Strand Mesh Format to be Used with `AVMesh` Library

Aaron Katz

8 April 2010

The data representation of strand grids should be kept compact. Following the conventions of the `CREATE-AV` mesh library, `AVMesh`, the strand grid description is organized into a strand specific mesh header followed by the actual strand mesh data. Actually, the strand specific mesh header is composed of three modules: *strand_header*, *strand_surf_patch*, and *strand_edge_patch*, all of which are listed below in Figures 1-3. The actual strand grid data is shown in Figure 4 in the module *strand_mesh_data*.

The strand specific mesh header, *strand_header*, contains the following data, shown in Figure 1. Note that the last five items in the list below are all the parameters used to generate the strand mesh. They are stored here, so that the mesh can be reproduced exactly, given the original surface mesh.

Since the strand header data will remain the same for 2D meshes, certain entries will be set to 0 or change meaning for 2D data. These changes are explained along with the 3D descriptions, when applicable.

- *surfaceOrVolume*: character which determines if the following grid data will be only the surface mesh, or the entire volume mesh description. Possible values are "surface" or "volume."
- *nSurfNodes*: number of surface nodes.
- *nTriFaces*: number of triangular surface faces. In 2D, the number of surface edges.
- *nQuadFaces*: number of quadrilateral surface faces. Set to zero for 2D meshes.
- *nBndEdges*: number of boundary edges. Note this is only the number of edges lying on an open boundary, such as a symmetry plane, and not internal edges of the surface description. In 2D, this number is the number of boundary nodes (which is either 0 for a closed set of edges like an airfoil, or 2 for an open set of edges like a flat plate.
- *nPtsPerStrand*: number of points along each strand. This number is used to describe all strands.

- *nSurfPatches*: number of surface patches.
- *nEdgePatches*: number of edge patches.
- *targetReynoldsNumber*: target Reynolds number
- *reynoldsRefLength*: reference length for target Reynolds number
- *deltaSpacing*: spacing of first mesh point off the body.
- *strandLength*: length of strands.
- *stretchRatio*: maximum stretching ratio between successive strand nodes.
- *smoothingThreshold*: threshold to determine when to stop strand pointing vector smoothing iterations.
- *strandDistribution*: description of the strand stretching method. Currently either "geometric" or "hypTan".

The body surface is composed of surface patches, which are used to assign boundary conditions. Surface patch information is expressed in the form of cell tags so that the patch assignment of surface regions is unambiguous. Each of the *nSurfPatches* surface patches has associated with it the follwing information, contained in the *strand_surf_patch* module, shown in Figure 2.

- *surfPatchID*: number of the surface patch appearing in the *faceTag* array.
- *surfPatchBody*: body to which the patch belongs
- *surfPatchComp*: component to which the patch belongs
- *surfPatchBCType*: boundary type of the patch

Additionally, for open surfaces that are not water-tight (eg. flat plate or geometries with symmetry planes), the surface grid will have open edge boundaries. These edge boundaries take the form of edge patches, of which there are *nEdgePatches*. Edge patch information is expressed in the form of edge tags so that the patch assignment of edge boundary regions is unambiguous. Each of the *nEdgePatches* edge patches has associated with it the following information, contained in the *strand_edge_patch* module, shown in Figure 3.

- *edgePatchID*: number of the edge patch appearing in the *edgeTag* array.
- *edgePatchBody*: body to which the patch belongs
- *edgePatchComp*: component to which the patch belongs
- *edgePatchBCType*: boundary type of the patch

- *nx,ny,nz*: unit outward normal of the face created by the extrusion of the boundary edge. Note that edge patches are required to be produces planar boundaries, but this should not be overly-restrictive. For 2D, textit(nz) is set to zero.

The actual mesh data is contained in the module *strand_mesh_data*, shown in Figure 4, and described below. Only 3D mesh data is shown. The 2D mesh desciption will be formulated shortly. For surface only meshes, only the first six arrays (through and including *xSurf*) will be present.

- *triConn*: three nodes forming each surface triangle. Dimension is *(3,nTriFaces)*.
- *quadConn*:four nodes forming each surface quadrilateral. Dimension is *(4,nQuadFaces)*.
- *bndEdges*: two nodes forming each boundary edge. Dimension is *(2,nBndEdges)*.
- *faceTag*: surface patch number for each face, where triangular faces are listed first, followed by quadrilateral faces. Dimension is *(nTriFaces+nQuadFaces)*.
- *edgeTag*: edge patch number for each boundary edge. Dimension is *(nBndEdges)*.
- *xSurf*: surface node locations. Dimension is *(3,nSurfNodes)*.
- *nodeClip*: clipping index for each node. Dimension is *(nSurfNodes)*.
- *faceClip*: clipping index for each face, where triangular faces are listed first, followed by quadrilateral faces. Dimension is *(nTriFaces+nQuadFaces)*.
- *pointingVec*: unit pointing vector for each surface node. Dimension is *(3,nSurfNodes)*.
- *xStrand*: coordinates of one-dimensional strand distribution. Dimension is *(nPtsPerStrand)*.

```
MODULE strand_header

IMPLICIT NONE

CHARACTER(32) :: surfaceOnly
INTEGER       :: nSurfNodes
INTEGER       :: nTriFaces
INTEGER       :: nQuadFaces
INTEGER       :: nBndEdges
INTEGER       :: nPtsPerStrand
INTEGER       :: nSurfPatches
INTEGER       :: nEdgePatches
REAL*8        :: targetReynoldsNumber
REAL*8        :: reynoldsRefLength
REAL*8        :: deltaSpacing
REAL*8        :: strandLength
REAL*8        :: stretchRatio
REAL*8        :: smoothingThreshold
CHARACTER(32) :: strandDistribution

END MODULE strand_header
```

Figure 1: FORTRAN 90 module containing strand mesh specific header data.

```
MODULE strand_surf_patch

IMPLICIT NONE

INTEGER       :: surfPatchID
CHARACTER(32) :: surfPatchBody
CHARACTER(32) :: surfPatchComp
CHARACTER(32) :: surfPatchBCType

END MODULE strand_surf_patch
```

Figure 2: FORTRAN 90 module containing surface patch data.

```
MODULE strand_edge_patch

IMPLICIT NONE

INTEGER          :: edgePatchID
CHARACTER(32)    :: edgePatchBody
CHARACTER(32)    :: edgePatchComp
CHARACTER(32)    :: edgePatchBCType
REAL*8           :: nx,ny,nx

END MODULE strand_edge_patch
```

Figure 3: FORTRAN 90 module containing edge patch data.

```
MODULE strand_mesh_data

IMPLICIT NONE

INTEGER,ALLOCATABLE,DIMENSION(:,:)  :: triConn
INTEGER,ALLOCATABLE,DIMENSION(:,:)  :: quadConn
INTEGER,ALLOCATABLE,DIMENSION(:,:)  :: bndEdges
INTEGER,ALLOCATABLE,DIMENSION(:   )  :: faceTag
INTEGER,ALLOCATABLE,DIMENSION(:   )  :: edgeTag
REAL*8  ,ALLOCATABLE,DIMENSION(:,:)  :: xSurf
INTEGER,ALLOCATABLE,DIMENSION(:   )  :: nodeClip
INTEGER,ALLOCATABLE,DIMENSION(:   )  :: faceClip
REAL*8  ,ALLOCATABLE,DIMENSION(:,:)  :: pointingVec
REAL*8  ,ALLOCATABLE,DIMENSION(:   )  :: xStrand

END MODULE strand_mesh_data
```

Figure 4: FORTRAN 90 module containing actual strand mesh data.