is the string `stdout`, it writes to the standard output stream.

coeff

The transformation coefficients. See *Specifying Coefficients* for more information.

x *double*

y *double*

z *double*

Coordinates of the optional translation

translate `before` | `after` | `none`

What order to apply the translation relative to the transformation specified by the **coeff** parameter. If `none`, no translation is performed.

version *boolean*

Print out **transform_ray**'s version and exit.

help *boolean*

Print out help and exit.

debug *list*

A list of debug flags. The available flags are:

`xfrm`

Output the final rotation and translation matrices to the standard error stream. The rotation is applied before the translation.

DESCRIPTION

transform_ray performs translation and rotation of rays (their position, direction, and polarization). It reads and writes rays in **bpipe** format.

Transformations may be supplied as a combination of an optional translation with any of the following:

- a quaternion
- rotation about an axis
- a 3x3 rotation matrix
- a 4x4 affine transformation matrix

Transformations are done by left-multiplying vectors by matrices. The translation may be applied before or after the above transformation. See the `translate` parameter for more information.

Specifying Coefficients

The **coeff** parameter accepts a number of different transformation coefficient specifications.

* Quaternions

  A quaternion is represented as four comma separated floating point numbers in the order *w*, *x*, *y*, *z*. White space is ignored.

  For example, a rotation about the *X* axis by 90 degrees would be specified as

```
coeff="0.70710678, 0.70710678, 0, 0"
```

* Rotations about an axis

  Rotations about arbitrary axes are represented by four comma separated floating point numbers in the order *angle* (in radians), *x*, *y*, *z*. The angle must be prefixed with a @ character. White space is ignored.

  For example, a rotation about the *X* axis by 90 degrees would be specified as

```
coeff="@1.5707963267949, 1, 0, 0"
```

* 3X3 Rotation Matrix

  A 3X3 rotation matrix is represented in row-major order by nine floating point numbers. Numbers may be separated by commas or spaces. The characters "}{()}" may be used to group numbers for convenience.

  For example, to represent a rotation about *X* by 90 degrees, the following matrix

```
1  0  0
0  0 -1
0  1  0
```

  may be specified as

```
coeff="(1, 0, 0), (0, 0, -1), ( 0, 1, 0 )"
```

  Recall that transformations are performed by left-multiplying vectors by matrices.

* 4X4 Affine Transformation Matrix

  An affine transformation matrix is represented in row-major order by sixteen floating point numbers. Numbers may be separated by commas or spaces. The characters "}{()}" may be used to group numbers for convenience.

  Recall that transformations are performed by left-multiplying vectors by matrices. The translation elements in the affine matrix should be in the rightmost column.

  For example, to represent a rotation about *X* by 90 degrees followed by a translation by *(3,4,5)* the following matrix

```
1  0  0 3
0  0 -1 4
0  1  0 5
0  0  0 1
```

  may be specified as

```
coeff="( 1, 0, 0, 3) ( 0, 0, -1, 4 ) (0, 1, 0, 5) ( 0, 0, 0, 1)"
```

COPYRIGHT & LICENSE

AUTHOR

T. Gaetz

D. Jerius