User Manual 4.15 Fields of view

Introduction

Scope

The fields of view are used to describe the instruments and sensors of a spacecraft : each of them has several fields of view and inhibition (the inhibition fields are also created as "fields of view" : they provide the same services). All the available fields are described thereafter.

Javadoc

All the fields of view and the associated interface are available in the package `fr.cnes.sirius.patrius.fieldsofview`

Library	Javadoc
Basis	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/package-summary.html Package fr.cnes.sirius.patrius.fieldsofview]

Package Overview

All fields implement the IFieldOfView interface.

Features Description

Omnidirectional field

This field is all the 3D space : any direction vector belongs to it. The angular distance to a vector has no sense here : this method always return 1 (the convention for a vector in the field being to return a positive value).

Circular field

This field is a right circular infinite cone defined by an angular aperture from a given direction.

Elliptic field

This field is an infinite elliptic cone defined by two an angular aperture from a given origin. (Cf [EllipticField](#)).

Nota : Angular distances are approximation. Moreover, if the user point is outside the cone and the closest point on the cone is its apex, then the angular distance return is $-\pi \over 2$).

Rectangle field

This field is a rectangle infinite cone defined the same way as the identical object of the geometry section : a main direction vector, abn approximative U vector (to define the local frame) and two aperture angles (see the Infinite rectangle cone chapter for more details).

Pyramidal field

This field describes any pyramidal field defined by an array of directions. The field must be created with the directions (vectors) in the right order : the inside of the field is on the side of the positive cross vector of two consecutive vetcors i and i+1 of the list (and the last to the first). This field proposes additional methods for field manipulation:

- `getIntersectionWith`: computes intersection with another PyramidalField. Be careful, this method may return an exception in case of strongly degenerated cases (multiple intersections, etc.)
- `getUnionWith`: computes union with another PyramidalField. Be careful, this method may return an exception in case of strongly degenerated cases (multiple unions, etc.)
- `getComplementaryFieldOfView`: returns the complementary field of view
- `isConvex`: says if the field is convex

Boolean field

This field is a boolean combination ("AND" or "OR") of two previously created fields.
A direction is in an "OR" field if it is at least in one of the two fields defining it.
A direction is in an "AND" field if it is in both of the two fields defining it.
Beware : the angular distance computed is approximative in some cases (for a direction in the field in an "OR" case, for a direction out of the field in an "AND" case), but the sign is always the right one, and the function is continuous around its zero (direction on the border of the field), so that it can be used fo events detection.

Invert field

The invert field is created from an existing one : and represent a field with a same border but an inverse definition of "in" and "out". The angular distance is so the opposite of the one of the original field.

Sector field

This field is a min/max longitude and min/max latitude aperture on the unit sphere.
It is defined by three vectors : the local "north pole" vector for this sphere, the min latitude- min longitude point vector (V1) and the max latitude - max longitude point vector (V2).
The angle from the pole vector to V1 must be greater than the angle from the pole vector to V2 (greater latitude). None of them can be aligned with or opposite to the "pole" vector (the field is undefined if V1 or V2 is on a pole).

Azimuth-Elevation Field

This field is defined by an azimuth/elevation array. The border is then described by linear segments between these points. The inside of the field is the sector of the unitary sphere with greaters elevation than the border.
The angular distance to the field limit is not the exact shortest distance: it is the exact angular distance on the local meridian (difference of elevation of the target and the linear interpolated local elevation). Concerning the field of view, the limit between two consecutive points is NOT a great circle (i.e. planar) limit: it is linear in azimuth-elevation. The fields with planar limits are the Rectangle or Pyramidal fields.

Azimuth and elevation follow the following convention:

For a given point P, defined by cartesian coordinates in frame (XYZ):

- This frame is such that x and y axis are in the same plane as reference azimut AzimutRef and such that the angle between reference azimut and x axis is defined by alpha angle in counter-clockwise (trigonometric) direction.
- If alpha = 0, x is along reference azimut (in the case of TopoCentricFrame points along North in general), y is along West direction
- On the above scheme, alpha = -30°, the x axis is along North-East and y axis along North-West
- Azimut of point P is equal to beta angle, counted in clockwise direction, starting from reference azimut. It is equal to 324° on the above scheme.

Getting Started

Modèle:SpecialInclusion prefix=$theme sub section="GettingStarted"/

Contents

Interfaces

Interface	Summary	Javadoc
IFieldOfView	Interface for all fields of view.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/IFieldOfView.html ...]
IGeometricFieldOfView	Interface for all fields of view with added geometric functionalities.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/IGeometricFieldOfView.html ...]

Classes

Class	Summary	Javadoc
OmnidirectionalField	Field incuding all the 3D space.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/OmnidirectionalField.html ...]
CircularField	Field described by a circular cone.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/CircularField.html ...]
RectangleField	Field described by a direction and two angles.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/RectangleField.html ...]

PyramidalField	Field described by an ordered array of directions.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/PyramidalField.html …]
BooleanField	Field defined by a boolean combination of two existing fields.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/BooleanField.html …]
InvertField	Field inverting an existing one.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/InvertField.html …]
EllipticField	This class describes an elliptic field of view.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/EllipticField.html …]
SectorField	This class describes a "sector" field of view.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/SectorField.html …]
AzimuthElevationField	This class describes a field of view defined by an azimuth/elevation array.	[[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/AzimuthElevationField.html …]

Methods

All fields implement the IFieldOfView interface (see [[[:Modèle:JavaDoc4.15]]/fr/cnes/sirius/patrius/fieldsofview/IFieldOfView.html javadoc]).

All fields can :

- return a boolean set to true if a user point (`direction`) is in the field,
- compute the minimal angular distance between a user point (`direction`) to their border,
- provide their name.

The associated methods are :

getAngularDistance(Vector3D) : double
> The convention is : the returned double is positive if the direction vector is in the field, negative otherwise, so it can be used directely by the event detector computations (the zero of the g function occurs when the direction comes in are goes out of the field).

```
// creation of a circular field of view
Vector3D axisVector = new Vector3D(2.0, 1.0, 3.0);
double apertureAngle = 1.2;
IFieldOfView field = new CircularFieldOfView(axisVector, apertureAngle);

// the direction
Vector3D direction = new Vector3D(1.0, 0.5, 2.0);

// angular distance to the direction
double angularDistance = field.getAngularDistance(direction);
```

isInTheField(Vector3D) : boolean

Returns true if the direction vector is in the field, false otherwise.

```
// angular distance to the direction
boolean direction in field = field.isInTheField(direction);
```

getName( ) : String
> Returns the String name of the field.

getMainDirection(Vector3D) : Vector3D
> Gets the main direction of the geometrical FOV. For every orthogonal section of the FOV, the main direction contains the center of such section.

getAngularDistance(Vector3D , AngularDistanceType) : double
> Computes the angular distance between a vector and the border of the field. The result is positive if the direction is in the field, negative otherwise. Several methods can be defined for the computation. The user can choose the more appropriate one from the enum `AngularDistanceType`. Gets the main direction of the geometrical FOV. For every orthogonal section of the FOV, the main direction contains the center of such section.

getAngularOpening(Vector3D) : double
> Gets the angular opening of the Field Of View (FOV) over a given direction. Considering the half-plane containing both the main direction of the FOV and the input direction, the angular opening is defined as the angle between the main direction and the intersection of the FOV border with the half-plane.

Tutorials

Tutorial 1

[Modèle:SpecialInclusion prefix=$theme sub section="Tuto1"/](#)

☒ Tips & Tricks

[Modèle:SpecialInclusion prefix=$theme sub section="Tips"/](#)

Récupérée de
« http://patrius.cnes.fr/index.php?title=User_Manual_4.15_Fields_of_view&oldid=3892 »
[Catégorie](#) :

- [User Manual 4.15 Spacecraft](#)

Menu de navigation

Outils personnels

- [18.221.59.121](#)
- [Discussion avec cette adresse IP](#)

PATRIUS

- [Welcome](#)

Evolutions

- [Main differences between V4.15 and V4.14](#)
- [Main differences between V4.14 and V4.13](#)
- [Main differences between V4.13 and V4.12](#)
- [Main differences between V4.12 and V4.11](#)
- [Main differences between V4.11 and V4.10](#)
- [Main differences between V4.10 and V4.9](#)
- [Main differences between V4.9 and V4.8](#)
- [Main differences between V4.8 and V4.7](#)
- [Main differences between V4.7 and V4.6.1](#)
- [Main differences between V4.6.1 and V4.5.1](#)
- [Main differences between V4.5.1 and V4.4](#)
- [Main differences between V4.4 and V4.3](#)
- [Main differences between V4.3 and V4.2](#)

* [Main differences between V4.2 and V4.1.1](#)
* [Main differences between V4.1.1 and V4.1](#)
* [Main differences between V4.1 and V4.0](#)
* [Main differences between V4.0 and V3.4.1](#)

User Manual

* [User Manual 4.15](#)
* [User Manual 4.14](#)
* [User Manual 4.13](#)
* [User Manual 4.12](#)
* [User Manual 4.11](#)
* [User Manual 4.10](#)
* [User Manual 4.9](#)
* [User Manual 4.8](#)
* [User Manual 4.7](#)
* [User Manual 4.6.1](#)
* [User Manual 4.5.1](#)
* [User Manual 4.4](#)
* [User Manual 4.3](#)
* [User Manual 4.2](#)
* [User Manual 4.1](#)
* [User Manual 4.0](#)
* [User Manual 3.4.1](#)
* [User Manual 3.3](#)

Tutorials

* [Tutorials 4.15](#)
* [Tutorials 4.14](#)
* [Tutorials 4.13.5](#)
* [Tutorials 4.12.1](#)
* [Tutorials 4.8.1](#)
* [Tutorials 4.5.1](#)
* [Tutorials 4.4](#)
* [Tutorials 4.1](#)
* [Tutorials 4.0](#)

Links

* [CNES freeware server](#)

Navigation

- Dernière modification de cette page le 21 novembre 2024 à 16:06.