RigidMotion 2.5

Stephanie Peron, Christophe Benoit, Pascal Raud
- Onera -

1 RigidMotion: compute/define rigid motions

1.1 Preamble

RigidMotion enables to define or compute rigid motions for arrays (as defined in Converter documentation) or for CGSN/Python trees (pyTrees).

This module is part of Cassiopee, a free open-source pre- and post-processor for CFD simulations.

To use it with the Converter array interface, you must import the RigidMotion module:

```
import RigidMotion as R
```

Then, in the following, a is an array, and A a list of arrays.
To use it with the pyTree interface, you must import the module:

```
import RigidMotion.PyTree as R
```

Then, in the following, a is a zone node and A is a list of zone nodes or a complete pyTree.

1.2 Prescribed motion of type 1

R.setPrescribedMotion1: set a precribed motion defined by a translation of the origin (tx,ty,tz), the center of a rotation (cx,cy,cz), the second point of the rotation axis (ex,ey,ez) and the rotation angle in degrees. They can depend on time t:

```
a = R.setPrescribedMotion1(a, motionName, tx="0", ty="0", tz="0", cx="0", cy="0", cz="0", ex="0", ey="0", ez="0", angle="0")
```

(See: setPrescribedMotion1PT.py)

R.setPrescribedMotion2: set a precribed motion defined by a elsA rotor motion:

```
a = R.setPrescribedMotion2(a, motionName, transl_speed, psi0, psi0_b, alp_pnt, alp_vct, alp0, rot_pnt, rot_vct, rot_omg, del_pnt, del_vct, del0, delc, dels, bet_pnt, bet_vct, bet0, betc, bets, tet_pnt, tet_vct, tet0, tetc, tets, span_vct, pre_lag_pnt, pre_lag_vct, pre_lag_ang, pre_con_pnt, pre_con_vct, pre_con_ang)
```

(See: setPrescribedMotion2PT.py)

R.setPrescribedMotion3: set a precribed motion defined by a constant speed rotation and translation. omega is in rad/time unit:

a = R.setPrescribedMotion3(a, motionName, transl_speed, axis_pnt, axis_vct, omega)

(See: setPrescribedMotion3PT.py)

1.3 General functions

R.evalPosition: evaluate the position at time t according to a motion.
If the motion is defined in a:

a = R.evalPosition(a, time)

Eval position at given time, when motion is described by a function. F(t) is a function describing motion. F(t) = (centerAbs(t), centerRel(t), rot(t)), where centerAbs(t) are the coordinates of the rotation center in the absolute frame, centerRel(t) are the coordinates of the rotation center in the relative (that is array's) frame and rot(t), the rotation matrix:

b = R.evalPosition(a, time, F) *.or.* B = R.evalPosition(A, time, F)

(See: evalPositionPT.py) (See: evalPosition2.py) (See: evalPosition2PT.py)

1.4 Example files

Example file: setPrescribedMotion1PT.py

```
# - setPrescribedMotion1 (pyTree) -
# Motion defined by time string
import RigidMotion.PyTree as R
import Converter.PyTree as C
import Geom.PyTree as D

a = D.sphere((1.2,0.,0.), 0.2, 30)
a = R.setPrescribedMotion1(a, 'trans', tx="{t}")

C.convertPyTree2File(a, 'out.cgns')
```

Example file: setPrescribedMotion2PT.py

```
# - setPrescribedMotion2 (pyTree) -
# Motion defined by a Cassiopee Solver rotor motion
import RigidMotion.PyTree as R
import Converter.PyTree as C
import Geom.PyTree as D

a = D.sphere((1.2,0.,0.), 0.2, 30)
a = R.setPrescribedMotion2(a, 'rotor', transl_speed=(0.1,0,0), rot_omg=1.)

C.convertPyTree2File(a, 'out.cgns')
```

Example file: setPrescribedMotion3PT.py

```
# - setPrescribedMotion3 (pyTree) -
# Motion defined by a constant speed and rotation speed
import RigidMotion.PyTree as R
import Converter.PyTree as C
```

```
import Geom.PyTree as D

a = D.sphere((1.2,0.,0.), 0.2, 30)
a = R.setPrescribedMotion3(a, 'mot', transl_speed=(1,0,0))

C.convertPyTree2File(a, 'out.cgns')
```

Example file: evalPositionPT.py

```
# - evalPosition (pyTree) -
import RigidMotion.PyTree as R
import Converter.PyTree as C
import Geom.PyTree as D

a = D.sphere((1.2,0.,0.), 0.2, 30)
a = R.setPrescribedMotion1(a, 'trans', tx="{t}")
b = R.evalPosition(a, time=0.1)

C.convertPyTree2File(b, 'out.cgns')
```

Example file: evalPosition2.py

```
# - evalPosition (array) -
import RigidMotion as R
import Transform as T
import Generator as G
import Converter as C
from math import *

# Coordinates of rotation center in absolute frame
def centerAbs(t): return [t, 0, 0]

# Coordinates of rotation center in relative frame
def centerRel(t): return [5, 5, 0]

# Rotation matrix
def rot(t):
    omega = 0.1
    m = [[cos(omega*t), -sin(omega*t), 0],
         [sin(omega*t), cos(omega*t),  0],
         [0,            0,             1]]
    return m

# Complete motion
def F(t): return (centerAbs(t), centerRel(t), rot(t))

# Create a structured Cartesian grid
a = G.cart((0,0,0), (1,1,1), (11,11,1))
# Move the mesh
t = 3.
b = R.evalPosition(a, t, F)
C.convertArrays2File([b], "out1.plt")

# Equivalent to:
c = T.rotate(a, (5,5,0), (0,0,1), 0.1*t*180/3.14)
c = T.translate(c, (t-5,-5,0))
C.convertArrays2File([c], "out2.plt")
```

Example file: evalPosition2PT.py

```
# - evalPosition (PyTree) -
```

```
import RigidMotion.PyTree as R
import Generator.PyTree as G
import Converter.PyTree as C
from math import *

# Coordonnees du centre de rotation dans le repere absolu
def centerAbs(t): return [t, 0, 0]

# Coordonnees du centre de la rotation dans le repere entraine
def centerRel(t): return [5, 5, 0]

# Matrice de rotation
def rot(t):
    omega = 0.1
    m = [[cos(omega*t), -sin(omega*t), 0],
         [sin(omega*t), cos(omega*t),  0],
         [0,            0,             1]]
    return m

# Mouvement complet
def F(t): return (centerAbs(t), centerRel(t), rot(t))

a = G.cart((0,0,0), (1,1,1), (11,11,2))

# Move the mesh
time = 3.
b = R.evalPosition(a, time, F); b[0]='move'
C.convertPyTree2File([a,b], "out.cgns")
```