OpenFoam Utilities

김병윤
대표

Open Source CFD Consulting

NEXT*f*oam
153-790, 서울특별시 금천구 가산동 갑을그레이트밸리 A동 1106호

February 2014

차 례

1 개요

공통 옵션

-case <dir>	작업 폴더의 경로를 지정, 디폴트는 cwd
-noFunctionObjects	functionObject를 실행하지 않음
-srcDoc	소스코드를 보여줌
-doc	application documentation을 보여줌
-help	사용법을 터미널 화면에 보여줌 "
-region <name>	지정한 region에만 적용
-parallel	병렬로 작동한다.
-overwrite	기존 격자 데이터에 덮어쓴다.
-roots <(dir1 .. dirN)>	slave root directories for distributed running
-constant	times list에 constant 폴더를 포함한다.
-time <ranges>	comma-separated time ranges - eg, ':10,20,40:70,1000:'
-latestTime	마지막 시간만 포함한다.
-noZero	times list에 0 폴더를 제외한다.
-dict <file>	system 폴더의 컨트롤 딕셔너리 파일을 지정한다.

2 Mesh - conversion

2.1 ansysToFoam

I-DEAS에서 내보낸 ANSYS 격자 파일을 오픈폼 격자로 변환

2.2 foamMeshToFluent

오픈폼 격자를 Fluent 격자로 변환

2.3 ideasUnvToFoam

I-DEAS unv 형식 격자를 오픈폼 격자로 변환

2.4 plot3dToFoam

기능

Plot3d 격자(ascii/formatted format)를 오픈폼 격자로 변환

사용법

> plot3dToFoam [options] <PLOT3D geom file>

옵션

-scale <factor>	크기의 축소 / 확대 단위 설정, 디폴트는 1
-2D <thickness>	2D 격자를 변환할 때 사용
-noblank	blank item 생략
-singleBlock	single block 격자일 때 사용
-writeZones	Fluent 의 cell zones 을 zones 으로 저장

사용 예

>plot3dToFoam -scale 0.001 -noblank -singleBlock test.x

2.5 vtkUnstructuredToFoam

아스키 vtk(lagacy format) 파일을 오픈폼 격자로 변환

2.6 cfx4ToFoam

CFX4 격자를 오픈폼 격자로 변환

2.7 foamToStarMesh

오픈폼 격자를 PROSTAR(v4) bnd/cel/vrt 형식으로 변환

2.8 kivaToFoam

KIVA 격자를 오픈폼 격자로 변환

2.9 sammToFoam

STAR-CD(v3) SAMM 격자를 오픈폼 격자로 변환

2.10 writeMeshObj

오픈폼 격자를 javaview 에서 볼 수 있게 3 개의 OBJ 파일로 변환 (for mesh debugging)

2.11 datToFoam

datToFoam(.dat) 격자를 point 파일로 변환. blockMesh 와 함께 사용

2.12 foamToSurface

오픈폼 격자의 경계면 (boundary) 만 surface format 으로 변환

2.13 mshToFoam

Adventure system 에서 만든 .msh 파일을 오픈폼 격자로 변환

2.14 ccm26ToFoam

Star-CCM+ 격자를 오픈폼 격자로 변환

2.15 star3ToFoam

STAR-CD(v3) PROSTAR 격자를 오픈폼 격자로 변환

2.16 star4ToFoam

STAR-CD(v4) PROSTAR 격자를 오픈폼 격자로 변환

2.17 fluentMeshToFoam

기능

Fluent 격자를 오픈폼 격자로 변환한다. writeZones, writeSets 옵션을 통해 Fluent의 region과 경계면을 zone이나 set으로 변환할 수 있다. Fluent의 internal(interior) 면과 baffle은 변환되지 않는다. MRF나 porous 등을 계산할 때 writeZones 옵션을 사용한다.

사용법

> fluentMeshToFoam [options] <Fluent mesh/cas file>

옵션

-scale <factor>	크기의 축소/확대 단위 설정, 디폴트는 1
-writeSets	Fluent의 cell zones과 경계면을 sets으로 저장
-writeZones	Fluent의 cell zones을 zones으로 저장

사용 예

>fluentMeeshToFoam -scale 0.001 -writeZones -writeSets test.cas

2.18 fluent3DMeshToFoam

Fluent 격자를 오픈폼 격자로 변환한다. 디폴트로 writeZones 옵션이 포함된다.

기능

Fluent 격자를 오픈폼 격자로 변환한다.

사용법

> fluent3DMeshToFoam [-options] <Fluent mesh/cas file>

옵션

-scale <factor>	크기의 축소 / 확대 단위 설정, 디폴트는 1
-cubit	special parsing of (incorrect) cubit files
-ignoreCellGroups <names>	지정된 cell group을 무시함
-ignoreFaceGroups <names>	지정된 face group을 무시함

사용 예

>fluent3DMeeshToFoam -scale 0.001 test.msh

2.19 gambitToFoam

GAMBIT 격자를 오픈폼 격자로 변환

2.20 netgenNeutralToFoam

Netgen v4.4에서 만든 neutral 파일을 오픈폼 격자로 변환

2.21 gmshToFoam

Gmsh에서 만든 .msh 파일을 오픈폼 격자로 변환

2.22 tetgenToFoam

tetgen에서 만든 .els, .node, .face 파일을 오픈폼 파일로 변환

3 Mesh - generation

3.1 blockMesh

multi-block mesh generator

3.2 extrudeMesh

기능

Extrude mesh from existing patch (by default outwards facing normals;optional flips faces) or from patch read from file.
Note: Merges close points so be careful.
Type of extrusion prescribed by run-time selectable model.

사용법

> extrudeMesh [-options]

사용 예

> extrudeMesh

Example of createPatchDict - of-2.2.x

```
// What to extrude:
//      patch   : from patch of another case ('sourceCase')
//      mesh    : as above but with original case included
//      surface : from externally read surface
constructFrom patch;//mesh;//surface;

// If construct from patch/mesh:
sourceCase "../cavity";
sourcePatches (movingWall);
// If construct from patch: patch to use for back (can be same as sourcePatch)
exposedPatchName movingWall;
// If construct from surface:
surface "movingWall.stl";

// Flip surface normals before usage. Valid only for extrude from surface or
    patch.
flipNormals false;

//- Linear extrusion in point-normal direction
//extrudeModel        linearNormal;

//- Linear extrusion in specified direction
//extrudeModel        linearDirection;

//- Wedge extrusion. If nLayers is 1 assumes symmetry around plane.
extrudeModel        wedge;

//- Extrudes into sphere around (0 0 0)
//extrudeModel        linearRadial;

//- Extrudes into sphere around (0 0 0) with specified radii
//extrudeModel        radial;

//- Extrudes into sphere with grading according to pressure (atmospherics)
//extrudeModel        sigmaRadial;

nLayers             10;

expansionRatio      1.0;    //0.9;

wedgeCoeffs
{
    axisPt      (0 0.1 -0.05);
    axis        (-1 0 0);
    angle       360;  // For nLayers=1 assume symmetry so angle/2 on each side
```

```
}

linearNormalCoeffs
{
    thickness       0.05;
}

linearDirectionCoeffs
{
    direction       (0 1 0);
    thickness       0.05;
}

linearRadialCoeffs
{
    R               0.1;
    // Optional inner radius
    Rsurface        0.01;
}

radialCoeffs
{
    // Radii specified through interpolation table
    R               table ((0 0.01)(3 0.03)(10 0.1));
}

sigmaRadialCoeffs
{
    RTbyg           1;
    pRef            1;
    pStrat          1;
}

// Do front and back need to be merged? Usually only makes sense for 360 degree
    wedges.
mergeFaces false;   //true;

// Merge small edges. Fraction of bounding box.
mergeTol 0;
```

3.3 extrudeToRegionMesh

기능

Extrude faceZones (internal or boundary faces) or faceSets (boundary faces only) into a separate mesh (as a different region).

- used to e.g. extrude baffles (extrude internal faces) or create liquid film regions.

- if extruding internal faces: create baffles in original mesh with mappedWall patches
- if extruding boundary faces: convert boundary faces to mappedWall patches
- extrude edges of faceZone as a <zone>_sidePatch
- extrude edges inbetween different faceZones as a (nonuniformTransform)cyclic <zoneA>_<zoneB>
- extrudes into master direction (i.e. away from the owner cell if flipMap is false)

```
Internal face extrusion
-----------------------

    +-------------+
    |             |
    |             |
    +---AAAAAAA---+
    |             |
    |             |
    +-------------+

    AAA=faceZone to extrude.


For the case of no flipMap the extrusion starts at owner and extrudes
into the space of the neighbour:

      +CCCCCCC+
      |       |         <= extruded mesh
      +BBBBBBB+

    +-------------+
    |             |
    | (neighbour) |
    |___CCCCCCC___|       <= original mesh (with 'baffles' added)
    |   BBBBBBB   |
    |(owner side) |
    |             |
    +-------------+

    BBB=mapped between owner on original mesh and new extrusion.
        (zero offset)
    CCC=mapped between neighbour on original mesh and new extrusion
        (offset due to the thickness of the extruded mesh)

For the case of flipMap the extrusion is the other way around: from the
```

```
neighbour side into the owner side.


Boundary face extrusion
-----------------------

    +--AAAAAAA--+
    |           |
    |           |
    +-----------+

    AAA=faceZone to extrude. E.g. slave side is owner side (no flipmap)

becomes

      +CCCCCCC+
      |       |          <= extruded mesh
      +BBBBBBB+

    +--BBBBBBB--+
    |           |        <= original mesh
    |           |
    +-----------+

    BBB=mapped between original mesh and new extrusion
    CCC=polypatch


Notes:
    - when extruding cyclics with only one cell inbetween it does not
      detect this as a cyclic since the face is the same face. It will
      only work if the coupled edge extrudes a different face so if there
      are more than 1 cell inbetween.
```

사용법

> extrudeToRegionMesh [-options]

사용 예

> extrudeRegionMesh

3.4 extrude2DMesh

기능

Takes 2D mesh (all faces 2 points only, no front and back faces) and creates a 3D mesh by extruding with specified thickness.

사용법

> extrudeToRegionMesh [-options] <surfaceFormat>

extrude2DMeshDict 파일을 사용하며 surfaceFormat 은 : MeshedSurface 와 polyMesh2D 가 있다.

사용 예

> extrude2DMesh -overwrite MeshedSurface (of23 tut. foamyQuadMesh)

Example of extrude2DMeshDict - of-2.3.x

```
extrudeModel        linearDirection;//radial;//wedge;

patchType           empty;//wedge;

nLayers             1;

expansionRatio      1.0;

linearDirectionCoeffs
{
    direction       (0 0 1);
    thickness       0.1;
}

wedgeCoeffs
{
    axisPt      (0 0 0);
    axis        (1 0 0);
    angle       10;
}
```

3.5 patchToPoly2DMesh

???

3.6 foamyHexMesh

3 차원 격자 생성 유틸리티

3.7 foamyQuadMesh

2 차원 격자 생성 유틸리티

3.8 snappyHexMesh

3 차원 격자 생성 유틸리티

4 Mesh - manipulation

4.1 attachMesh

기능

Attach topologically detached mesh using prescribed mesh modifiers.

사용법

> attachMesh [-options]

사용 예

> attachMesh

4.2 autoPatch

기능

Divides external faces into patches based on (user supplied) feature angle.

사용법

> autoPatch [-options] <feature angle[0-180]>

사용 예

> autoPatch -overwrite 89

4.3 checkMesh

기능

checks validity of a mesh

사용법

> checkMesh [options]

옵션

- allGeometry : include bounding box checks
- allTopology : include extra topology checks

- meshQuality : read user-defined mesh quality criterions from system/meshQualityDict

- noTopology : skip checking the mesh topoloty

사용 예

>checkMesh

4.4 createBaffles

기능

Makes internal faces into boundary faces. Does not duplicate points, unlike mergeOrSplitBaffles.

Note

if any coupled patch face is selected for baffling the opposite member has to be selected for baffling as well.

- if the patch already exists will not override it nor its fields
- if the patch does not exist it will be created together with 'calculated' patchfields unless the field is mentioned in the patchFields section.

사용법

> createBaffles [options]

사용 예

> createBaffles

Example of createBafflesDict - of-2.3.x

```
// Sample for creating baffles:
// - usually converting internal faces into two boundary faces
// - or converting boundary faces into a boundary face
//   (internalFacesOnly=false)(though should use really createPatch to do this)
// - specification in one of two modes:
//      - patchPairs : create two patches of same type, same input
//      - patches    : create patches separately, full control over what
//                  to create on what side
//                      (this mode can also create duplicate (overlapping)
//                       sets of baffles:
//                       - internalFacesOnly = false
//                       - have 4 entries in patches:
//                           - master
//                           - slave
//                           - additional master
```

```
//                          - additional slave)


// Whether to convert internal faces only (so leave boundary faces intact).
// This is only relevant if your face selection type can pick up boundary faces.
internalFacesOnly true;

// Optionally do not read/convert/write any fields.
//noFields true;

// Baffles to create.
baffles
{
    baffle1
    {
        //- Use surface to select faces and orientation.
        type        searchableSurface;
        surface     triSurfaceMesh;
        name        baffle1D.stl;
        //- Optional flip
        //flip        false;


        // Generate patchGroup baffle1 with two patches:
        //  - baffle1_master
        //  - baffle1_slave
        patchPairs
        {
            type            wall;
            //- Optional override of added patchfields. If not specified
            //  any added patchfields are of type calculated.
            patchFields
            {
                U
                {
                    type            fixedValue;
                    value           uniform (0 0 0);
                }
            }
        }
    }

    cyclicFaces
    {
        //- Select faces and orientation through a searchableSurface
        type        searchableSurface;
        surface     searchablePlate;

        origin      (0.099 -0.006 0.004);
        span        (0 0.012 0.012);
```

```
        // Generate patches explicitly
        patches
        {
            master
            {
                //- Master side patch

                name            fan_half0;
                type            cyclic;
                neighbourPatch  fan_half1;

                patchFields
                {
                    p
                    {
                        type            fan;
                        patchType       cyclic;
                        jump            uniform 0;
                        value           uniform 0;
                        jumpTable       polynomial 1((100 0));
                    }
                }
            }
            slave
            {
                //- Slave side patch

                name            fan_half1;
                type            cyclic;
                neighbourPatch  fan_half0;

                patchFields
                {
                    p
                    {
                        type            fan;
                        patchType       cyclic;
                        value           uniform 0;
                    }
                }
            }
        }
    }
}
```

4.5 createPatch

기능

This application/dictionary controls:
- optional: create new patches from boundary faces (either given as a set of patches or as a faceSet)
- always: order faces on coupled patches such that they are opposite. This is done for all coupled faces, not just for any patches created.
- optional: synchronise points on coupled patches.

1. Create cyclic:
- specify where the faces should come from
- specify the type of cyclic. If a rotational specify the rotationAxis and centre to make matching easier
- always create both halves in one invocation with correct 'neighbourPatch' setting.
- optionally pointSync true to guarantee points to line up.

2. Correct incorrect cyclic:
This will usually fail upon loading:
"face 0 area does not match neighbour 2 by 0.0100005%"
" – possible face ordering problem."
- in polyMesh/boundary file:
- loosen matchTolerance of all cyclics to get case to load
- or change patch type from 'cyclic' to 'patch' and regenerate cyclic as above

사용법

> createPatch [options]

사용 예

> createPatch

Example of createPatchDict - of-2.3.x

```
// This application/dictionary controls:
// - optional: create new patches from boundary faces (either given as
//   a set of patches or as a faceSet)
// - always: order faces on coupled patches such that they are opposite. This
//   is done for all coupled faces, not just for any patches created.
// - optional: synchronise points on coupled patches.
// - always: remove zero-sized (non-coupled) patches (that were not added)

// 1. Create cyclic:
// - specify where the faces should come from
// - specify the type of cyclic. If a rotational specify the rotationAxis
```

```
//   and centre to make matching easier
// - always create both halves in one invocation with correct 'neighbourPatch'
//   setting.
// - optionally pointSync true to guarantee points to line up.

// 2. Correct incorrect cyclic:
// This will usually fail upon loading:
//  "face 0 area does not match neighbour 2 by 0.0100005%"
//  " -- possible face ordering problem."
// - in polyMesh/boundary file:
//      - loosen matchTolerance of all cyclics to get case to load
//      - or change patch type from 'cyclic' to 'patch'
//        and regenerate cyclic as above

// Do a synchronisation of coupled points after creation of any patches.
// Note: this does not work with points that are on multiple coupled patches
//       with transformations (i.e. cyclics).
pointSync false;

// Patches to create.
patches
(
    {
        // Name of new patch
        name cyc_half0;

        // Dictionary to construct new patch from
        patchInfo
        {
            type cyclic;
            neighbourPatch cyc_half1;

            // Optional: explicitly set transformation tensor.
            // Used when matching and synchronising points.
            transform rotational;
            rotationAxis (1 0 0);
            rotationCentre (0 0 0);
            // transform translational;
            // separationVector (1 0 0);

            // Optional non-default tolerance to be able to define cyclics
            // on bad meshes
            //matchTolerance 1E-2;
        }

        // How to construct: either from 'patches' or 'set'
        constructFrom patches;

        // If constructFrom = patches : names of patches. Wildcards allowed.
        patches (periodic1);
```

```
        // If constructFrom = set : name of faceSet
        set f0;
    }
    {
        // Name of new patch
        name cyc_half1;

        // Dictionary to construct new patch from
        patchInfo
        {
            type cyclic;
            neighbourPatch cyc_half0;

            // Optional: explicitly set transformation tensor.
            // Used when matching and synchronising points.
            transform rotational;
            rotationAxis (1 0 0);
            rotationCentre (0 0 0);
            // transform translational;
            // separationVector (1 0 0);
        }

        // How to construct: either from 'patches' or 'set'
        constructFrom patches;

        // If constructFrom = patches : names of patches. Wildcards allowed.
        patches (periodic2);

        // If constructFrom = set : name of faceSet
        set f0;
    }
);
```

Example of createPatchDict - of-2.2.x

```
pointSync false;
patches
(
    {
        name  inlet;
        patchInfo
        {
             type patch;
        }
        constructFrom set;
        patches ("perioidc.*");
        set f0;
    }
);
```

Example of createPatchDict - of-1.6-ext

```
matchTolerance  1e-3;
pointSync true;

patchInfo
(
    {
        name  airfoil;
        dictionary
        {
            type  wall;
        }
        constructFrom patches;
        patches ( wall_top wall_bottom);
  }
);
```

4.6 deformedGeom

기능

Deforms a polyMesh using a displacement field U and a scaling factor supplied as an argument.

사용법

> deformedGeom [options] <scale factor>

사용 예

> ???

4.7 flattenMesh

기능

Flattens the front and back planes of a 2D cartesian mesh.

사용법

> flattenMesh [options]

사용 예

> ???

4.8 insideCells

기능

Picks up cells with cell centre 'inside' of surface.
Requires surface to be closed and singly connected.

사용법

> insideCells [options] <surfaceFile> <cellSet>

사용 예

> ???

4.9 mergeMeshes

기능

merges two meshes

사용법

> mergeMeshes [options] <masterCase> <addCase>

옵션

- addRegion <name> : specify alternative mesh region for the additional mesh
- masterResion <name> : specify alternative mesh region for the master mesh

사용 예

> ???

4.10 mergeOrSplitBaffles

기능

Detects faces that share points (baffles). Either merge them or duplicate the points.

Notes

- can only handle pairwise boundary faces. So three faces using the same points is not handled (is illegal mesh anyway)
- there is no option to only split/merge some baffles.
- surfaces consisting of duplicate faces can be topologically split if the points on the interior of the surface cannot walk to all the cells that use them in one go.

- Parallel operation (where duplicate face is perpendicular to a coupled boundary) is supported but not really tested. (Note that coupled faces themselves are not seen as duplicate faces)

사용법

> mergeOrSplitBaffles [options]

옵션

- detectOnly : find baffles only, but do not merge or split them
- split : topologically split duplicate surfaces

사용 예

> ???

4.11 mirrorMesh

기능

격자를 mirror 시킨다. system/mirrorMeshDict 파일에서 대칭면을 설정한다.

사용법

> mirrorMesh [options]

옵션

-overwrite	기존 격자 데이터에 덮어쓴다.
-parallel	병렬로 작동한다.

사용 예

>mirrorMesh -overwrite

Example of mirrorMeshDict

```
planeType           pointAndNormal;
pointAndNormalDict
{
    basePoint       (0 0 0);
    normalVector    (0 0 1);
}
planeTolerance      1e-3;
```

4.12 moveDynamicMesh

기능

Mesh motion and topological mesh change utility.

사용법

> moveDynamicMesh [options]

옵션

- checkAMI : check AMI weights

사용 예

> moveDynamicMesh

4.13 moveEngineMesh

기능

Solver for moving meshes for engine caculation.

사용법

> moveEngineMesh [options]

사용 예

> ???

4.14 moveMesh

기능

Solver for moving meshes.

사용법

> moveMesh [options]

사용 예

> ???

4.15 objToVTK

기능

Read obj line (not surface) file and convert into vtk.

사용법

> objToVTK [options] <OBJ file> <output VTK file>

사용 예

> ???

4.16 orientFaceZone

기능

Corrects orientation of faceZone.

- correct in parallel - excludes coupled faceZones from walk
- correct for non-manifold faceZones - restarts walk

사용법

> orientFaceZone [options] <faceZone> <outsidePoint>

사용 예

> ???

4.17 polyDualMesh

기능

Calculates the dual of a polyMesh. Adheres to all the feature and patch edges.

사용법

> polyDualMesh [options] <featureAngle [0-180]>

Detects any boundary edge ¿ angle and creates multiple boundary faces for it. Normal behaviour is to have each point become a cell (1.5 behaviour)

옵션

- concaveMultiCells : split cells on concave boundary edges into multiple cells

  Creates multiple cells for each point on a concave edge. Might limit the amount of distortion on some meshes.

- doNotPreserveFaceZones : disable the default behaviour of preserving faceZones by having multiple faces inbetween cells

- splitAllFaces : have multiple faces inbetween cells

  Normally only constructs a single face between two cells. This single face might be too distorted. splitAllFaces will create a single face for every original cell the face passes

through. The mesh will thus have multiple faces inbetween two cells! (so is not strictly upper-triangular anymore - checkMesh will complain)

사용 예

> ???

4.18 refineMesh

기능

Utility to refine cells in multiple directions.

Either supply -all option to refine all cells (3D refinement for 3D cases; 2D for 2D cases) or reads a refineMeshDict with

- cellSet to refine
- directions to refine

사용법

> refineMesh [options]

사용 예

> refineMesh : LTSInterFoam tutorial - DTCHull

Example of mirrorMeshDict

```
// Cells to refine; name of cell set
set c0;

// Type of coordinate system:
// - global : coordinate system same for every cell. Usually aligned with
//   x,y,z axis. Specify in globalCoeffs section below.
// - patchLocal : coordinate system different for every cell. Specify in
//   patchLocalCoeffs section below.
coordinateSystem global;
//coordinateSystem patchLocal;

// .. and its coefficients. x,y in this case. (normal direction is calculated
// as tan1^tan2)
globalCoeffs
{
    tan1 (1 0 0);
    tan2 (0 1 0);
}

patchLocalCoeffs
{
```

```
        patch outside;  // Normal direction is facenormal of zero'th face of patch
        tan1 (1 0 0);
    }

    // List of directions to refine
    directions
    (
        tan1
        tan2
        normal
    );

    // Whether to use hex topology. This will
    // - if patchLocal: all cells on selected patch should be hex
    // - split all hexes in 2x2x2 through the middle of edges.
    useHexTopology  true;

    // Cut purely geometric (will cut hexes through vertices) or take topology
    // into account. Incompatible with useHexTopology
    geometricCut    false;

    // Write meshes from intermediate steps
    writeMesh       false;
```

4.19 renumberMesh

기능

Renumbers the cell list in order to reduce the bandwidth, reading and renumbering all fields from all the time directories.
By default uses bandCompression (CuthillMcKee) but will read system/renumberMeshDict if -dict option is present

사용법

> renumberMesh [options]

옵션

- frontWidth : calculate the rms of the frontwidth

사용 예

> renumberMesh

Example of mirrorMeshDict

```
// Write maps from renumbered back to original mesh
writeMaps true;
```

```
// Optional entry: sort cells on coupled boundaries to last for use with
// e.g. nonBlockingGaussSeidel.
sortCoupledFaceCells false;

// Optional entry: renumber on a block-by-block basis. It uses a
// blockCoeffs dictionary to construct a decompositionMethod to do
// a block subdivision) and then applies the renumberMethod to each
// block in turn. This can be used in large cases to keep the blocks
// fitting in cache with all the the cache misses bunched at the end.
// This number is the approximate size of the blocks - this gets converted
// to a number of blocks that is the input to the decomposition method.
//blockSize 1000;

// Optional entry: sort points into internal and boundary points
//orderPoints false;

method          CuthillMcKee;
//method          Sloan;//manual;//random;//structured;//spring;
//method          zoltan;             // only if compiled with zoltan support

//CuthillMcKeeCoeffs
//{
//    // Reverse CuthillMcKee (RCM) or plain
//    reverse true;
//}

manualCoeffs
{
    // In system directory: new-to-original (i.e. order) labelIOList
    dataFile "cellMap";
}

// For extruded (i.e. structured in one direction) meshes
structuredCoeffs
{
    // Patches that mesh was extruded from. These determine the starting
    // layer of cells
    patches (movingWall);
    // Method to renumber the starting layer of cells
    method  random;

    // Renumber in columns (depthFirst) or in layers
    depthFirst true;

    // Reverse ordering
    reverse false;
}

springCoeffs
{
```

```
    // Maximum jump of cell indices. Is fraction of number of cells
    maxCo 0.01;

    // Limit the amount of movement; the fraction maxCo gets decreased
    // with every iteration
    freezeFraction 0.999;

    // Maximum number of iterations
    maxIter 1000;
}

blockCoeffs
{
    method          scotch;
    //method hierarchical;
    //hierarchicalCoeffs
    //{
    //    n           (1 2 1);
    //    delta       0.001;
    //    order       xyz;
    //}
}

zoltanCoeffs
{
    ORDER_METHOD    LOCAL_HSFC;
}
```

4.20 rotateMesh

기능

Rotates the mesh and fields from the direcion n1 to the direction n2.

사용법

> rotateMesh [options] <n1> <n2>

사용 예

> ???

4.21 setSet

기능

Manipulate a cell/face/point/ set or zone interactively.

사용법

> setSet [options]

옵션

- batch <file> : process in batch mode, using input from specified file
- loop : execute batch commands for all timesteps
- noSync : do not synchronise selection across coupled patches
- noVTK : do not write VTK files
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option

사용 예

> setSet -batch testBatch

4.22 setsToZones

기능

Add pointZones/faceZones/cellZones to the mesh from similar named pointSets/faceSets/-cellSets.

There is one catch: for faceZones you also need to specify a flip condition which basically denotes the side of the face. In this app it reads a cellSet (xxxCells if 'xxx' is the name of the faceSet) which is the masterCells of the zone.
There are lots of situations in which this will go wrong but it is the best I can think of for now.

If one is not interested in sideNess specify the -noFlipMap command line option.

사용법

> setsToZones [options]

옵션

- noFlipMap : ignore orientation of faceSet
- noZero : exclude the '0/' dir from the times list

사용 예

> setsToZones -noFlipMap

4.23 singleCellMesh

기능

Reads all fields and maps them to a mesh with all internal faces removed (singleCellFvMesh) which gets written to region "singleCell".

Used to generate mesh and fields that can be used for boundary-only data.
Might easily result in illegal mesh though so only look at boundaries in paraview.

사용법

> singleCellMesh [options]

사용 예

> ???

4.24 splitMesh

기능

Splits mesh by making internal faces external. Uses attachDetach.

Generates a meshModifier of the form:

```
Splitter
{
    type                        attachDetach;
    faceZoneName                membraneFaces;
    masterPatchName             masterPatch;
    slavePatchName              slavePatch;
    triggerTimes                runTime.value();
}
```

so will detach at the current time and split all faces in membraneFaces into masterPatch and slavePatch (which have to be present but of 0 size)

사용법

> splitMesh [options] <faceSet> <masterPatch> <slavePatch>

사용 예

> ???

4.25 splitMeshRegions

기능

Splits mesh into multiple regions.

Each region is defined as a domain whose cells can all be reached by cell-face-cell walking without crossing

- boundary faces
- additional faces from faceset (-blockedFaces faceSet).
- any face inbetween differing cellZones (-cellZones)

Output is:

- volScalarField with regions as different scalars (-detectOnly) or
- mesh with multiple regions and mapped patches. These patches either cover the whole interface between two region (default) or only part according to faceZones (-useFaceZones) or
- mesh with cells put into cellZones (-makeCellZones)

Note

- cellZonesOnly does not do a walk and uses the cellZones only. Use this if you don't mind having disconnected domains in a single region. This option requires all cells to be in one (and one only) cellZone.
- cellZonesFileOnly behaves like -cellZonesOnly but reads the cellZones from the specified file. This allows one to explicitly specify the region distribution and still have multiple cellZones per region.
- useCellZonesOnly does not do a walk and uses the cellZones only. Use this if you don't mind having disconnected domains in a single region. This option requires all cells to be in one (and one only) cellZone.
- prefixRegion prefixes all normal patches with region name (interface patches already have region name prefix)
- Should work in parallel. cellZones can differ on either side of processor boundaries in which case the faces get moved from processor patch to directMapped patch. Not the faces get moved from processor patch to mapped patch. Not very well tested.
- If a cell zone gets split into more than one region it can detect the largest matching region (-sloppyCellZones). This will accept any region that covers more than 50% of the zone. It has to be a subset so cannot have any cells in any other zone.

- If explicitly a single region has been selected (-largestOnly or -insidePoint) its region name will be either

  - name of a cellZone it matches to or

  - "largestOnly" respectively "insidePoint" or

  - polyMesh::defaultRegion if additionally -overwrite(so it will overwrite the input mesh!)

- writes maps like decomposePar back to original mesh:

  - pointRegionAddressing : for every point in this region the point in the original mesh

  - cellRegionAddressing : ,, cell ,, cell ,,

  - faceRegionAddressing : ,, face ,, face in the original mesh + 'turning index'. For a face in the same orientation this is the original facelabel+1, for a turned face this is -facelabel-1

  - boundaryRegionAddressing : for every patch in this region the patch in the original mesh (or -1 if added patch)

사용법

> splitMeshRegions [options]

옵션

- blockedFaces <faceSet> : specify additional region boundaries that walking does not cross
- cellZones : additionally split cellZones off into separate regions
- cellZonesFileOnly <file> : like -cellZonesOnly, but use specified file
- cellZonesOnly : use cellZones only to split mesh into regions; do not use walking
- detectOnly : do not write mesh
- insidePoint <point> : only write region containing point
- largestOnly : only write largest region
- makeCellZones : place cells into cellZones instead of splitting mesh
- prefixRegion : prefix region name to all patches, not just coupling patches
- sloppyCellZones :try to match heuristically regions to existing cell zones
- useFaceZones : use faceZones to patch inter-region faces instead of single patch

사용 예

> ???

4.26 stitchMesh

기능

'Stitches' a mesh.

Takes a mesh and two patches and merges the faces on the two patches (if geometrically possible) so the faces become internal.

Can do

- 'perfect' match: faces and points on patches align exactly. Order might be different though.
- 'integral' match: where the surfaces on both patches exactly match but the individual faces not
- 'partial' match: where the non-overlapping part of the surface remains in the respective patch.

Note

Is just a front-end to perfectInterface/slidingInterface.

Comparable to running a meshModifier of the form (if masterPatch is called "M" and slavePatch "S"):

```
couple
{
    type                    slidingInterface;
    masterFaceZoneName      MSMasterZone
    slaveFaceZoneName       MSSlaveZone
    cutPointZoneName        MSCutPointZone
    cutFaceZoneName         MSCutFaceZone
    masterPatchName         M;
    slavePatchName          S;
    typeOfMatch             partial or integral
}
```

사용법

> stitchMesh [options] <masterPatch> <slavePatch>

옵션

- partial : couple partially overlapping patches
- perfect : couple perfectly aligned patches
- toleranceDict <file> : dictionary file with tolerance

사용 예

> ???

4.27 subsetMesh

기능

Selects a section of mesh based on a cellSet.

The utility sub-sets the mesh to choose only a part of interest. Check the setSet/cellSet/topoSet utilities to see how to select cells based on various shapes.

The mesh will subset all points, faces and cells needed to make a sub-mesh but will not preserve attached boundary types.

사용법

> subsetMesh [options] <cellSet>

옵션

- patch <name> : add exposed internal faces to specified patch instead of to 'oldInternalFaces'
- resultTime <time> : specify a time for the resulting mesh

사용 예

> ???

4.28 topoSet

기능

Operates on cellSets/faceSets/pointSets through a dictionary

사용법

> topoSet [options]

옵션

- noSync : do not synchronise selection across coupled patches
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option

사용 예

> topoSet

4.29 transformPoints

기능

격자를 확대 / 축소, 회전, 이동시킨다.

사용법

> transformPoints [options]

옵션

-region	특정 region을 지정한다.
-overwrite	기존 격자 데이터에 덮어 쓴다.
-parallel	병렬로 작동한다.
-rollPitchYaw <'vector'>	'(roll pitch yaw)' 형식으로 주어진 각도(degree) 만큼 작동한다.
-roots <(dir1 .. dirN)>	slave root directories for distributed running
-rotate <'(vectorA vectorB)'>	주어진 두 벡터만큼 회전한다.
-rotateField	벡터와 텐서 필드 데이터도 회전한다.
-scale <'vector'>	주어진 벡터만큼 x,y,z 방향으로 확대 / 축소한다.
-translate <'vector'>	주어진 벡터만큼 이동한다.
-yawPitchRoll <'vector'>	transform in terms of '(yaw pitch roll)' in degrees

사용 예

> transformPoints -overwrite -scale "(0.001 0.001 0.001)"

> transformPoints -overwrite -rotateAlongVector "(0 0 1) 90"

(this is for of-1.6-ext, not sure for of-2.2.x)

4.30 zipUpMesh

기능

Reads in a mesh with hanging vertices and zips up the cells to guarantee that all polyhedral cells of valid shape are closed.

Meshes with hanging vertices may occur as a result of split hex mesh conversion or integration or coupled math face pairs.

사용법

> zipUpMesh [options]

사용 예

> ???

5 Mesh - advanced

5.1 autoRefineMesh

기능

Utility to refine cells near to a surface.

사용법

> autoRefineMesh [options]

사용 예

> ???

Example of autoRefineMeshDict

```
// Surface to keep to
surface         "plexi.obj";

// What is outside. These points have to be inside a cell (so not on a face!)
outsidePoints   ((-0.99001 -0.99001 -0.99001));

//
// Selection of cells to refine
//

// If smallest edge of mesh > maxEdgeLen select all cut cells for refinement.
// If < maxEdgeLen select only those cut cells which are closer than
// curvatureDistance to surface
// and with cos of angle between normals on surface < curvature.
maxEdgeLen          0.1;
curvatureDistance   1.0;
curvature           0.9;

// if > 0: Remove inside cells at every step. Inside is given by number of
// layers separating outside from inside.
// (note that we cannot remove outside
// cells since these contain the outsidePoints)
// Do not use this option if you want mesh to spill through a hole which is
// not visible on the coarsest level but only becomes visible after refinement
nCutLayers 2;

// Refine until smallest edge of mesh < minEdgeLen
minEdgeLen      0.1;

// Or until the number of cells would become more than (stops one level before
// this)
cellLimit       2500000;
```

```
//
// Selection of final set
//

// Select based on side of surface. Usually select inside cells and project
// outwards or select outside cells and project inwards.
selectCut       false;
selectInside    false;
selectOutside   true;
// Leave out cell closer than nearDistance to the surface. Usually
// 0.5*minEdgeLen. Set to -1 to disable.
nearDistance    -1;

// Some cells on the surface of the selected cells might have all their
// points on the 'outside'. These would get flattened when projecting so
// are either kept and refined (selectHanging) or removed from the set
selectHanging   false;

//
// Refinement parameters
//

// Type of coordinate system
coordinateSystem global;
//coordinateSystem patchLocal;

// .. and its coefficients. x,y in this case. (normal = tan1^tan2)
globalCoeffs
{
    tan1 (1 0 0);
    tan2 (0 1 0);
}

patchLocalCoeffs
{
    patch outside;  // Normal direction is facenormal of zero'th face of patch
    tan1 (1 0 0);

}

// List of directions to refine
directions
(
    tan1
    tan2
    normal
);

// refinement level difference between neighbouring cells. Set to large if
// there is no need for a limit.
```

```
splitLevel      2;

// Cut purely geometric (will cut hexes through vertices) or take topology
// into account.
geometricCut    false;

// Whether to use hex topology. This will never cut hex through vertices.
useHexTopology  yes;

// Write meshes from intermediate steps
writeMesh       true;
```

5.2 collapseEdges

기능

Collapses short edges and combines edges that are in line.

- collapse short edges. Length of edges to collapse provided as argument.
- merge two edges if they are in line. Maximum angle provided as argument.
- remove unused points.
- collapse faces:
  - with small areas to a single point
  - that have a high aspect ratio (i.e. sliver face) to a single edge

Optionally checks the resulting mesh for bad faces and reduces the desired face length factor for those faces attached to the bad faces.

When collapsing an edge with one point on the boundary it will leave the boundary point intact. When both points inside it chooses random. When both points on boundary random again.

사용법

> collapseEdges [options]

옵션

- collapseFaceSet <faceSet> : Collapse faces that are in the supplied face set
- collapseFaces : Collapse small and sliver faces as well as small edges

사용 예

> ???

5.3 combinePatchFaces

기능

Checks for multiple patch faces on same cell and combines them.
Multiple patch faces can result from e.g. removal of refined neighbouring cells, leaving 4 exposed faces with same owner.

Rules for merging:

- only boundary faces (since multiple internal faces between two cells not allowed anyway)
- faces have to have same owner
- faces have to be connected via edge which are not features (so angle between them < feature angle)
- outside of faces has to be single loop
- outside of face should not be (or just slightly) concave (so angle between consecutive edges < concaveangle

E.g. to allow all faces on same patch to be merged:

```
combinePatchFaces 180 -concaveAngle 90
```

사용법

> combinePatchFaces [options] <featureAngle [0..180]>

옵션

- concaveAngle <degrees> : specify concave angle [0..180] (default: 30 degrees)
- meshQuality : read user-defined mesh quality criterions from system/meshQualityDict

사용 예

> ???

Example of meshQualityDict

```
//- Maximum non-orthogonality allowed. Set to 180 to disable.
maxNonOrtho         65;

//- Max skewness allowed. Set to <0 to disable.
maxBoundarySkewness 50;

//- Max skewness allowed. Set to <0 to disable.
maxInternalSkewness 10;
```

```
//- Max concaveness allowed. Is angle (in degrees) below which concavity
//  is allowed. 0 is straight face, <0 would be convex face.
//  Set to 180 to disable.
maxConcave          80;

//- Minimum pyramid volume. Is absolute volume of cell pyramid.
//  Set to a sensible fraction of the smallest cell volume expected.
//  Set to very negative number (e.g. -1E30) to disable.
minVol              1e-20;

//- Minimum quality of the tet formed by the face-centre
//  and variable base point minimum decomposition triangles and
//  the cell centre. This has to be a positive number for tracking
//  to work. Set to very negative number (e.g. -1E30) to
//  disable.
//     <0 = inside out tet,
//      0 = flat tet
//      1 = regular tet
minTetQuality       1e-30;

//- Minimum face area. Set to <0 to disable.
minArea             -1;

//- Minimum face twist. Set to <-1 to disable. dot product of face normal
//- and face centre triangles normal
minTwist            0.0;

//- minimum normalised cell determinant
//- 1 = hex, <= 0 = folded or flattened illegal cell
minDeterminant      0.001;

//- minFaceWeight (0 -> 0.5)
minFaceWeight       0.02;

//- minVolRatio (0 -> 1)
minVolRatio         0.01;

//must be >0 for Fluent compatibility
minTriangleTwist    -1;
```

5.4 modifyMesh

기능

Manipulates mesh elements.

```
Actions are:
    (boundary)points:
        - move
```

```
(boundary)edges:
    - split and move introduced point

(boundary)faces:
    - split(triangulate) and move introduced point

edges:
    - collapse

cells:
    - split into polygonal base pyramids around newly
     introduced mid point
```

Is a bit of a loose collection of mesh change drivers.

사용법

> modifyMesh [options]

사용 예

> ???

Example of modifyMeshDict

```
// Move boundary points:
// Every entry is two coordinates. First one is location of the point to move,
// the second is the position to move to.
pointsToMove
(
    (( -0.17861 -0.45073 0.75276)( -0.18 -0.45073 0.75276))
);

// Split boundary edge in two:
// First coord is a point on the (boundary) edge to cut, second is the
// position of the newly introduced point
edgesToSplit
(
    (( -0.17692 -0.45312 0.74516)( -0.18 -0.45 0.742))
);

// Triangulate a boundary face:
// First coord is a point on the face to triangulate. It will introduce a
// point on the face, triangulate and move the point to the second coordinate.
facesToTriangulate
(
    (( -0.039123 -0.45045 0.74083) (-0.03844 -0.45049 0.73572))
);
```

```
// Boundary edges to collapse. First coord is point on the edge, second
// is coordinate to collapse to.
edgesToCollapse
(
    ((0.054975 0.099987 0.0044074)(0.054975 0.099987 0.0044074))
);

// Split cells:
// First coord is a point inside the cell to split. A point inside the cell will
// be introduced and the cell will get decomposed into polygonal base pyramids
// with this new point as top. (so the original faces will not get split)
cellsToSplit
(
    (( -0.039123 -0.45045 0.74083) (-0.03844 -0.45049 0.73572))
);

// Change patch:
// Changes patchID of boundary faces. Coord selects the face, label is the
// patch index.
facesToRepatch
(
    (( -0.039123 -0.45045 0.74083) 1)
);

//// Create cell:
//// Creates a cell on the boundary given a face covering a cavity. Gets
//// the vertices of the face (outwards pointing normal) and a point internal
//// to the new cell. (used to check the orientation of the face). Walks all
//// boundary faces reachable from any edge on the face and constructs cell
//// from it.
//cellsToCreate
//(
//    (
//        ((0 0 0) (1 0 0) (1 1 0) (0 1 0))   // vertices of face
//        (0.5 0.5 0.1)                       // cell centre
//    )
//);
```

5.5 PDRMesh

기능

Mesh and field preparation utility for PDR type simulations.

Reads

- cellSet giving blockedCells
- faceSets giving blockedFaces and the patch they should go into

NOTE

To avoid exposing wrong fields values faceSets should include faces contained in the blockedCells cellset.

- coupledFaces reads coupledFacesSet to introduces mixe-coupled baffles

Subsets out the blocked cells and splits the blockedFaces and updates fields.

The face splitting is done by duplicating the faces. No duplication of points for now (so checkMesh will show a lot of 'duplicate face' messages)

사용법

> PDRMesh [options]

사용 예

> ???

Example of modifyMeshDict

```
//- Per faceSet the patch the faces should go into blocked baffles
blockedFaces ((blockedFacesSet blockedFaces));

//- Per faceSet the patch the faces should go into coupled baffles
coupledFaces
{
    coupledFacesSet
    {
        wallPatchName               baffleWall;
        cyclicMasterPatchName       baffleCyclic_half0;
    }
}

//- Name of cellSet that holds the cells to fully remove
blockedCells blockedCellsSet;

//- All exposed faces that are not specified in blockedFaces go into
//  this patch
defaultPatch outer;
```

5.6 refineHexMesh

기능

Refine a hex mesh by 2x2x2 cell splitting.

사용법

> refineHexMesh [options] <cellSet>

사용 예

> ???

5.7 refinementLevel

기능

Tries to figure out what the refinement level is on refined cartesian meshes. Run BEFORE snapping.

Writes

- volScalarField 'refinementLevel' with current refinement level.
- cellSet 'refCells' which are the cells that need to be refined to satisfy 2:1 refinement.

Works by dividing cells into volume bins.

사용법

> refinementLevel [options]

사용 예

> ???

5.8 refineWallLayer

기능

경계층 격자의 첫번째 cell을 주어진 비율에 따라 높이 방향으로 두 개로 나누어 준다.

사용법

>refineWallLayer [OPTIONS] <patchName> <edgeWeight>

옵션

-overwrite	기존 격자 데이터에 덮어쓴다.
-useSet <name>	지정된 cellSet 에서만 적용한다.

사용 예

>refineWallLayer -overwrite wing 0.5

5.9 removeFaces

기능

Utility to remove faces (combines cells on both sides).

Takes faceSet of candidates for removal and writes faceSet with faces that will actually be removed. (because e.g. would cause two faces between the same cells). See removeFaces in dynamicMesh library for constraints.

사용법

>removeFaces [OPTIONS] <faceSet>

사용 예

> ???

5.10 selectCells

기능

Select cells in relation to surface.

Divides cells into three sets:

- cutCells : cells cut by surface or close to surface.
- outside : cells not in cutCells and reachable from set of user-defined points (outsidePoints)
- inside : same but not reachable.

Finally the wanted sets are combined into a cellSet 'selected'. Apart from straightforward adding the contents there are a few extra rules to make sure that the surface of the 'outside' of the mesh is singly connected.

사용법

>selectCells [OPTIONS]

사용 예

> ???

5.11 splitCells

기능

Utility to split cells with flat faces.

Uses a geometric cut with a plane dividing the edge angle into two so might produce funny cells. For hexes it will use by default a cut from edge onto opposite edge (i.e. purely topological).

Options:

- split cells from cellSet only
- use geometric cut for hexes as well

The angle is the angle between two faces sharing an edge as seen from inside each cell. So a cube will have all angles 90. If you want to split cells with cell centre outside use e.g. angle 200

사용법

>splitCells [options] <edgeAngle [0..360]>

옵션

- geometry : use geometric cut for hexes as well
- set <name> : split cells from specified cellSet only
- tol <scalar> : edge snap tolerance (default 0.2)

사용 예

> ???

6 preprocessing

6.1 applyBoundaryLayer

기능

Apply a simplified boundary-layer model to the velocity and turbulence fields based on the 1/7th power-law.

The uniform boundary-layer thickness is either provided via the -ybl option or calculated as the average of the distance to the wall scaled with the thickness coefficient supplied via the option -Cbl. If both options are provided -ybl is used.

사용법

> applyBoundaryLayer [options]

옵션

- Cbl <scalar> : boundary-layer thickness as Cbl * mean distance to wall
- writenut : write nut field
- ybl <scalar> : specify the boundary-layer thickness

사용 예

> ???

6.2 applyWallFunctionBoundaryConditions

기능

Updates OpenFOAM RAS cases to use the new (v1.6) wall function framework.

Attempts to determine whether case is compressible or incompressible, or can be supplied with -compressible command line argument.

사용법

> applyWallFunctionBoundaryConditions [options]

옵션

- compressible : force use of compressible wall functions. Default is auto-detect.

사용 예

> ???

6.3 boxTurb

기능

Makes a box of turbulence which conforms to a given energy spectrum and is divergence free.

사용법

> boxTurb [options]

사용 예

> ???

6.4 changeDictionary

기능

Utility to change dictionary entries, e.g. can be used to change the patch type in the field and polyMesh/boundary files.

Reads dictionaries (fields) and entries to change from a dictionary. E.g. to make the *movingWall* a *fixedValue* for p but all other *walls* a zeroGradient boundary condition, the *system/changeDictionaryDict* would contain the following:

```
dictionaryReplacement
{
    p                           // field to change
    {
        boundaryField
        {
            ".*Wall"            // entry to change
            {
                type            zeroGradient;
            }
            movingWall          // entry to change
            {
                type            fixedValue;
                value           uniform 123.45;
            }
        }
    }
}
```

Replacement entries starting with '∼' will remove the entry.

사용법

> changeDictionary [options]

옵션

- dict <file> : read control dictionary from specified location
- disablePatchGroups : disable matching keys to patch groups
- enableFunctionEntries : enable expansion of dictionary directives - #include, #codeStream etc
- instance <name> : override instance setting (default is the time name)
- literalRE : treat regular expressions literally (i.e., as a keyword)

- time <ranges> : comma-separated time ranges - eg, ':10,20,40:70,1000:'

사용 예

> changeDictionary

Example of changeDictionaryDict - of-2.3.x

```
dictionaryReplacement
{
    boundary
    {
        ".*"
        {
            type            mappedPatch;
        }
    }

    T
    {
        internalField   uniform 300;

        boundaryField
        {
            ".*"
            {
                type            zeroGradient;
            }

            minY
            {
                type            fixedValue;
                value           uniform 500;
            }
        }
    }

    rho
    {
        internalField   uniform 8000;

        boundaryField
        {
            ".*"
            {
                type            zeroGradient;
            }

        }
    }
```

```
    K
    {
        internalField   uniform 80;

        boundaryField
        {
            ".*"
            {
                type            zeroGradient;
            }

        }
    }

    cp
    {
        internalField   uniform 450;

        boundaryField
        {
            ".*"
            {
                type            zeroGradient;
            }

        }
    }
}
```

6.5 createExternalCoupledPatchGeometry

기능

Application to generate the patch geometry (points and faces) for use with the externalCoupled boundary condition.

On execution, the field <fieldName> is read, and its boundary conditions interrogated for the presence of an *externalCoupled* type. If found, the patch geometry (points and faces) for the coupled patches are output to the communications directory.

Note

The addressing is patch-local, i.e. point indices for each patch point used for face addressing starts at index 0.

사용법

> createExternalCoupledPatchGeometry [options] <fieldName>

사용 예

> ???

6.6 dsmcInitialise

기능

Initialise a case for dsmcFoam by reading the initialisation dictionary system/dsmcInitialise.

사용법

> dsmcInitialise [options]

사용 예

> ???

6.7 engineSwirl

기능

Generates a swirling flow for engine calulations.

사용법

> engineSwirl [options]

사용 예

> ???

6.8 faceAgglomerate

기능

Agglomerate boundary faces using the pairPatchAgglomeration algorithm. It writes a map from the fine to coarse grid.

사용법

> faceAgglomerate [options]

사용 예

> faceAgglomerate

Example of viewFactorsDict - of-2.3.x

```
// Write agglomeration as a volScalarField with calculated boundary values
writeFacesAgglomeration   true;
```

```
//Debug option
debug                       0;

//Dump connectivity rays
dumpRays                    false;

// Per patch (wildcard possible) the coarsening level
bottomAir_to_heater
{
    nFacesInCoarsestLevel     30;
    featureAngle              10;
}
```

6.9 foamUpgradeCyclics

기능

Tool to upgrade mesh and fields for split cyclics.

사용법

> foamUpgradeCyclics [options]

옵션

- enableFunctionEntries : enable expansion of dictionary directives - #include, #codeStream etc
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option
- test : test only; do not change any files
- time <ranges> : comma-separated time ranges - eg, ':10,20,40:70,1000:'

사용 예

> ???

6.10 foamUpgradeFvSolution

기능

Simple tool to upgrade the syntax of system/fvSolution.solvers.

사용법

> foamUpgradeFvSolution [options]

옵션

- test : suppress writing the updated system/fvSolution file

사용 예

> ???

6.11 mapFields

기능

주어진 시간 폴더에 있는 모든 fields 데이터를 하나의 격자에서 다른 격자에 interpolation 한다.2.3 버전에서 병렬 작동과 매핑 기법 설정이 가능하게 되었다. 매핑 기법은 다음의 세 가지이다.

- direct: the meshes are assumed to be of identical topology, with one-to-one correspondence between cells, but perhaps with different addressing, e.g. order of points, faces, cells; mapping is volume conservative
- nearest: cell values on the new mesh are sampled based on the nearest cell to the source mesh; not volume conservative
- conservative (default): volume weighted for internal cells, using face area weight AMI for boundaries.

사용법

> ampFields [options] <sourceCase>>

옵션

-consistent <name>	원본과 대상의 격자 및 경계조건이 동일할 때 사용
-fields <list>	'(U T p)'와 같이 리스트에서 지정한 필드에만 작동
-mapMethod <word>	매핑(mapping) 기법을 지정
-noLagrangian	라그랑지안 관련 데이터는 제외한다.
-sourceRegion <word>	원본의 특정 region을 지정
-sourceTime <scalar 혹은 'latestTime'>	원본의 시간을 지정
-targetRegion <word>	대상의 region을 지정
-subtract	subtract mapped source from target
-parallel	병렬로 작동한다.
-roots <(dir1 .. dirN)>	slave root directories for distributed running

사용 예

>mapFields -consistent -fields '(U p)' -sourceTime 100 ../testup/testSource

Example of mapFieldsDict - of-2.3

```
// Specify how to map patches. There are three different options:
// - patch exists in the source case: specify mapping (patchMap)
// - patch should be interpolated from internal values in source case (
    cuttingPatches)
// - patch should not be mapped. Default if not in patchMap or cuttingPatches

// List of pairs of target/source patches for mapping
patchMap
(
    lid movingWall
);

// List of target patches cutting the source domain (these need to be handled
    specially e.g. interpolated from internal values)
cuttingPatches
(
    fixedWalls
);
```

6.12 mdInitialise

molecular dynamics(MD) 해석시 필드를 초기화 한다.

6.13 setFields

기능

딕셔너리에서 설정한 cell 혹은 경계면에 특정한 값을 준다.

사용법

> setFields [options]

옵션

-region <name>	지정한 region에만 적용
-parallel	병렬로 작동한다.
-roots <(dir1 .. dirN)>	slave root directories for distributed running

사용 예

>setFields -region test

Example of setFieldsDict - of-2.3

```
defaultFieldValues
(
    volScalarFieldValue alpha1 0
```

```
    volVectorFieldValue U (0 0 0)
);
regions
(
    // Set cell values
    // (does zerogradient on boundaries)
    boxToCell
    {
        box (0 0 -1) (0.1461 0.292 1);
        fieldValues
        (
            volScalarFieldValue alpha1 1
        );
    }
    // Set patch values (using ==)
    boxToFace
    {
        box (0 0 -1) (0.1461 0.292 1);
        fieldValues
        (
            volScalarFieldValue alpha1 1
        );
    }
);
```

6.14 viewFactorsGen

기능

viewFactor 복사열전달 모델을 사용할 때 view factor를 계산한다.

View factors are calculated based on a face agglomeration array (finalAgglom generated by faceAgglomerate utility).

Each view factor between the agglomerated faces i and j (Fij) is calculated using a double integral of the sub-areas composing the agglomaration.

The patches involved in the view factor calculation are taken from the Qr volScalarField (radiative flux) when is greyDiffusiveRadiationViewFactor otherwise they are not included.

사용법

> viewFactorsGen [options]

옵션

-region <name>	지정한 region에만 적용
-parallel	병렬로 작동한다.
-roots <(dir1 .. dirN)>	slave root directories for distributed running

사용 예

>viewFactorsGen -region test

6.15 wallFunctionTable

Generates a table suitable for use by tabulated wall functions.

7 parallelProcessing

7.1 decomposePar

기능

Automatically decomposes a mesh and fields of a case for parallel execution of OpenFOAM.

사용법

> decomposePar [options]

옵션

- allRegions : operate on all regions in regionProperties
- cellDist : write cell distribution as a labelList - for use with 'manual' decomposition method or as a volScalarField for post-processing.
- copyUniform : copy any uniform/ directories too
- fields : use existing geometry decomposition and convert fields only
- force : remove existing processor*/ subdirs before decomposing the geometry
- ifRequired : only decompose geometry if the number of domains has changed
- noSets : skip decomposing cellSets, faceSets, pointSets
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option

사용 예

> decomposePar -force -latestTime

Example of createPatchDict - of-2.2.x

```
numberOfSubdomains  2;

//- Keep owner and neighbour on same processor for faces in zones:
// preserveFaceZones (heater solid1 solid3);
```

```
//- Keep owner and neighbour on same processor for faces in patches:
//  (makes sense only for cyclic patches)
//preservePatches (cyclic_half0 cyclic_half1);

//- Keep all of faceSet on a single processor. This puts all cells
//  connected with a point, edge or face on the same processor.
//  (just having face connected cells might not guarantee a balanced
//  decomposition)
// The processor can be -1 (the decompositionMethod chooses the processor
// for a good load balance) or explicitly provided (upsets balance).
//singleProcessorFaceSets ((f0 -1));

//- Keep owner and neighbour of baffles on same processor (i.e. keep it
//  detectable as a baffle). Baffles are two boundary face sharing the
//  same points.
//preserveBaffles true;

//- Use the volScalarField named here as a weight for each cell in the
//  decomposition.  For example, use a particle population field to decompose
//  for a balanced number of particles in a lagrangian simulation.
// weightField dsmcRhoNMean;

method          scotch;//hierachical;//simple;//metis;//manual;//multiLevel;
// method          structured;  // does 2D decomposition of structured mesh

multiLevelCoeffs
{
    // Decomposition methods to apply in turn. This is like hierarchical but
    // fully general - every method can be used at every level.

    level0
    {
        numberOfSubdomains  64;
        //method simple;
        //simpleCoeffs
        //{
        //    n           (2 1 1);
        //    delta       0.001;
        //}
        method scotch;
    }
    level1
    {
        numberOfSubdomains  4;
        method scotch;
    }
}

// Desired output
```

```
simpleCoeffs
{
    n           (2 1 1);
    delta       0.001;
}

hierarchicalCoeffs
{
    n           (1 2 1);
    delta       0.001;
    order       xyz;
}

metisCoeffs
{
 /*
    processorWeights
    (
        1
        1
        1
        1
    );
  */
}

scotchCoeffs
{
    //processorWeights
    //(
    //    1
    //    1
    //    1
    //    1
    //);
    //writeGraph  true;
    //strategy "b";
}

manualCoeffs
{
    dataFile    "decompositionData";
}

structuredCoeffs
{
    // Patches to do 2D decomposition on. Structured mesh only; cells have
    // to be in 'columns' on top of patches.
    patches     (movingWall);
```

```
    // Method to use on the 2D subset
    method      scotch;
}

//// Is the case distributed? Note: command-line argument -roots takes
//// precedence
//distributed     yes;
//// Per slave (so nProcs-1 entries) the directory above the case.
//roots
//(
//    "/tmp"
//    "/tmp"
//);
```

7.2 reconstructPar

기능

Reconstructs fields of a case that is decomposed for parallel execution of OpenFOAM.

사용법

> reconstructPar [options]

옵션

- allRegions : operate on all regions in regionProperties
- fields <list> : specify a list of fields to be reconstructed. Eg, '(U T p)' - regular expressions not currently supported
- lagrangianFields <list> : specify a list of lagrangian fields to be reconstructed. Eg, '(U d)' -regular expressions not currently supported, positions always included.
- newTimes : only reconstruct new times (i.e. that do not exist already)
- noLagrangian : skip reconstructing lagrangian positions and fields
- noSets : skip reconstructing cellSets, faceSets, pointSets
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option

사용 예

> reconstructPar -latestTime

7.3 reconstructParMesh

기능

Reconstructs a mesh using geometric information only.

Writes point/face/cell procAddressing so afterwards reconstructPar can be used to reconstruct fields.

Note

- uses geometric matching tolerance (set with -mergeTol (at your option)

If the parallel case does not have correct procBoundaries use the

- fullMatch option which will check all boundary faces (bit slower).

사용법

> reconstructParMesh [options]

옵션

- cellDist : write cell distribution as a labelList - for use with 'manual' decomposition method or as a volScalarField for post-processing.
- fullMatch : do (slower) geometric matching on all boundary faces
- mergeTol <scalar> : specify the merge distance relative to the bounding box size(default 1e-7)
- -noZero : exclude the '0/' dir from the times list

사용 예

> reconstructParMesh

7.4 redistributePar

기능

Redistributes existing decomposed mesh and fields according to the current settings in the decomposeParDict file.

Must be run on maximum number of source and destination processors.
Balances mesh and writes new mesh to new time directory.

Can also work like decomposePar:

```
# Create empty processor directories (have to exist for argList)
mkdir processor0
        ..
mkdir processorN

# Copy undecomposed polyMesh
cp -r constant processor0

# Distribute
mpirun -np ddd redistributePar -parallel
```

사용법

> redistributePar [options]

옵션

- mergeTol <scalar> : specify the merge distance relative to the bounding box size (default 1e-6)

사용 예

> ???

8 PostProcessing - data conversion

8.1 foamDataToFluent

기능

Translates OpenFOAM data to Fluent format. Need dictionary file.

사용법

> foamDataToFluent [options]

사용 예

> foamDataToFluent

Example of foamDataToFluentDict - of-2.3.x

```
p               1;
U               2;
```

8.2 foamToEnsight

기능

Translates OpenFOAM data to EnSight format.

An Ensight part is created for the internalMesh and for each patch.

사용법

> foamToEnsight [options]

옵션

- ascii : write in ASCII format instead of 'C Binary'
- cellZone <word> : specify cellZone to write
- faceZones <wordReList> : specify faceZones to write - eg '(slice "mfp-.*")'.
- fields <wordReList> : specify fields to export (all by default) - eg '("U.*")'.
- noPatches : suppress writing any patches
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option
- nodeValues : write values in nodes
- patches <wordReList> : specify particular patches to write - eg '(outlet "inlet.*")'. An empty list suppresses writing the internalMesh.

사용 예

> ???

8.3 foamToEnsightParts

기능

Translates OpenFOAM data to Ensight format.
An Ensight part is created for each cellZone and patch.

사용법

> foamToEnsightParts [options]

옵션

- ascii : write in ASCII format instead of 'C Binary'
- index <start> : ignore the time index contained in the uniform/time file and use simple indexing when creating the files
- name <subdir> : define sub-directory name to use for Ensight data (default:"Ensight")
- noMesh : suppress writing the geometry. Can be useful for converting partial results for a static geometry
- width <n> : width of Ensight data subdir

사용 예

> ???

8.4 foamToGMV

기능

Translates foam output to GMV readable files.

A free post-processor with available binaries from http://www-xdiv.lanl.gov/XCM/gmv/

사용법

> foamToGMV [options]

사용 예

> ???

8.5 foamToTecplot360

기능

Tecplot binary file format writer.

사용법

> foamToTecplot360 [options]

옵션

- fields <names> : Convert selected fields only. For example, -fields '(p T U)'
- cellSet <name> : Restrict conversion to the cellSet
- faceSet <name> : Restrict conversion to the faceSet

- nearCellValue : Output cell value on patches instead of patch value itself
- noInternal : Do not generate file for mesh, only for patches
- noPointValues : No pointFields
- noFaceZones : No faceZones
- excludePatches <patchNames> : Specify patches (wildcards) to exclude. For example,

```
-excludePatches '( inlet_1 inlet_2 "proc.*")'
```

- useTimeName : use the time index in the VTK file name instead of the time index

사용 예

> ???

8.6 foamToTetDualMesh

기능

Converts polyMesh results to tetDualMesh.

사용법

> foamToTetDualMesh [options]

사용 예

> ???

8.7 foamToVTK

기능

Legacy VTK file format writer.

- handles volScalar, volVector, pointScalar, pointVector, surfaceScalar fields.
- mesh topo changes.
- both ascii and binary.
- single time step writing.
- write subset only.
- automatic decomposition of cells; polygons on boundary undecomposed since handled by vtk.

사용법

> foamToVTK [options]

옵션

- allPatches : combine all patches into a single file
- ascii : write in ASCII format instead of binary
- cellSet <name> : convert a mesh subset corresponding to the specified cellSet
- excludePatches <wordReList> : a list of patches to exclude - eg '(inlet ".*Wall")'
- faceSet <name> : restrict conversion to the specified faceSet
- fields <wordList> : only convert the specified fields - eg '(p T U)'
- nearCellValue : use cell value on patches instead of patch value itself
- noFaceZones : no faceZones
- noInternal : do not generate file for mesh, only for patches
- noLinks : don't link processor VTK files - parallel only
- noPointValues : no pointFields
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option
- pointSet <name> : restrict conversion to the specified pointSet
- poly : write polyhedral cells without tet/pyramid decomposition
- surfaceFields : write surfaceScalarFields (e.g., phi)
- useTimeName : use the time name instead of the time index when naming the files

사용 예

> ???

8.8 smapToFoam

기능

Translates a STAR-CD SMAP data file into OpenFOAM field format.

사용법

> smapToFoam [options] <SMAP fileName>

사용 예

> ???

9 Postprocessing - graphics

9.1 ensightFoamReader

9.2 PV3Readers

9.3 PV4Readers

10 Postprocessing - lagrangian

10.1 particleTracks

기능

tracked-parcel-type cloud를 계산한 문제에서 입자의 데이터를 VTK 파일로 써 준다. particleTrackProperties 파일을 참조한다.

사용법

> particleTracks [options]

사용예

> particleTracks -latestTime

Example of particleTrackProperties - of-2.3

```
cloudName       reactingCloud1;
sampleFrequency 1;
maxPositions    1000000;
setFormat       vtk;    // see sampleDict for set formats
```

10.2 steadyParticleTracks

기능

정상상태 cloud를 계산한 문제에서 입자의 데이터를 VTK 파일로 써 준다. particleTrackProperties 파일을 참조한다. 병렬연산 결과의 경우는 반드시 reconstruct를 해 주어야 한다.

사용법

> particleTracks [options]

옵션

사용 예

> steadyParticleTracks -latestTime

Example of particleTrackProperties - of-2.3

```
cloudName       reactingCloud1Tracks;
fields ( d U T );
```

11 Postprocessing - miscellaneous

11.1 dsmcFieldCalc

Calculate intensive fields (U and T) from averaged extensive fields from a DSMC calculation.

11.2 engineCompRatio

Calculate the geometric compression ratio. Note that if you have valves and/or extra volumes it will not work, since it calculates the volume at BDC and TCD.

11.3 execFlowFunctionObjects

기능

system/controlDict 파일 혹은 지정된 딕셔너리 파일에서 선언된 functionObjects를 실행한다.

사용법

> execFlowFunctionObjects [options]

옵션

-noFlow	suppress creating flow models

사용 예

> execFlowFunctionObjects -latestTime

11.4 foamListTimes

기능

list times using timeSelector

사용법

> foamListTimes [options]

옵션

-processor	list times from processor0/ directory

사용 예

> foamListTimes -latestTime

11.5 pdfPlot

기능

Generates a graph of a probability distribution function.

사용법

> pdfPlot [options]

사용 예

> pdfPlot

Example of pdfDict - of-2.3

```
// Number of intervals/bins in pdf plot
nIntervals      20;
// Number of samples
nSamples        10000;
// Type of pdf
type            RosinRammler;
// Write data flag
writeData       true;
// PDF model coefficients
RosinRammlerDistribution
{
    minValue        1e-06;
    maxValue        200e-06;
    d               60.0e-06;
    n               0.8;
}
```

11.6 postChannel

Post-processes data from channel flow calculations.

11.7 ptot

기능

total pressure를 계산해서 각 폴더에 써 준다.

사용법

> ptot [options]

사용 예

> ptot -latestTime

11.8 temporalInterpolate

기능

Interpolate fields between time-steps e.g. for animation.

사용법

> temporalInterpolate [options]

옵션

-divisions <integer>	specify number of temporal sub-divisions to create (default=1)
-fields <list>	specifiy a list of fields. Eg, '(U p)' - regular expressions not support
-interpolationType <word>	specify type of interpolation (linear or spline)

사용 예

> temporalInterpolate

11.9 wdot

기능

Calculates and writes wdot for each time.

사용법

> wdot [options]

사용 예

> wdot -latestTime

11.10 writeCellCenters

기능

Write the three components of the cell centres as volScalarFields so they can be used in postprocessing in thresholding.

사용법

> writeCellCenters [options]

사용 예

> writeCellCenters -latestTime

12 Postprocessing - patch

12.1 patchAverage

기능

지정된 면(patch)에서 지정된 필드(field)을 평균값을 계산한다.

사용법

> patchAverage [options] <fieldName> <patchName>

사용 예

> patchAverage -latestTime p testPatch

12.2 patchIntegrate

기능

지정된 면(patch)에서 지정된 필드(field)을 적분(integrate)을 계산한다.

사용법

> patchIntegrate [options] <fieldName> <patchName>

사용 예

> patchIntegrate -latestTime p testPatch

13 Postprocessing - sampling

13.1 probeLocations

기능

지정한 위치에서 값을 추출한다.

사용법

> probeLocations [options]

사용 예

> probeLocations -region test

Example of probesDict - of-2.3

```
// Fields to be probed. runTime modifiable!
fields
( p );
// Locations to be probed. runTime modifiable!
probeLocations
(
    (0.0254 0.0253 0.0)
    (0.0508 0.0253 0.0)
    (0.0762 0.0253 0.0)
    (0.1016 0.0253 0.0)
    (0.1270 0.0253 0.0)
    (0.1524 0.0253 0.0)
    (0.1778 0.0253 0.0)
);
```

13.2 sample

기능

주어진 interpolation schemes, sampling options, write formats에 따라 필드 데이터를 추출한다. Keyword는 다음과 같다.

- setFormat : set output format
  - xmgr
  - jplot
  - gnuplot
  - raw
- surfaceFormat : surface output format, choice of
  - null : suppress output
  - foamFile : separate points, faces and values file
  - dx : DX scalar or vector format
  - vtk : VTK ascii format
  - raw : x y z value format for use with e.g. gnuplot 'splot'.
  - obj : Wavefron stl. Does not contain values!
  - stl : ascii stl. Does not contain values!
- interpolationScheme : interpolation scheme, choice of

  - cell : use cell-centre value; constant over cells (default)
  - cellPoint : use cell-centre and vertex values
  - cellPointFace : use cell-centre, vertex and face values.
    # vertex values determined from neighbouring cell-centre values
    # face values determined using the current face interpolation scheme for the field (linear, limitedLinear, etc.)

- fields : list of fields to sample
- sets : list of sets to sample, choice of
  - uniform evenly distributed points on line
  - face one point per face intersection
  - midPoint one point per cell, inbetween two face intersections
  - midPointAndFace combination of face and midPoint
  - curve specified points, not nessecary on line, uses tracking
  - cloud specified points, uses findCell
    # Option axis: how to write point coordinate. Choice of
    - x/y/z: x/y/z coordinate only
    - xyz: three columns (probably does not make sense for anything but raw)
    - distance: distance from start of sampling line (if uses line) or distance from first specified sampling point
    # Type specific options:
    - uniform, face, midPoint, midPointAndFace : start and end coordinate
    - uniform: extra number of sampling points
    - curve, cloud: list of coordinates
- surfaces : list of surfaces to sample, choice of
  - plane : values on plane defined by point, normal.
  - patch : values on patch.

사용법

> sample [options]

사용 예

> sample

Example of sampleDict - of-2.3

```
setFormat raw; // xmgr, jplot, gnuplot, vtk, ensight, csv
```

```
// Surface output format. Choice of
//      null        : suppress output
//      ensight     : Ensight Gold format, one field per case file
//      foamFile    : separate points, faces and values file
//      dx          : DX scalar or vector format
//      vtk         : VTK ascii format
//      raw         : x y z value format for use with e.g. gnuplot 'splot'.
//
// Note:
// other formats such as obj, stl, etc can also be written (by proxy)
// but without any values!
surfaceFormat vtk;

// optionally define extra controls for the output formats
formatOptions
{
    ensight
    {
        format  ascii;
    }
}

// interpolationScheme. choice of
//      cell          : use cell-centre value only; constant over cells(default)
//      cellPoint     : use cell-centre and vertex values
//      cellPointFace : use cell-centre, vertex and face values.
//      pointMVC      : use point values only (Mean Value Coordinates)
//      cellPatchConstrained : like 'cell' but uses cell-centre except on
    boundary faces where it uses the boundary value. For use with e.g.
    patchCloudSet.
// 1] vertex values determined from neighbouring cell-centre values
// 2] face values determined using the current face interpolation scheme for the
    field (linear, gamma, etc.)
interpolationScheme cellPoint;

// Fields to sample.
fields
(
    p
    U
);

// Set sampling definition: choice of
//      uniform             evenly distributed points on line
//      face                one point per face intersection
//      midPoint            one point per cell, inbetween two face intersections
//      midPointAndFace     combination of face and midPoint//
//      polyLine            specified points, not nessecary on line, uses
    tracking
//      cloud               specified points, uses findCell
```

```
//      triSurfaceMeshPointSet  points of triSurface
//
// axis: how to write point coordinate. Choice of
// - x/y/z: x/y/z coordinate only
// - xyz: three columns (probably does not make sense for anything but raw)
// - distance: distance from start of sampling line (if uses line) or distance
    from first specified sampling point
//
// type specific:
//      uniform, face, midPoint, midPointAndFace : start and end coordinate
//      uniform : extra number of sampling points
//      polyLine, cloud : list of coordinates
//      patchCloud : list of coordinates and set of patches to look for nearest
//      patchSeed : random sampling on set of patches. Points slightly off
//                  face centre.
sets
(
    lineX1
    {
        type        uniform;
        axis        distance;

        //- cavity. Slightly perturbed so not to align with face or edge.
        start       (0.0201 0.05101 0.00501);
        end         (0.0601 0.05101 0.00501);
        nPoints     10;
    }

    lineX2
    {
        type        face;
        axis        x;

        //- cavity
        start       (0.0001 0.0525 0.00501);
        end         (0.0999 0.0525 0.00501);
    }

    somePoints
    {
        type    cloud;
        axis    xyz;
        points  ((0.049 0.049 0.00501)(0.051 0.049 0.00501));
    }

    somePatchPoints
    {
        // Sample nearest points on selected patches. Looks only up to
        // maxDistance away. Any sampling point not found will get value
        // pTraits<Type>::max (usually VGREAT)
```

```
        // Use with interpolations:
        // - cell (cell value)
        // - cellPatchConstrained (boundary value)
        // - cellPoint (interpolated boundary value)
        type        patchCloud;
        axis        xyz;
        points      ((0.049 0.099 0.005)(0.051 0.054 0.005));
        maxDistance 0.1;    // maximum distance to search
        patches     (".*Wall.*");
    }

    patchSeed
    {
        type        patchSeed;
        patches     (".*Wall.*");
        // Number of points to seed. Divided amongst all processors according to
            fraction of patches they hold.
        maxPoints   100;
    }

);

// Surface sampling definition
//
// 1] patches are not triangulated by default
// 2] planes are always triangulated
// 3] iso-surfaces are always triangulated surfaces
(
    constantPlane
    {
        type            plane;    // always triangulated
        basePoint       (0.0501 0.0501 0.005);
        normalVector    (0.1 0.1 1);

        //- Optional: restrict to a particular zone
        // zone        zone1;
        //- Optional: do not triangulate (only for surfaceFormats that support
            polygons)
        //triangulate     false;
    }

    interpolatedPlane
    {
        type            plane;    // always triangulated
        // make plane relative to the coordinateSystem (Cartesian)
        coordinateSystem
        {
            origin      (0.0501 0.0501 0.005);
        }
        basePoint       (0 0 0);
```

```
    normalVector    (0.1 0.1 1);
    interpolate     true;
}

walls_constant
{
    type            patch;
    patches         ( ".*Wall.*" );
    // Optional: whether to leave as faces (=default) or triangulate
    // triangulate     false;
}

walls_interpolated
{
    type            patch;
    patches         ( ".*Wall.*" );
    interpolate     true;
    // Optional: whether to leave as faces (=default) or triangulate
    // triangulate     false;
}

nearWalls_interpolated
{
    // Sample cell values off patch. Does not need to be the near-wall cell,
        can be arbitrarily far away.
    type            patchInternalField;
    patches         ( ".*Wall.*" );
    interpolate     true;

    // Optional: specify how to obtain sampling points from the patch face
        centres (default is 'normal')
    //
    //  //- Specify distance to offset in normal direction
    offsetMode  normal;
    distance    0.1;
    //
    //  //- Specify single uniform offset
    //  offsetMode  uniform;
    //  offset      (0 0 0.0001);
    //
    //  //- Specify offset per patch face
    //  offsetMode  nonuniform;
    //  offsets     ((0 0 0.0001) (0 0 0.0002));

    // Optional: whether to leave as faces (=default) or triangulate
    // triangulate     false;
}

interpolatedIso
{
```

```
    // Iso surface for interpolated values only
    type            isoSurface;    // always triangulated
    isoField        rho;
    isoValue        0.5;
    interpolate     true;

    //zone            ABC;          // Optional: zone only
    //exposedPatchName fixedWalls;  // Optional: zone only

    // regularise      false;    // Optional: do not simplify
    // mergeTol        1e-10;    // Optional: fraction of mesh bounding box
                                 // to merge points (default=1e-6)
}
constantIso
{
    // Iso surface for constant values.
    // Triangles guaranteed not to cross cells.
    type            isoSurfaceCell;    // always triangulated
    isoField        rho;
    isoValue        0.5;
    interpolate     false;
    regularise      false;              // do not simplify
    // mergeTol        1e-10;    // Optional: fraction of mesh bounding box
                                 // to merge points (default=1e-6)
}

triangleCut
{
    // Cutingplane using iso surface
    type            cuttingPlane;
    planeType       pointAndNormal;
    pointAndNormalDict
    {
        basePoint       (0.4 0 0.4);
        normalVector    (1 0.2 0.2);
    }
    interpolate     true;

    //zone            ABC;          // Optional: zone only
    //exposedPatchName fixedWalls;  // Optional: zone only

    // regularise      false;    // Optional: do not simplify
    // mergeTol        1e-10;    // Optional: fraction of mesh bounding box
                                 // to merge points (default=1e-6)
}

distance
{
    // Isosurface from signed/unsigned distance to surface
    type            distanceSurface;
```

```
        signed          true;

        // Definition of surface
        surfaceType     triSurfaceMesh;
        surfaceName     integrationPlane.stl;
        // Distance to surface
        distance        0.0;

        //cell            false;// optional: use isoSurface instead
                                // of isoSurfaceCell
        interpolate     false;
        regularise      false;  // Optional: do not simplify
        // mergeTol 1e-10;      // Optional: fraction of mesh bounding box
                                // to merge points (default=1e-6)
    }

    triSurfaceSampling
    {
        // Sampling on triSurface
        type        sampledTriSurfaceMesh;
        surface     integrationPlane.stl;
        source      boundaryFaces;  // What to sample: cells (nearest cell)
                                    // insideCells (only triangles inside cell)
                                    // boundaryFaces (nearest boundary face)
        interpolate true;
    }
);
```

14 Postprocessing - scalarField

14.1 pPrime2

Calculates and writes the scalar field of pPrime2 (sqr(p - pMean)) at each time

15 Postprocessing - stressField

15.1 stressComponents

Calculates and writes the scalar fields of the six components of the stress tensor sigma for each time.

16 Postprocessing - turbulence

16.1 createTurbulenceFields

Creates a full set of turbulence fields.

16.2 R

기능

Calculates and writes the Reynolds stress R for the current time step.

사용법

> R [options]

옵션

-compressible	calculate compressible R 작동한다.

사용 예

> R -region test

17 Postprocessing - velocityField

17.1 Co

Calculates and writes the Co number as a volScalarField obtained from field phi.
The -noWrite option just outputs the max values without writing the field.

17.2 enstrophy

Calculates and writes the enstrophy of the velocity field U.
The -noWrite option just outputs the max/min values without writing the field.

17.3 flowType

Calculates and writes the flowType of velocity field U.
The -noWrite option has no meaning.

The flow type parameter is obtained according to the following equation:

$$lambda = \frac{|D| - |Omega|}{|D| + |Omega|} \tag{17.1}$$

-1 = rotational flow
0 = simple shear flow
1 = planar extensional flow

17.4 Lambda2

Calculates and writes the second largest eigenvalue of the sum of the square of the symmetrical and anti-symmetrical parts of the velocity gradient tensor.
The -noWrite option has no meaning.

17.5 Mach

Calculates and optionally writes the local Mach number from the velocity field U at each time.
The -nowrite option just outputs the max value without writing the field.

17.6 Pe

Calculates and writes the Pe number as a surfaceScalarField obtained from field phi.
The -noWrite option just outputs the max/min values without writing the field.

17.7 Q

Calculates and writes the second invariant of the velocity gradient tensor.

$$Q = 0.5 * (sqr(tr(gradU)) - tr(((gradU)\&(gradU)))) \qquad [1/s^2] \qquad (17.2)$$

The -noWrite option just outputs the max/min values without writing the field.

17.8 streamFunction

Calculates and writes the stream function of velocity field U at each time.

17.9 uprime

Calculates and writes the scalar field of uprime (sqrt(2/3 k)).
The -noWrite option just outputs the max/min values without writing the field.

17.10 vorticity

Calculates and writes the vorticity of velocity field U.
The -noWrite option just outputs the max/min values without writing the field.

18 Postprocessing - wall

18.1 wallGradU

Calculates and writes the gradient of U at the wall.

18.2 wallHeatFlux

Calculates and writes the heat flux for all patches as the boundary field of a volScalarField and also prints the integrated flux for all wall patches.

18.3 wallShearStress

기능

Calculates and writes the wall shear stress, for the specified times when using RAS turbulence models.
(Of2.2 이상과 1.6-nxt 에서만 작동)

사용법

> wallShearStress [options]

옵션

-compressible	calculate compressible wall shear stress

사용 예

> wallShearStress -compressible

18.4 yPlusLES

Calculates and reports yPlus for all wall patches, for the specified times when using LES turbulence models.

18.5 yPlusRAS

기능

Calculates and reports yPlus for all wall patches, for the specified times when using RAS turbulence models.

사용법

> yPlusRAS [options]

옵션

-compressible	calculate compressible y+

사용 예

> yPlusRAS

19 Postprocessing - noise

Utility to perform noise analysis of pressure data using the noiseFFT library.

Control settings are read from the $FOAM_CASE/system/noiseDict dictionary, or user-specified dictionary using the -dict option. Pressure data is read using a CSV reader:

Usage of noise - of-2.3

```
pRef         101325;
N            65536;
nw           100;
f1           25;
fU           10000;
graphFormat raw;

csvFileData
{
    fileName        "pressureData"
    nHeaderLine     1;
    refColumn       0;
    componentColumns (1);
    separator       " ";
}
```

Property	Description	Required	Default value
pRef	Reference pressure	no	0
N	Number of samples in sampling window	no	65536
nw	Number of sampling window	no	100
fl	Lower frequency band	no	25
fU	Upper frequency band	no	10000
graphFormat	Output graph format	no	raw

Current graph outputs include:

- FFT of the pressure data
- narrow-band PFL (pressure-fluctuation level) spectrum
- one-third-octave-band PFL spectrum
- one-third-octave-band pressure spectrum

SeeAlso CSV.H, noiseFFT.H

20 Postprocessing - foamCalc

기능

This utility calculates new fields from existing ones. This replaces some of the utilities in previous versions of OpenFOAM, such as magU and Ucomponents.

사용법

> foamCalc <calcType> <fieldName1 ... fieldNameN>

calcType

- randomise
- magSqr
- magGrad
- addSubtract
- div
- mag
- interpolate
- components

사용 예

> foamCalc div U

> foamCalc components U

21 thermophysical

21.1 adiabaticFlameT

Calculates the adiabatic flame temperature for a given fuel over a range of unburnt temperatures and equivalence ratios.

21.2 chemkinToFoam

Converts CHEMKINIII thermodynamics and reaction data files into OpenFOAM format.

21.3 equilibriumCO

Calculates the equilibrium level of carbon monoxide.

21.4 equilibriumFlameT

Calculates the equilibrium flame temperature for a given fuel and pressure for a range of unburnt gas temperatures and equivalence ratios; the effects of dissociation on O2, H2O and CO2 are included.

21.5 mixtureAdiabaticFlameT

Calculates the adiabatic flame temperature for a given mixture at a given temperature.

22 surface

22.1 surfaceAdd

기능

두 개의 STL 파일을 하나로 합친다.

사용법

> surfaceAdd [options] <surface file 1> <surface file 2> <output surface file>

옵션

-mergeRegions	combine regions from both surfaces
-points <file>	provide additional points

사용 예

> surfaceAdd wing.stl fuselage.stl vehicle.stl

22.2 surfaceAutoPatch

기능

Patches surface according to feature angle. Like autoPatch

사용법

> surfaceAutoPatch [options] <input surface file> <output surface file> <includedAngle [0..180]>

사용 예

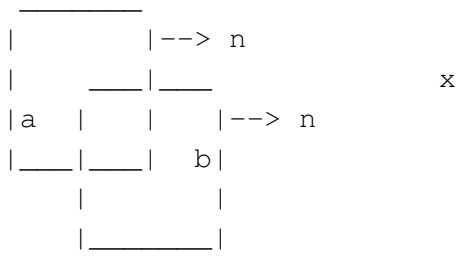
> surfaceAutoPatch surface1.stl surface2.stl 89

22.3 surfaceBooleanFeatures

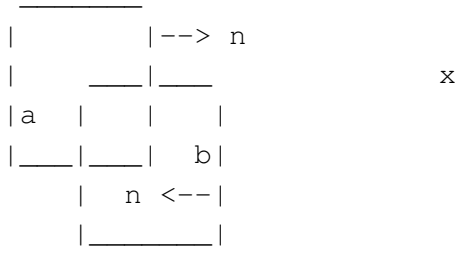
기능

Generates the extendedFeatureEdgeMesh for the interface between a boolean operation on two surfaces. Assumes that the orientation of the surfaces is correct:

+ if the operation is union or intersection, that both surface's normals (n) have the same orientation with respect to a point, i.e. surfaces and b are orientated the same with respect to point x:



+ if the operation is a subtraction, the surfaces should be oppositely oriented with respect to a point, i.e. for (a - b), then b's orientation should be such that x is "inside", and a's orientation such that x is "outside"



When the operation is peformed - for union, all of the edges generates where one surfaces cuts another are all "internal" for union, and "external" for intersection, b - a and a - b. This has been assumed, formal (dis)proof is invited.

사용법

> surfaceBooleanFeatures [options] <action> <surface file> <surface file>

action에 들어갈 수 있는 것은 union, difference, intersection 세 가지이다.

옵션

- invertedSpace : do the surfaces have inverted space orientation, i.e. a point at infinity is considered inside. This is only sensible for union and intersection.
- perturb : Perturb surface points to escape degenerate intersections
- surf1Baffle : Mark surface 1 as a baffle
- surf2Baffle : Mark surface 2 as a baffle

사용 예

> surfaceBooleanFeatures union a.stl b.stl

> surfaceBooleanFeatures difference a.stl b.stl

> surfaceBooleanFeatures intersection a.stl b.stl

22.4 surfaceCheck

기능

Checks geometric and topological quality of a surface.

사용법

> surfaceCheck [options] <surfaceFile>

옵션

- blockMesh : write vertices/blocks for blockMeshDict
- checkSelfIntersection : also check for self-intersection
- splitNonManifold : split surface along non-manifold edges (default split is fully disconnected)
- verbose : verbose operation

사용 예

> surfaceCheck a.stl

22.5 surfaceClean

기능

- removes baffles
- collapses small edges, removing triangles.
- converts sliver triangles into split edges by projecting point onto base of triangle.

사용법

> surfaceClean [options] <surfaceFile> <min length> <min quality> <output surfaceFile>

옵션

- noClean : perform some surface checking/cleanup on the input surface

사용 예

> surfaceClean blob.stl 0.001 0.0001 aa.stl

22.6 surfaceCoarsen

기능

Surface coarsening using 'bunnylod':
Polygon Reduction Demo. By Stan Melax (c) 1998, mailto:melax@cs.ualberta.ca, http://www.cs.ualberta.ca/ m

사용법

> surfaceCoarsen [options] <surfaceFile> <reductionFactor> <output surfaceFile>

사용 예

> surfaceCoarsen aa.stl 0.5 aaa.stl

22.7 surfaceConvert

기능

Converts from one surface mesh format to another.

사용법

> surfaceConvert [options] <inputFile> <outputFile>

옵션

- clean : perform some surface checking/cleanup on the input surface
- group : reorder faces into groups ; one per region
- scale <factor> : geometry scaling factor - default is 1
- writePrecision <label>: write to output with the specified precision

사용 예

> surfaceConvert a.stl a.obj

22.8 surfaceFeatureConvert

기능

Converts between edgeMesh formats

사용법

> surfaceFeatureConvert [options] <inputFile> <outputFile>

옵션

- scale <factor> : geometry scaling factor - default is 1

사용 예

> surfaceFeatureConvert constant/triSurfacea/a.eMesh aa.obj

22.9 surfaceFeatureExtract

기능

Extracts and writes surface features to file. All but the basic feature extraction is WIP.
Curvature calculation is an implementation of the algorithm from:
"Estimating Curvatures and their Derivatives on Triangle Meshes" by S. Rusinkiewicz

surfaceFeatureExtractDict 파일에서 설정

사용법

> surfaceFeatureExtract [options]

사용 예

> surfaceFeatureExtract

Example of surfaceFeatureExtractDict - of-2.3

```
surface1.stl
{
    // How to obtain raw features (extractFromFile || extractFromSurface)
    extractionMethod    extractFromSurface;
    extractFromSurfaceCoeffs
    {
        // Mark edges whose adjacent surface normals are at an angle less
        // than includedAngle as features
        // - 0  : selects no edges
        // - 180: selects all edges
        includedAngle   120;

        // Do not mark region edges
        geometricTestOnly       yes;
    }
    // Write options : Write features to obj format for postprocessing
        writeObj                yes;
}

surface2.nas
{
    extractionMethod    extractFromFile;
    extractFromFileCoeffs
    {
        // Load from an existing feature edge file
        featureEdgeFile "constant/triSurface/featureEdges.nas";
    }
    trimFeatures
    {
        // Remove features with fewer than the specified number of edges
        minElem         0;
        // Remove features shorter than the specified cumulative length
        minLen          0.0;
    }
    subsetFeatures
    {
        // Use a plane to select feature edges
        // (normal)(basePoint)
        // Keep only edges that intersect the plane will be included
        plane               (1 0 0)(0 0 0);

        // Select feature edges using a box
        // (minPt)(maxPt)
        // Keep edges inside the box:
        insideBox           (0 0 0)(1 1 1);
```

```
        // Keep edges outside the box:
        outsideBox          (0 0 0)(1 1 1);

        // Keep nonManifold edges (edges with >2 connected faces where
        // the faces form more than two different normal planes)
        nonManifoldEdges    yes;

        // Keep open edges (edges with 1 connected face)
        openEdges           yes;
    }
    addFeatures
    {
        // Add (without merging) another extendedFeatureEdgeMesh
        name                axZ.extendedFeatureEdgeMesh;

        // Optionally flip features (invert all normals, making
        // convex<->concave etc)
        //flip                false;
    }

    // Output the curvature of the surface
    curvature               no;

    // Output the proximity of feature points and edges to each other
    featureProximity        no;

    // The maximum search distance to use when looking for other feature
    // points and edges
    maxFeatureProximity     1;

    // Out put the closeness of surface elements to other surface elements.
    closeness               no;

    writeObj                yes;

    // Write surface proximity and curvature fields to vtk format for
       postprocessing
    writeVTK                no;
}
```

22.10 surfaceFind

기능

Finds nearest face and vertex

사용법

> surfaceFind [options] <surfaceFile>

옵션

- x <X> : Tho point x-coordinate (if non-zero)
- y <Y> : Tho point y-coordinate (if non-zero)
- z <Z> : Tho point z-coordinate (if non-zero)

사용 예

> surfaceFind -x 1.0 -y 1.0 -z 1.0 aa.stl

22.11 surfaceHookUp

기능

Find close open edges and stitches the surface along them surfaceHookUpDict 파일 사용

사용법

> surfaceHookUp [options] <hookTolerance>

사용 예

> surfaceHookUp 0.001

Example of surfaceHookUpDict - of-2.3

```
surface1.stl    {type triSurfaceMesh;}
surface2.stl    {type triSurfaceMesh;}
```

유틸리티 샐행 후 constant/triSurface/ 에 hookedSurface_surface1.stl, hookedSurface_surface2.stl 파일이 생성됨

22.12 surfaceInertia

기능

Calculates the inertia tensor, principal axes and moments of a command line specified triSurface. Inertia can either be of the solid body or of a thin shell.

사용법

> surfaceInertia [options] <surfaceFile>

옵션

- density <scalar> : Specify density, kg/m3 for solid properties, kg/m2 for shell properties
- referencePoint <vector> : Inertia relative to this point, not the centre of mass
- shellProperties : inertia of a thin shell

사용 예

> surfaceInertia aa.stl

실행 결과

```
Density: 1
Mass: 3.81692242653
Centre of mass: (-0.327178917926 -0.153075966523 -0.308033904217)
Surface area: 12.2005766287
Inertia tensor around centre of mass:
(1.50150822071 -8.82186717843e-08 9.77508444744e-08 -8.8218671784
3e-08 1.50150863257 5.13293667848e-08 9.77508444744e-08 5.1329366
7848e-08 1.39819923051)
eigenValues (principal moments): (1.39819923051 1.50150823249 1.5
0150862078)
eigenVectors (principal axes):
(-9.46199242918e-07 -4.9685168278e-07 0.999999999999)
(0.976541227374 0.215330516272 1.03098962001e-06)
(-0.215330453522 0.976541241211 2.81450793599e-07)

Writing scaled principal axes at centre of mass of "aa.stl" to "a
xes.obj"

End
```

22.13 surfaceLambdaMuSmooth

기능

Smooths a surface using lambda/mu smoothing.
To get laplacian smoothing, set lambda to the relaxation factor and mu to zero.
Provide an edgeMesh file containing points that are not to be moved during smoothing in order to preserve features.
lambda/mu smoothing : G. Taubin, IBM Research report Rc-19923 (02/01/95) "A signal processing approach to fair surface design"

사용법

> surfaceLambdaMuSmooth [options] <surfaceFile> <lambda (0..1)> <mu (0..1)> <iterations> <output surfaceFile>

옵션

- featureFile <file> : fix points from a file containing feature points and edges

사용 예

> ???

22.14 surfaceMeshConvert

기능

Converts between surface formats with optional scaling or transformations (rotate/translate) on a coordinateSystem.
The filename extensions are used to determine the file format type.

사용법

> surfaceMeshConvert [options] <inputFile> <outputFile>

옵션

- clean : perform some surface checking/cleanup on the input surface
- dict <file> : read control dictionary from specified location
- from <system> : specify the source coordinate system, applied after '-scaleIn'
- scaleIn <factor> : geometry scaling factor on input
- scaleOut <factor> : geometry scaling factor on Output
- to <system> : specify the target coordinate system, applied before '-scaleOut'
- tri : triangulate surface

사용 예

> surfaceMeshConvert aa.stl bb.obj ???

Example of coordinateSystems - of-2.3

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       IOPtrList<coordinateSystem>;
    object      coordinateSystems;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

7
(
system_9
{
    type            cartesian;
```

```
    origin          (1.03291515 -0.114391257 -0.0826236662);
    e3              (1 0 0);
    e1              (0 1 0);
    // STARCDRotation  (0 90 90);
}

system_10
{
    type            cartesian;
    origin          (0.623151719 -0.286472935 -0.113933262);
    e3              (0.99508851 0.09829095 0.01173645);
    e1              (0.01179356 0 -0.99993045);
    // STARCDRotation  (5.6403745 -0.0664172952 89.3275351);
}

system_15
{
    type            cartesian;
    origin          (0.644772231 -0.240036493 0.155972187);
    e3              (-0.01346388 -0.90616979 -0.42269969);
    e1              (0.00627978 0.42265304 -0.90626981);
    // STARCDRotation  (-90.8512386 0 115.005148);
}

system_16
{
    type            cartesian;
    origin          (0.540824938 -0.240036415 0.15928296);
    e3              (-0.01346388 -0.90616979 -0.42269969);
    e1              (0.00627978 0.42265304 -0.90626981);
    // STARCDRotation  (-90.8512386 0 115.005148);
}

system_17
{
    type            cartesian;
    origin          (0.436877646 -0.240036339 0.162593737);
    e3              (-0.01346388 -0.90616979 -0.42269969);
    e1              (0.00627978 0.42265304 -0.90626981);
    // STARCDRotation  (-90.8512386 0 115.005148);
}

system_18
{
    type            cartesian;
    origin          (0.332930354 -0.240036261 0.16590451);
    e3              (-0.01346388 -0.90616979 -0.42269969);
    e1              (0.00627978 0.42265304 -0.90626981);
    // STARCDRotation  (-90.8512386 0 115.005148);
}
```

```
system_21
{
    type            cartesian;
    origin          (0.55863733 -0.300866705 0.00317260982);
    e3              (0.42110287 0.02470132 -0.90667647);
    e1              (0.90646036 0.02342535 0.42164069);
    // STARCDRotation  (-178.185897 -0.71772221 -155.059695);
}
)
```

22.15 surfaceMeshConvertTesting

Converts from one surface mesh format to another, but primarily used for testing functionality.

22.16 surfaceMeshImport

기능

Import from various third-party surface formats into surfMesh with optional scaling or transformations (rotate/translate) on a coordinateSystem.

사용법

> surfaceMeshImport [options] <inputFile>

옵션

- clean : perform some surface checking/cleanup on the input surface
- dict <file> : read control dictionary from specified location
- from <coordinateSystem> : specify a local coordinate system when reading files
- to <coordinateSystem> : specify a local coordinate system when writing files
- name <name> : specify an alternative surface name when writing - default is 'default'
- scaleIn <factor> : geometry scaling factor on input -default is 1
- scaleOut <factor> : geometry scaling factor on Output - default is 1

사용 예

> surfaceMeshInport -name cc bb.stl : 0/surfaces/cc folder is generated

22.17 surfaceMeshExport

기능

Export from surfMesh to various third-party surface formats with optional scaling or transformations (rotate/translate) on a coordinateSystem.

사용법

> surfaceMeshExport [options] <outputFile>

옵션

- clean : perform some surface checking/cleanup on the input surface
- dict <file> : read control dictionary from specified location
- from <coordinateSystem> : specify the source coordinate system, applied after '-scaleIn'
- to <coordinateSystem> : specify the target coordinate system, applied before '-scaleOut'
- name <name> : specify an alternative surface name when reading - default is 'default'
- scaleIn <factor> : geometry scaling factor on input -default is 1
- scaleOut <factor> : geometry scaling factor on Output - default is 1

사용 예

> surfaceMeshExport -name cc bb.stl : 0/surface/cc 폴더의 surface mesh를 읽어 현재 폴더에 bb.stl이라는 파일을 쓴다.

22.18 surfaceMeshInfo

기능

Miscellaneous information about surface meshes.

사용법

> surfaceMeshInfo [options] <surfaceFile>

옵션

- areas : display area of each face
- scale <factor> : geometry scaling factor - default is 1
- xml : write output in XML format

사용 예

> surfaceMeshInfo aa.stl

실행 결과

```
nPoints    : 1538
nFaces     : 3072
area       : 12.2006
```

22.19 surfaceMeshTriangulate

기능

Extracts surface from a polyMesh. Depending on output surface format triangulates faces. Region numbers on faces cannot be guaranteed to be the same as the patch indices.
Optionally only triangulates named patches.
If run in parallel the processor patches get filtered out by default and the mesh gets merged (based on topology).

사용법

> surfaceMeshTriangulate [options] <output file>

옵션

- exclusiveProcPatches : exclude processor patches
- faceZones <(fz0 .. fzN)> : triangulate selected faceZones (wildcards supported)
- patches <(patch0 .. patchN)> : only triangulate selected patches (wildcards supported)

사용 예

> surfaceMeshTriangulate aa.stl

22.20 surfaceOrient

기능

Set normal consistent with respect to a user provided 'outside' point. If the -inside option is used the point is considered inside.

사용법

> surfaceOrient [options] <surfaceFile> <visiblePoint> <output surfaceFile>

옵션

- inside : treat provided point as being inside
- usePerceTest : determine orientation by counting number of intersections

사용 예

> surfaceOrient aa.stl '(10 0 0)' oo.stl

22.21 surfacePointMerge

기능

Merges points on surface if they are within absolute distance. Since absolute distance use with care!

사용법

> surfacePointMerge [options] <surfaceFile> <merge distance> <output surfaceFile>

사용 예

> surfacePointMerge aa.stl 0.001 bb.stl

22.22 surfaceRedistributePar

기능

(Re)distribution of triSurface. Either takes an undecomposed surface or an already decomposed surface and redistributes it so that each processor has all triangles that overlap its mesh.

Note

- best decomposition option is hierarchGeomDecomp since guarantees square decompositions.
- triangles might be present on multiple processors.
- merging uses geometric tolerance so take care with writing precision.

사용법

> surfaceRedistributePar [options] <triSurfaceMesh> <distributionType>
distributionType : follow, fronzen, independent

옵션

- keppNonMapped : preserve surface outside of mesh bounds

사용 예

> ????

22.23 surfaceRefineRedGreen

기능

Refine by splitting all three edges of triangle ('red' refinement). Neighbouring triangles (which are not marked for refinement get split in half ('green' refinement). (R. Verfuerth, "A review of a posteriori error estimation and adaptive mesh refinement techniques", Wiley-Teubner, 1996)

사용법

> surfaceRefineRedGreen [options] <surfaceFile> <output surfaceFile>

사용 예

> surfaceRefineRedGreen aa.stl bb.stl

22.24 surfaceSplitByPatch

기능

Writes regions of triSurface to separate files.

사용법

> surfaceSplitByPatch [options] <input surfaceFile>

사용 예

> surfaceSplitByPatch aa.stl

22.25 surfaceSplitByTopology

기능

Strips any baffle parts of a surface. A baffle region is one which is reached by walking from an open edge, and stopping when a multiply connected edge is reached.

사용법

> surfaceSplitByTopology [options] <input surfaceFile> <output surfaceFile>

사용 예

> surfaceSplitByTopology aa.stl bb.stl

22.26 surfaceSplitNonManifolds

기능

Takes multiply connected surface and tries to split surface at multiply connected edges by duplicating points. Introduces concept of
- borderEdge. Edge with 4 faces connected to it.
- borderPoint. Point connected to exactly 2 borderEdges.
- borderLine. Connected list of borderEdges.

By duplicating borderPoints this will split 'borderLines'. As a preprocessing step it can detect borderEdges without any borderPoints and explicitly split these triangles.

The problems in this algorithm are:
- determining which two (of the four) faces form a surface. Done by walking face-edge-face while keeping and edge or point on the borderEdge borderPoint.
- determining the outwards pointing normal to be used to slightly offset the duplicated point.

Uses sortedEdgeFaces quite a bit.

Is tested on simple borderLines resulting from extracting a surface from a hex mesh. Will quite possibly go wrong on more complicated border lines (i.e. ones forming a loop).

Dumps surface every so often since might take a long time to complete.

사용법

> surfaceSplitNonManifolds [options] <surfaceFile> <output surfaceFile>

사용 예

> ???

22.27 surfaceSubset

기능

A surface analysis tool which sub-sets the triSurface to choose only a part of interest. Based on subsetMesh.

사용법

> surfaceSubset [options] <surfaceSubsetDict> <surfaceFile> <output surfaceFile>

사용 예

> surfaceSubset surfaceSubsetDict aa.stl bb.stl

Example of coordinateSystems - of-2.3

```
// Select triangles by label
faces #include "badFaces";

// Select triangles using given points (local point numbering)
localPoints ( );

// Select triangles using given edges
edges ();

// Select triangles (with face centre) inside box
zone
(
    (0      -10000  125)
    (10000   10000  10000)
);

// Select triangles (with face centre) inside or outside of another surface.
// (always selects triangles that are 'on' other surface)
surface
{
    name "sphere.stl";
    outside     yes;
}

// Select triangles on plane
plane
{
    planeType embeddedPoints;
    embeddedPointsDict
    {
        //point1 (-937.259845440205 160.865349115986 240.738791238078);
        //point2 (-934.767379895778 9.63875523747379 14.412359671298);
        //point3 (44.4744688899417 121.852927962709 182.352485273106);
        point1 (-957.895294591874 242.865936478961 162.286611511875);
        point2 (-961.43140327772 4.53895551562943 3.04159982093444);
        point3 (91.2414146173805 72.1504381996692 48.2181961945329);
    }

    // Distance from plane
    distance 0.1;
    // Normal difference to plane
    cosAngle 0.99;
}

// Extend selection with edge neighbours
```

```
addFaceNeighbours no;

// Invert selection
invertSelection false;
```

22.28 surfaceToPatch

기능

Reads surface and applies surface regioning to a mesh. Uses boundaryMesh to do the hard work.

사용법

> surfaceToPatch [options] <surfaceFile>

옵션

- faceSet <name> : only repatch the faces in specified faceSet

- tol <scalar> : search tolerance as fraction of mesh size (default 1e-3)

사용 예

> surfaceToPatch aa.stl

22.29 surfaceTransformPoints

기능

Transform (scale/rotate) a surface.
Like transformPoints but for surfaces.

The rollPitchYaw option takes three angles (degrees):
- roll (rotation about x) followed by
- pitch (rotation about y) followed by
- yaw (rotation about z)

The yawPitchRoll does yaw followed by pitch followed by roll.

사용법

> surfaceTransformPoints [options] <surfaceFile> <output surfaceFile>

옵션

- rollPitchYaw <vector> : transform in terms of '(roll pitch yaw)' in degrees
- yawPitchRoll <vector> : transform in terms of '(yaw pitch roll)' in degrees
- rotate <(vectorA vectorB)> : transform in terms of a rotation between <vectorA> and <vectorB> - eg, '((1 0 0) (0 0 1))'
- scale <vector> : scale by the specified amount - eg, '(0.001 0.001 0.001)' for a uniform [mm] to [m] scaling
- translate <vector> : translate by the specified <vector> - eg, '(1 0 0)'

사용 예

> transformPoints -scale '(0.001 0.001 0.001)' aa.stl

23 miscellaneous

23.1 expandDictionary

기능

Read the dictionary provided as an argument, expand the macros etc. and write the resulting dictionary to standard output.

Note

The -list option can be useful when determining which files are actually included by a directory. It can also be used to determine which files may need to be copied when transferring simulation to another environment. The following code snippet could be a useful basis for such cases:

```
for i in . 0 constant system
do
    find $i -maxdepth 1 -type f -exec expandDictionary -list '{}' \;
done | sed -ne '/^"\//!{ s/^"//; s/"$//; p }' | sort | uniq
```

사용법

> expandDictionary [options] <inputDict>

옵션

- list : Report the #include/#includeIfPresent to stdout only

사용 예

> ???

23.2 foamDebugSwitches

기능

Write out all library debug switches.

사용법

> foamDebugSwitches [options]

옵션

- new : output switches that are known from the libraries but that do not seem to be known in the current etc/controlDict
- old : output switches that appear to be unknown in the current etc/controlDict

사용 예

> ???

23.3 foamFormatConvert

기능

Converts all IOobjects associated with a case into the format specified in the controlDict.

Mainly used to convert binary mesh/field files to ASCII.

Problem: any zero-size List written binary gets written as '0'. When reading the file as a dictionary this is interpreted as a label. This is (usually) not a problem when doing patch fields since these get the 'uniform', 'nonuniform' prefix. However zone contents are labelLists not labelFields and these go wrong. For now hacked a solution where we detect the keywords in zones and redo the dictionary entries to be labelLists.

사용법

> foamFormatConvert [options]

옵션

- noConstant : exclude the 'constant/' dir in the times list
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option

사용 예

> foamFormatConvert

23.4 foamHelp

Top level wrapper utility around foam help utilities

23.5 foamInfoExec

기능

Interrogates a case and prints information to stdout.

사용법

> foamInfoExec [options]

옵션

- dict <file> : read control dictionary from specified location
- entry <name> : report the named entry for the specified dictionary
- keywords : report keywords for the specified dictionary
- times : list available times

사용 예

> ???

23.6 patchSummary

기능

Writes fields and boundary condition info for each patch at each requested time instance.

Default action is to write a single entry for patches/patchGroups with the same boundary conditions. Use the -expand option to print every patch separately. In case of multiple groups matching it will print only the first one.

사용법

> patchSummary [options]

옵션

- expand : Do not combine patches
- noZero : exclude the '0/' dir from the times list, has precedence over the -zeroTime option

사용 예

> patchSummary -time 0 > log