SequenceOfAttitudes 4.4

SequenceOfAttitudes 4.4

```
public class SequenceOfAttitudeLaws {

    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 iniDate = 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(90.);
        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, iniDate);

        // Using the Meeus model for the Sun.
        final CelestialBody sun = new MeeusSun();
        final double sunRadius = Constants.SUN_RADIUS;

        // Definition of the Earth ellipsoid for later atmospheric density
computation
        final Frame ITRF = FramesFactory.getITRF();
        final double earthRadius = Constants.WGS84_EARTH_EQUATORIAL_RADIUS;
        final GeometricBodyShape earth = new
ExtendedOneAxisEllipsoid(earthRadius, Constants.WGS84_EARTH_FLATTENING, ITRF,
```

```
"EARTH");

        // Initializing attitude sequence
        final AttitudesSequence seqAtt = new AttitudesSequence();

        // Building a first attitude law (Sun pointing)
        final Vector3D firstAxis = new Vector3D(1., 0., 0.);
        final Vector3D secondAxis = new Vector3D(0., 1., 0.);
        final AttitudeLaw sunPointingLaw = new SunPointing(sun, firstAxis,
secondAxis, sun);

        // Building a second attitude law (LVLH)
        final AttitudeLaw lvlhLaw = new LofOffset(LOFType.LVLH);

        // Events that will switch from a law to another
        final double maxCheck = 10.;
        final double threshold = 1.e-3;
        final EventDetector eventEntryEclipse = new EclipseDetector(sun,
sunRadius, earth, earthRadius, 0,
                maxCheck, threshold, Action.RESET_STATE, Action.RESET_STATE);
        final EventDetector eventExitEclipse = new EclipseDetector(sun,
sunRadius, earth, earthRadius, 0,
                maxCheck, threshold, Action.RESET_STATE, Action.RESET_STATE);

        //Adding switches
        seqAtt.addSwitchingCondition(lvlhLaw, eventEntryEclipse, true, false,
sunPointingLaw);
        seqAtt.addSwitchingCondition(sunPointingLaw, eventExitEclipse, false,
true, lvlhLaw);

        testByPropagation(iniOrbit, seqAtt, sun);

    }

    public static void testByPropagation ( final Orbit iniOrbit, final
AttitudesSequence seqAtt, final CelestialBody sun ) throws PatriusException {

        // We create a spacecratftstate
        final SpacecraftState iniState = new SpacecraftState(iniOrbit);

        // Initialization of the Runge Kutta integrator with a 2 s step
        final double pasRk = 2.;
        final FirstOrderIntegrator integrator = new
ClassicalRungeKuttaIntegrator(pasRk);

        // Initialization of the propagator
        final NumericalPropagator propagator = new
NumericalPropagator(integrator);
        propagator.resetInitialState(iniState);

        // Forcing integration using cartesian equations
```

```
        propagator.setOrbitType(OrbitType.CARTESIAN);

        // Adding the attitude sequence
        propagator.setAttitudeProvider(seqAtt);
        seqAtt.registerSwitchEvents(propagator);

        // Loop every 10 mn ...
        final double step = 600.;
        final double epsilon = 1.e-12;
        for (int i = 1; i <= 20; i++) {

            AbsoluteDate date = iniOrbit.getDate().shiftedBy(i*step);

            final SpacecraftState state = propagator.propagate(date);

            // Attitude in LVLH
            final Attitude attLVLH = state.getAttitude(LOFType.LVLH);
            final double psiLVLH  =
attLVLH.getRotation().getAngles(RotationOrder.ZYX)[0];
            final double tetaLVLH =
attLVLH.getRotation().getAngles(RotationOrder.ZYX)[1];

            // Attitude in GCRF
            final Attitude attGCRF = state.getAttitude();
            final double psiGCRF  =
attGCRF.getRotation().getAngles(RotationOrder.ZYX)[0];
            final double tetaGCRF =
attGCRF.getRotation().getAngles(RotationOrder.ZYX)[1];

            // Direction of the Sun from the cdg of the satellite
            final Vector3D sunPos = sun.getPVCoordinates(date,
FramesFactory.getGCRF()).getPosition();
            final Vector3D satPos = state.getPVCoordinates().getPosition();
            final Rotation sunDir = new Rotation(Vector3D.PLUS_I,
sunPos.subtract(satPos));

            // Sun direction
            final double psiSun  = sunDir.getAngles(RotationOrder.ZYX)[0];
            final double tetaSun = sunDir.getAngles(RotationOrder.ZYX)[1];

            if ( (FastMath.abs(psiLVLH) < epsilon) || (FastMath.abs(tetaLVLH)
< epsilon) ) {
                System.out.println(date+" => LVLH mode");
            } else {
                System.out.println(date+" => Sun pointing mode");
                System.out.println("    Delta Psi  =
"+FastMath.toDegrees(psiSun-psiGCRF)+" deg");
                System.out.println("    Delta Teta =
"+FastMath.toDegrees(tetaSun-tetaGCRF)+" deg");
            }
```

```
        }

    }

}
```

Récupérée de « http://patrius.cnes.fr/index.php?title=SequenceOfAttitudes_4.4&oldid=2471 »

Menu de navigation

Outils personnels

- 18.191.237.228
- 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

- 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

- 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 12:01.

- Politique de confidentialité
- À propos de Wiki
- Avertissements

-