basics_1

March 26, 2021

1 Basics 1: Coordinate transformations

1. Convert the following state-vector from Cartesian components to Kepler elements:
   $x = 8751268.4691$ $[m]$
   $y = -7041314.6869$ $[m]$
   $z = 4846546.9938$ $[m]$
   $\dot{x} = 332.2601039$ $[ms^{-1}]$
   $\dot{y} = -2977.0815768$ $[ms^{-1}]$
   $\dot{z} = -4869.8462227$ $[ms^{-1}]$
2. Convert the following state-vector from Kepler elements to Cartesian components: $a = 12158817.9615$ $[m]$
   $e = 0.014074320051$ $[m]$
   $i = 52.666016957$ $[deg]$
   $RAAN = 323.089150643$ $[deg]$
   $\omega = 148.382589129$ $[deg]$
   $M = 112.192638384$ $[deg]$

1.1 Initialization and unit testing

[1]:
```
from math import degrees, radians
import numpy as np
import numpy.testing as npt
from transformations import cartesian_to_kepler, kepler_to_cartesian
```

[13]:
```
# Run unittests
! python -m unittest discover ./
```

```
…
----------------------------------------------------------------------
Ran 7 tests in 0.015s

OK
```

1.2 1. Cartesian -> Kepler

[3]:
```
position = [8751268.4691, -7041314.6869, 4846546.9938]
velocity = [332.2601039, -2977.0815768, -4869.8462227]

# Transform to kepler elements
a, e, i, raan, omega, true_anomaly, eccentric_anomaly, mean_anomaly =␣
 ↪cartesian_to_kepler(
    r=position,
    v=velocity,
)

# Print results
print(f"{a=}")
print(f"{e=}")
print(f"{degrees(i)=}")
print(f"{degrees(raan)=}")
print(f"{degrees(omega)=}")
print(f"{degrees(true_anomaly)=}")
print(f"{degrees(eccentric_anomaly)=}")
print(f"{degrees(mean_anomaly)=}")
```

```
a=12273086.180973208
e=0.005022166693730592
degrees(i)=109.818773825313
degrees(raan)=132.23369779015522
degrees(omega)=105.06673298740603
degrees(true_anomaly)=50.02799134864682
degrees(eccentric_anomaly)=49.80782656785884
degrees(mean_anomaly)=49.58801968983559
```

[4]:
```
# Check
position_2, velocity_2 = kepler_to_cartesian(
    a=a,
    e=e,
    i=i,
    raan=raan,
    omega=omega,
    mean_anomaly=mean_anomaly
)
npt.assert_allclose(position_2, position)
npt.assert_allclose(velocity_2, velocity)
```

1.3 2. Kepler -> Cartesian

[10]:
```
a = 12158817.9615
e = 0.014074320051
i = radians(52.666016957)
raan = radians(323.089150643)
omega = radians(148.382589129)
mean_anomaly = radians(112.192638384)

# Transform to cartesian components
position, velocity = kepler_to_cartesian(
    a=a,
    e=e,
    i=i,
    raan=raan,
    omega=omega,
    mean_anomaly=mean_anomaly,
)

# Print results
print(f"{position=}")
print(f"{velocity=}")
```

```
position=array([-5760654.23005053, -4856967.48824368, -9627444.86215477])
velocity=array([ 4187.66125138, -3797.5451854 ,  -683.61512604])
```

[12]:
```
# Check
a_2, e_2, i_2, raan_2, omega_2, true_anomaly_2, \
    eccentric_anomaly_2, mean_anomaly_2 = cartesian_to_kepler(
        r=position,
        v=velocity)

npt.assert_allclose(a_2, a)
npt.assert_allclose(e_2, e)
npt.assert_allclose(i_2, i)
npt.assert_allclose(raan_2, raan)
npt.assert_allclose(omega_2, omega)
npt.assert_allclose(mean_anomaly_2, mean_anomaly)
```

[ ]:
```

```