LOFOffsetAttitudeLaw 4.4

LOFOffsetAttitudeLaw 4.4

```
public class LOFOffsetAttitudeLaw {

    public static void main(String[] args) throws PatriusException,
IOException, URISyntaxException {

        // Patrius Dataset initialization (needed for example to get the UTC
time
        PatriusDataset.addResourcesFromPatriusDataset() ;

        // Recovery of the UTC time scale using a "factory" (not to duplicate
such unique object)
        final TimeScale TUC = TimeScalesFactory.getUTC();

        // Date of the orbit given in UTC time scale)
        final AbsoluteDate date = new AbsoluteDate("2010-01-01T12:00:00.000",
TUC);

        // Getting the frame with wich will defined the orbit parameters
        // As for time scale, we will use also a "factory".
        final Frame GCRF = FramesFactory.getGCRF();

        // Initial orbit
        final double sma = 7200.e+3;
        final double exc = 0.01;
        final double inc = FastMath.toRadians(98.);
        final double pa = FastMath.toRadians(0.);
        final double raan = FastMath.toRadians(0.);
        final double anm = FastMath.toRadians(0.);
        final double MU = Constants.WGS84_EARTH_MU;

        final KeplerianParameters par = new KeplerianParameters(sma, exc,
inc, pa, raan, anm, PositionAngle.MEAN, MU);
        final Orbit iniOrbit = new KeplerianOrbit(par, GCRF, date);

        // Building a first attitude law
        final AttitudeLaw attitudeLaw0= new LofOffset(LOFType.TNW);
        final Attitude att0 = attitudeLaw0.getAttitude(iniOrbit);

        // Building a second attitude law with a 45 deg rotation on Z axis
        final double psi  = FastMath.toRadians(45.);
        final double teta = 0.;
        final double phi  = 0.;
        final AttitudeLaw attitudeLaw = new LofOffset(LOFType.TNW,
RotationOrder.ZYX, psi, teta, phi);
```

```
        final Attitude att = attitudeLaw.getAttitude(iniOrbit);

        // Rotation of the X axis
        Vector3D vec0 = att0.getRotation().applyTo(Vector3D.PLUS_I);
        Vector3D vec  = att.getRotation().applyTo(Vector3D.PLUS_I);
        double cos = vec.dotProduct(vec0);
        double ang = FastMath.acos(cos);
        System.out.println(FastMath.toDegrees(ang));

        // Rotation of the Y axis
        vec0 = att0.getRotation().applyTo(Vector3D.PLUS_J);
        vec  = att.getRotation().applyTo(Vector3D.PLUS_J);
        cos = vec.dotProduct(vec0);
        ang = FastMath.acos(cos);
        System.out.println(FastMath.toDegrees(ang));

        // Z axis comparison
        vec0 = att0.getRotation().applyTo(Vector3D.PLUS_K);
        vec  = att.getRotation().applyTo(Vector3D.PLUS_K);
        final Vector3D dVec = vec.subtract(vec0);
        final double norm = dVec.getNorm();
        System.out.println(norm);

    }

}
```


Menu de navigation

Outils personnels

- 3.137.166.61
- Discussion avec cette adresse IP
- Créer un compte
- Se connecter

Espaces de noms

- Page
- Discussion

Variantes

Affichages

- Lire

- Voir le texte source
- Historique
- Exporter en PDF

Plus

Rechercher

Rechercher Lire

PATRIUS

Evolutions

User Manual

- 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

- Accueil
- Modifications récentes
- Page au hasard
- Aide

Outils

- Pages liées
- Suivi des pages liées
- Pages spéciales
- Adresse de cette version
- Information sur la page
- Citer cette page

- Dernière modification de cette page le 3 octobre 2019 à 11:59.

- Politique de confidentialité
- À propos de Wiki

- Avertissements
-