Transform V1.4
- User Guide -

C. Benoit, G. Jeanfaivre, S. Peron et P. Raud
Onera / DSNA

September 23, 2009

1 Transform : transformation module

1.1 Preamble

Transform module works on arrays or on PyTrees containing grid information (coordinates must be defined).
For use with the array interface, you have to import Transform module:

```
import Transform as T
```

Then, a designates an array, and A designates a list of arrays.
For use with the PyTree interface:

```
import Transform.PyTree as T
```

Then, a designates a zone node, A designates a list of zone nodes or a complete pyTree.

1.2 Simple operations

Warning : the PyTree functions *subzone, oneovern, join, reorder* delete the 'BCMatch' boundary conditions. They can be rebuilt by the connectMatch function of Connector module.

Extract a subzone of a:

```
b = T.subzone(a, (imin,jmin,kmin), (imax,jmax,kmax))
```

(See : Examples/transform/subzone.py) (See : Examples/transform/subzonePT.py)

Join two arrays in one (if possible) or join a list of arrays in one (if possible). Warning : join does not take into account modifications of the boundary conditions:

```
c = T.join(a, b) .or. c = T.join( A )
```

(See : Examples/transform/join.py) (See : Examples/transform/joinPT.py)

Patch (replace) b into a from position (i,j,k):

```
c = T.patch(a, b, (i,j,k))
```

(See : Examples/transform/patch.py) (See : Examples/transform/patchPT.py)

Extract one point over N points from a:

```
b = T.oneovern(a, (Ni,Nj,Nk)) .or. B = T.oneovern(A, (Ni,Nj,Nk))
```

(See : Examples/transform/oneovern.py) (See : Examples/transform/oneovernPT.py)

Change the (i,j,k) ordering of structured array a. I becomes J, J becomes I and K becomes -K:

```
b = T.reorder(a, (2,1,-3)) .or. B = T.reorder(A, (2,1,-3))
```

Change the ordering of a TRI- or QUAD-array. All elements are numbered in the same way:

```
b = T.reorder(a, (-1,)) .or. B = T.reorder(A, (-1,))
```

(See : Examples/transform/reorder.py) (See : Examples/transform/reorderPT.py)

Functions such as Post.usurp and Post.integ require that the orientation of block normals must be the same for all blocks. The function reorderAll reorders the connectivity of some blocks, with respect to a reference surface block. All the blocks are supposed to represent a unique solid surface. The parameter 'dir' allows the user to change the orientation of the normals. Default value is 1, and 'dir' = -1 changes the normals to the opposite direction:

```
B = P.reorderAll( A, dir )
```

NB: if reorder fails, A is returned.

(See : Examples/post/reorderAll.py) (See : Examples/post/reorderAllPT.py)

Add a k plane. If applied on a pytree, add kmax = 2 on all BCs.

```
b = T.addkplane(a) .or. B = T.addkplane(A)
```

(See : Examples/transform/addkplane.py) (See : Examples/transform/addkplanePT.py)

1.3 Mesh positioning

Following functions are applicable to both structured and unstructured grids.

Make a translation of a vector (0.,1.,0.):

```
b = T.translate(a, (0.,1.,0.)) .or. B = T.translate(A, (0.,1.,0.))
```

(See : Examples/transform/translate.py)

Make a rotation of center (0.2,0.2,0.) around axis z of angle 18. (degrees):

```
b = T.rotate(a, (0.2,0.2,0.), (0.,0.,1.), 18.) .or. B = T.rotate(A, (0.2,0.2,0.), (0.,0.,1.), 18.)
```

(See : Examples/transform/rotate.py)

1.4 Mesh transformation

Following functions, except "perturbate", are applicable to both structured and unstructured grids.

Make an homothety of center C (0.,0.,0.) and of multiplication factor alpha = 0.1 (CM' = alpha * CM):

```
b = T.homothety(a, (0.,0.,0.), 0.1) .or. B = T.homothety(A, (0.,0.,0.), 0.1)
```

(See : Examples/transform/homothety.py)

Make a contraction of a regarding a plane defined by point (0.,0.,0.) and by dir1 (1,0,0) and dir2 (0,1,0) and of multiplication factor alpha = 0.1:

```
b = T.contract(a, (0.,0.,0.), (1,0,0), (0,1,0), 0.1) .or. B = T.contract(A, (0.,0.,0.), (1,0,0), (0,1,0), 0.1)
```

(See : Examples/transform/contract.py)

Make a symmetry of a considering the plane passing by a point (1.,2.,3.) and defined by two vectors (1,0,0) and (0,1,0). Beware the (i,j,k) trihedra may be modified. Use reorder to avoid this:

```
b = T.symetrize(a, (1.,2.,3.), (1,0,0), (0,1,0)) .or. B = T.symetrize(A, (1.,2.,3.), (1,0,0), (0,1,0))
```

(See : Examples/transform/symetrize.py) Perturbate randomly a with a given radius:

```
b = T.perturbate(a, 0.1)
```

(See : Examples/transform/perturbate.py)

Split a i-array following curvature. Sensibility is the variation of curvature triggering split (1.e-2 for instance):

```
A = T.splitCurvature(a, 1.e-2)
```

(See : Examples/transform/splitCurvature.py) (See : Examples/transform/splitCurvaturePT.py)

Split a i-array following curvature, using B-splines. The curve can be closed or not. Parameter cs is a threshold curvature, so that the initial curve is split at points of curvature greater than cs:

```
B = T.splitSpline(a, cs)
```

(See : Examples/transform/splitSpline.py) (See : Examples/transform/splitSplinePT.py)

Split an unstructured array into connex parts:

```
B = T.splitConnexity(a)
```

(See : Examples/transform/splitConnexity.py)

Split structured blocks when their number of points is greater than N:

```
B = T.splitSize(a, N) .or. B = T.splitSize(A, N)
```

(See : Examples/transform/splitSize.py) (See : Examples/transform/splitSizePT.py)

Split structured blocks where matching joins have an even number of connections:

```
B = T.splitMultiplePts(A)
```

(See : Examples/transform/splitMultiplePts.py) (See : Examples/transform/splitMultiplePtsPT.py)

Deform a by moving point (i,j,k) of vector (dx,dy,dz). Fourth argument controls the zone affected by point movement. If 0, only point is modified, if 1, all mesh is affected. Last argument is a damping coefficient in j similar to stick:

```
b = T.deformPoint(a, (i,j,k), (dx,dy,dz), 0.5, 0.4)
```

(See : Examples/transform/deformPoint.py) (See : Examples/transform/deformPointPT.py)

Deform a by moving each point of a given vector array (array):

```
b = T.deform(a, vector)
```

Deform a by moving each point of a given vector defined by variables 'dx','dy','dz', defined in a (pyTree):

```
b = T.deform(a, ['dx','dy','dz'])
```

(See : Examples/transform/deform.py) (See : Examples/transform/deformPT.py)

Project a surface mesh a onto a list of surfaces S following a given direction:

```
b = T.projectDir(a, S, (1.,0,0))
```

(See : Examples/transform/projectDir.py)

Project a surface mesh a onto a list of surfaces S following normals:

```
b = T.projectOrtho(a, S)
```

(See : Examples/transform/projectOrtho.py)

Project a surface mesh a onto a list of surfaces S following rays issued from P:

```
b = T.projectRay(a, S, P)
```

(See : Examples/transform/projectRay.py)

Move a following motion function to time t. 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) frame and rot(t), the rotation matrix:

```
b = T.move(a, F, t) .or. B = T.move(A, F, t)
```

(See : Examples/transform/move.py)

1.5 Constructive geometry operations

Stick b on a. Windows that are stuck together are (j = 1) windows. Third argument is a threshold. Points that are further than this threshold are not stuck. Last argument is a damping coefficient in [0,1]. If 0, no damping is performed in j. If 1, all points are damped:

```
c = T.stick(a, b, 0.1, 0.5)
```

(See : Examples/transform/stick.py) (See : Examples/transform/stickPT.py)

1.6 Example files

Example file : Examples/transform/subzone.py

```
# - subzone (array) -
import Converter as C
import Transform as T
import Generator as G

a = G.cart((0,0,0), (1,1,1), (10,20,10))
a = C.initVars(a, 'celln', 1)
b = T.subzone(a, (3,3,3), (7,8,5))
C.convertArrays2File([b], 'out.plt')
```

Example file : Examples/transform/subzonePT.py

```
# - subzone (pyTree)-
import Converter.PyTree as C
import Transform.PyTree as T
import Generator.PyTree as G

a = G.cart((0,0,0), (1,1,1), (10,20,1))
a = T.subzone(a, (3,3,1), (7,8,1))
t = C.newPyTree(['Base',2]); t[1][2].append(a)
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/join.py

```
# - join (array) -
import Geom as D
import Transform as T
import Converter as C
import Generator as G

a1 = G.cartTetra((0.,0.,0.), (1.,1.,1), (11,11,1))
a2 = G.cartTetra((9.,0.,0.), (1.,1.,1), (10,10,1))
a = T.join (a1, a2)
C.convertArrays2File([a], 'out.plt')
```

Example file : Examples/transform/joinPT.py

```
# - join (pyTree) -
import Geom.PyTree as D
import Transform.PyTree as T
import Converter.PyTree as C

a1 = D.naca(12., 5001)
a2 = D.line((1.,0.,0.),(20.,0.,0.),5001)
a = T.join(a1, a2)
t = C.newPyTree(['Base',1]); t[1][2].append(a)
C.convertPyTree2File(t, "out.cgns")
```

Example file : Examples/transform/patch.py

```
# - patch (array) -
import Transform as T
import Generator as G
import Converter as C

c1 = G.cart((0,0,0), (0.01,0.01,1), (201,101,1))
c2 = G.cart((0,0,0), (0.01,0.01,1), (51,81,1))
c2 = T.rotate(c2, (0,0,0),(0,0,1),0.2)
a = T.patch(c1, c2,(1,1,1))
C.convertArrays2File([a], 'out.plt')
```

Example file : Examples/transform/patchPT.py

```
# - patch (pyTree) -
import Transform.PyTree as T
import Generator.PyTree as G
import Converter.PyTree as C

c1 = G.cart((0,0,0), (0.01,0.01,1), (201,101,1))
c2 = G.cart((0,0,0), (0.01,0.01,1), (51,81,1))
a = T.patch(c1, c2,(1,1,1))
t = C.newPyTree(['Base',2]); t[1][2].append(a)
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/oneovern.py

```
# - oneovern (array) -
import Transform as T
import Converter as C
import Generator as G

a = G.cart((0,0,0), (1,1,1), (10,10,1))
a2 = T.oneovern(a, (2,2,2))
C.convertArrays2File([a2],"out.plt", "bin_tp")
```

Example file : Examples/transform/oneovernPT.py

```
# - oneovern (pyTree) -
import Transform.PyTree as T
import Converter.PyTree as C
import Generator.PyTree as G

a = G.cart((0,0,0), (1,1,1), (10,10,1))
a2 = T.oneovern(a, (2,2,1)); a2[0] = 'cart2'
t = C.newPyTree(['Base',2]); t[1][2] = t[1][2] + [a,a2]
C.convertPyTree2File(t, "out.cgns")
```

Example file : Examples/transform/reorder.py

```
# - reorder (array) -
import Generator as G
import Transform as T
import Converter as C

# Structure
a = G.cart((0,0,0),(1,1,1),(5,7,9))
a = T.reorder(a, (3,-2,-1))
C.convertArrays2File([a], 'out.plt')

# Non structure
a = G.cartTetra((0,0,0),(1,1,1),(3,3,1))
c = a[2]; c[1,0] = 5; c[2,0] = 2
a = T.reorder(a, (1,))
C.convertArrays2File([a], 'out.plt')
```

Example file : Examples/transform/reorderPT.py

```
# - reorder (pyTree) -
import Generator.PyTree as G
import Transform.PyTree as T
import Converter.PyTree as C

a = G.cart((0,0,0), (1,1,1), (8,9,20))
a = T.reorder(a, (2,1,-3))
t = C.newPyTree(['Base']); t[1][2].append(a)
C.convertPyTree2File(t, "out.cgns")
```

Example file : Examples/post/reorderAll.py

```
# - reorderAll (array) -
import Converter as C
import Generator as G
import Post as P
import Transform as T

#-------------------------------------------------
# test 1 : blocs recouvrants pareillement orientes
#-------------------------------------------------
ni = 30 ; nj = 40

# Maillage en noeuds
m1 = G.cart((0,0,0), (10./(ni-1),10./(nj-1),1), (ni,nj,1))
m2 = T.rotate(m1, (0.2,0.2,0.), (0.,0.,1.), 15.)
a = [m1,m2]
C.convertArrays2File(a, "out2.plt", "bin_tp")
a = P.reorderAll(a,1)
C.convertArrays2File(a, "out3.plt", "bin_tp")

#-------------------------------------------------
# test 2 : blocs recouvrants orientes differemment
#-------------------------------------------------
ni = 30 ; nj = 40
# Maillage en noeuds
m1 = G.cart((0,0,0), (10./(ni-1),10./(nj-1),1), (ni,nj,1))
m2 = T.rotate(m1, (0.2,0.2,0.), (0.,0.,1.), 15.)
m2 = T.reorder(m2,(-1,2,3))
a = [m1,m2]
C.convertArrays2File(a, "out4.plt","bin_tp")

a = P.reorderAll(a,1)
C.convertArrays2File(a, "out5.plt","bin_tp")

#--------------------------------
# test 3: blocs sans intersection
```

```
#---------------------------------
ni = 30 ; nj = 40

# Maillage en noeuds
m1 = G.cart((0,0,0), (10./(ni-1),10./(nj-1),1), (ni,nj,1))
m2 = T.translate(m1, (0.,12,0.))
a = [m1,m2]
C.convertArrays2File(a, "out0.plt", "bin_tp")
a = P.reorderAll(a)
C.convertArrays2File(a, "out1.plt", "bin_tp")
```

Example file : Examples/post/reorderAllPT.py

```
# - reorderAll (pyTree) -
import Converter.PyTree as C
import Generator.PyTree as G
import Post.PyTree as P
import Transform.PyTree as T

ni = 30 ; nj = 40; nk = 1
m1 = G.cart((0,0,0), (10./(ni-1),10./(nj-1),1), (ni,nj,nk)); m1[0]='cart1'
m2 = T.rotate(m1, (0.2,0.2,0.), (0.,0.,1.), 15.);
m2 = T.reorder(m2,(-1,2,3)); m2[0]='cart2'
t = C.newPyTree(['Base',2]); t[1][2].append(m1); t[1][2].append(m2)
t = P.reorderAll(t,1)
C.convertPyTree2File(t, "out.cgns", "bin_cgns")
```

Example file : Examples/transform/addkplane.py

```
# - addkplane (array) -
import Geom as D
import Transform as T
import Converter as C

a = D.naca(12., 501)
a = T.addkplane(a)
C.convertArrays2File([a], "out.plt")
```

Example file : Examples/transform/addkplanePT.py

```
# - addkplane (pyTree) -
import Generator.PyTree as G
import Transform.PyTree as T
import Converter.PyTree as C

a = G.cart((0.,0.,0.),(0.1,0.1,1.),(10,10,2))
a = C.addBC2Zone(a, 'wall1', 'BCWall', 'imin')
a = C.addBC2Zone(a, 'overlap1', 'BCOverlap', 'imax')
a = T.addkplane(a)
t = C.newPyTree(['Base']); t[1][2].append(a)
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/translate.py

```
# - translate (array) -
import Transform as T
import Generator as G
import Converter as C

a = G.cart( (0,0,0), (1,1,1), (10,10,1))
b = T.translate(a, (-1.,0.,0.))
C.convertArrays2File([a,b], 'out.plt')
```

Example file : Examples/transform/rotate.py

```
# - rotate (array) -
import Generator as G
import Transform as T
import Converter as C

a = G.cart( (0,0,0), (1,1,1), (10,10,1))
b = T.rotate(a, (0.,0.,0.), (0.,0.,1.), 30.)
C.convertArrays2File([a,b], 'out.plt')
```

Example file : Examples/transform/homothety.py

```
# - homothety (array) -
import Generator as G
import Transform as T
import Converter as C

a = G.cart((0,0,0), (1,1,1), (10,10,1))
b = T.homothety(a, (0.,0.,0.), 2.)
C.convertArrays2File([a,b], 'out.plt')
```

Example file : Examples/transform/contract.py

```
# - contract (array) -
import Generator as G
import Transform as T
import Converter as C

a = G.cart( (0,0,0), (1,1,1), (10,10,10))
b = T.contract(a, (0.,0.,0.), (1,0,0), (0,1,0), 0.1)
C.convertArrays2File([a,b], 'out.plt')
```

Example file : Examples/transform/symetrize.py

```
# - symetrize (array) -
import Generator as G
import Transform as T
import Converter as C

a = G.cart( (0,0,0), (1,1,1), (10,10,1))
b = T.symetrize(a, (0.,0.,0.), (1,0,0), (0,0,1))
C.convertArrays2File([a,b], "out.plt", "bin_tp")
```

Example file : Examples/transform/perturbate.py

```
# - perturbate (array) -
import Generator as G
import Transform as T
import Converter as C

a = G.cart((0,0,0), (1,1,1), (10,10,1))
a = C.addVars(a, 'ro')
a = T.perturbate(a, 0.1)
C.convertArrays2File([a], "out.plt", "bin_tp")
```

Example file : Examples/transform/splitCurvature.py

```
# - splitCurvature (array) -
import Converter as C
import Transform as T
import Geom as D

naca = D.naca(12., 5001)
line = D.line((1.,0.,0.), (20.,0.,0.), 5001)
a = T.join(naca, line)
line2 = D.line((20.,0.,0.), (1.,0.,0.), 5001)
a = T.join(line2, a)
list = T.splitCurvature(a, 1.e-2)
C.convertArrays2File(list, "out.plt", "bin_tp")
```

Example file : Examples/transform/splitCurvaturePT.py

```
# - splitCurvature (pyTree) -
import Converter.PyTree as C
import Geom.PyTree as D
import Transform.PyTree as T

a = D.naca(12,101)
a2 = D.line((1,0,0), (2,0,0), 50)
a = T.join(a, a2)
a2 = D.line((2,0,0), (1,0,0), 50)
a = T.join(a, a2)
zones = T.splitCurvature(a, 1.e-2)
t = C.newPyTree(['Base',1]); t[1][2].append(a)
t[1][2] = t[1][2] + zones
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/splitSpline.py

```
# - splitSpline (array) -
import Converter as C
import Transform as T
import Geom as D

pts = C.array('x,y,z', 7, 1, 1)
x = pts[1][0]; y = pts[1][1]; z = pts[1][2]
x[0]= 6.; x[1] = 5.4; x[2]=4.8; x[3] = 2.5; x[4]  = 0.3
y[0]=10.; y[1]=0.036; y[2]=-5.;y[3]=0.21;y[4]=0.26;y[5]=7.
z[0]=1.; z[1]=1.; z[2]=1.;z[3]=1.;z[4]=1.;z[5]=1.; z[6]=1.

a = D.bezier( pts, 50 )
L = T.splitSpline(a)
C.convertArrays2File([a]+L,"out.plt","bin_tp")
```

Example file : Examples/transform/splitSplinePT.py

```
# - splitSpline (pyTree)-
import Converter.PyTree as C
import Geom.PyTree as D
import Transform.PyTree as T

a = D.naca(12.5000)
zones = T.splitSpline(a, 10.)
t = C.newPyTree(['Base',1]); t[1][2].append(a)
t[1][2] = t[1][2] + zones
C.convertPyTree2File(t, "out.cgns", "bin_cgns")
```

Example file : Examples/transform/splitConnexity.py

```
# - splitConnexity (array) -
import Converter as C
import Transform as T
import Geom as D

a = D.text2D("CASSIOPEE")
B = T.splitConnexity(a)
C.convertArrays2File(B, 'out.plt')
```

Example file : Examples/transform/splitSize.py

```
# - splitSize (array) -
import Generator as G
import Transform as T
import Converter as C

a = G.cart((0,0,0),(1,1,1),(50,20,10))
B = T.splitSize(a, 2000)
C.convertArrays2File(B, 'out.plt')
```

Example file : Examples/transform/splitSizePT.py

```
# - splitSize (pyTree) -
import Generator.PyTree as G
import Transform.PyTree as T
import Converter.PyTree as C

t = C.newPyTree(['Base'])
a = G.cart((0,0,0),(1,1,1),(50,20,10)); t[1][2].append(a)
t = T.splitSize(t, 2000)
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/splitMultiplePts.py

```
# - splitMultiplePts (array) -
import Generator as G
import Transform as T
import Converter as C

z0 = G.cart((0.,0.,0.),(0.1,0.1,1.),(10,10,1))
z1 = T.subzone(z0,(1,1,1),(5,10,1))
z2 = T.subzone(z0,(5,1,1),(10,5,1))
z3 = T.subzone(z0,(5,5,1),(10,10,1))
zones = [z1,z2,z3]
zones = T.splitMultiplePts(zones)
C.convertArrays2File(zones, 'out.plt')
```

Example file : Examples/transform/splitMultiplePtsPT.py

```
# - splitMultiplePts (pyTree) -
import Generator.PyTree as G
import Transform.PyTree as T
import Converter.PyTree as C

z0 = G.cart((0.,0.,0.),(0.1,0.1,1.),(10,10,2))
z1 = T.subzone(z0,(1,1,1),(5,10,2)); z1[0] = 'cart1'
z2 = T.subzone(z0,(5,1,1),(10,5,2)); z2[0] = 'cart2'
z3 = T.subzone(z0,(5,5,1),(10,10,2)); z3[0] = 'cart3'
z0 = T.translate(z0,(-0.9,0.,0.)); z0[0] = 'cart0'
z4 = G.cart((-0.9,0.9,0.),(0.1,0.1,1.),(19,5,2)); z4[0] = 'cart4'
t = C.newPyTree(['Base',3]); t[1][2] = t[1][2] + [z1,z2,z3,z4]

import Connector.PyTree as X
t = X.connectMatch(t)
t = C.fillEmptyBCWith(t, 'wall', 'BCWall', '2D')
t = T.splitMultiplePts(t)
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/deformPoint.py

```
# - deformPoint (array) -
import Generator as G
import Transform as T
import Converter as C

a1 = G.cart((0,0,0), (1,1,1), (10,10,1))
a2 = T.deformPoint(a1, (1,1,1), (0.1,0.1,0.), 0.5, 0.4)
C.convertArrays2File([a2], "out.plt", "bin_tp")
```

Example file : Examples/transform/deformPointPT.py

```
# - deformPoint (PyTree) -
import Generator.PyTree as G
import Transform.PyTree as T
import Converter.PyTree as C
```

```
a = G.cart((0,0,0), (1,1,1), (10,10,3))
a = T.deformPoint(a, (1,1,1), (0.1,0.1,0.), 0.5, 0.4)
t = C.newPyTree(['Base',3]); t[1][2].append(a)
C.convertPyTree2File(t, "out.cgns", "bin_cgns")
```

Example file : Examples/transform/deform.py

```
# - deform (array) -
import Converter as C
import Generator as G
import Geom as D
import Transform as T

a = D.sphere( (0,0,0), 1., 50 )
n = G.getNormalMap(a)
n = C.center2Node(n) ; n[1] = n[1]*10
b = T.deform(a, n)
C.convertArrays2File([a,b], 'out.plt')
```

Example file : Examples/transform/deformPT.py

```
# - deform (pyTree) -
import Converter.PyTree as C
import Generator.PyTree as G
import Geom.PyTree as D
import Transform.PyTree as T

a = G.cart((0.,0.,0.),(1.,1.,1.),(10,10,2))
vect = ['hx','hy','hz']
a = C.addVars(a, vect)
a = C.initVars(a, 'hx', 10.)
a = T.deform(a, vect)
t = C.newPyTree(['Base',3]); t[1][2].append(a)
C.convertPyTree2File(t, 'out.cgns')
```

Example file : Examples/transform/projectDir.py

```
# - projectDir (array) -
import Geom as D
import Converter as C
import Generator as G
import Transform as T
a = D.sphere((0,0,0), 1., 20)
b = G.cart((1.1,-0.1,-0.1),(0.1,0.1,0.1), (1,5,5))
c = T.projectDir(b, [a], (1.,0,0))
C.convertArrays2File([a,b,c], 'out.plt', 'bin_tp')
```

Example file : Examples/transform/projectOrtho.py

```
# - projectOrtho (array) -
import Geom as D
import Converter as C
import Generator as G
import Transform as T
a = D.sphere((0,0,0), 1., 400)
b = G.cart((-0.5,-0.5,-0.5),(0.05,0.05,0.1), (20,20,1))
c = T.projectOrtho(b, [a])
C.convertArrays2File([a,b,c], 'out.plt', 'bin_tp')
```

Example file : Examples/transform/projectRay.py

```
# - projectRay (array) -
import Geom as D
import Converter as C
import Generator as G
import Transform as T
a = D.sphere((0,0,0), 1., 20)
b = G.cart((1.1,-0.1,-0.1),(0.1,0.1,0.1), (1,5,5))
c = T.projectRay(b, [a], (0,0,0))
C.convertArrays2File([a,b,c], 'out.plt')
```

Example file : Examples/transform/move.py

```
# - move (array) -
import Transform as T
import Generator as G
import Converter 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))

#-----------------------------------
# Create a structured Cartesian grid
#-----------------------------------
a = G.cart((0,0,0), (1,1,1), (11,11,1))

# Move the mesh
t = 3.
b = T.move(a, F, t)
C.convertArrays2File([b], "news.plt", "bin_tp")

# 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], "news2.plt", "bin_tp")

#----------------------------
# Create an unstructured mesh
#----------------------------
a = G.cartTetra((0.,0.,0.), (0.1,0.1,0.2), (10,10,10))
# Move the mesh
t = 3.
b = T.move(a, F, t)
C.convertArrays2File([b], "newu.plt", "bin_tp")

# 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], "newu2.plt", "bin_tp")
```

Example file : Examples/transform/stick.py

```
# - stick (array) -
import Geom as D
import Generator as G
import Transform as T
import Converter as C

a = D.naca(12., 5001)
a1 = T.addkplane(a)
a = G.cart((0.,0.,0.), (0.001,0.001,1.), (100,50,1))
a = G.enforcePlusY(a, 7.5e-6, (20,50))
a2 = T.translate(a, (-0.03,0.05,0));
b = T.stick(a1, a2, 0.5, 1)
C.convertArrays2File([b], "out.plt", "bin_tp")
```

Example file : Examples/transform/stickPT.py

```
# - stick (pyTree)-
import Geom.PyTree as D
import Generator.PyTree as G
import Transform.PyTree as T
import Converter.PyTree as C

a = D.naca(12., 5001); a1 = T.addkplane(a)
a = G.cart((0.,0.,0.), (0.001,0.001,1.), (100,50,1))
a = G.enforcePlusY(a, 7.5e-6, (20,50))
a2 = T.translate(a, (-0.03,0.05,0))
b = T.stick(a1, a2, 0.5, 1)
t = C.newPyTree(['Base',2]);t[1][2].append(b)
C.convertPyTree2File(t, "out.cgns")
```