

新TreeFoamの紹介

(TreeFoam ver 3.31.250809)

<修正内容>

1. remeshAndMapping.pyコマンド追加
dynamicMesh使用時に潰れたmeshをremesh、mappingして計算継続
2. meshViewerに時間進む、戻るボタンを追加
dynamicMesh使用時にメッシュ変化の確認を容易にする為
3. tutorials表示のバグ修正
4. まとめ

25/08/09 藤井

1. remeshAndMapping.pyコマンド追加

1-1. コマンドの内容

dynamicMeshを使った計算で、メッシュが潰れて計算不能に陥った時、
メッシュを切り直して、旧メッシュのfieldを新メッシュのfieldにmappingして計算を継続。
→ リメッシュ、マッピングを行うコマンド

caseDir：解析用case

- ・ tutorialsからdynamicMeshのcase (例:movingCone) をコピーして、caseを完成させる。
- ・ 計算条件を設定。meshを変化させながら算開始。
- ・ meshが潰れ計算不能直前のtimeFolder内から

meshの全patch形状をstl形式で出力。

fieldデータ

- ・ mapping結果を受取、再計算開始

tempCaseDir：メッシュ作成用case

- ・ stlを準備してsnappyHexMeshでメッシュを作成

- ・ 新しいstlを使ってメッシュを作成

- ・ 新しいmesh上でfieldデータをmapping

- ・ mappingしたfieldを解析caseに戻す

meshコピー

stlをコピー

fieldをコピー

mappingを戻す

patch形状をstl形式で出力する方法
以下のコマンドを使って、全patchのstlを取得している。

0F-9以上 :surfaceMeshTriangulate
0F-v2406以上 :foamToSurface -tri

(0F-8以前、0F-v2306以前は、未確認)

remeshAndMapping.pyの処理内容

1-2. コマンド利用方法

caseDir内からFOAM端末を起動し、以下の様に入力して実行する。

```
$ remeshAndMapping.py --tempCase ../movingValve_mesh
```

「../movingValve_mesh」がtempCaseDirになる。実行後は、直ぐにリスタートできる。

コマンドのhelpを出力

latestTimeにて、patch形状をstl形式で読み取り、snappyHexMeshでメッシュを切り直す。remesh後、field内容を新meshにmappingする。remeshする為には、予めsnappyHexMeshでメッシュを作成するtempcaseを準備しておく。

```
remeshAndMapping.py [option]
```

<option>

```
-c, --case      :caseDirを指定 (省略時は、currDir)
-t, --tempCase  :tempCaseDirを指定
                  このcaseでmesh作成、mappingを行う。
-i, --iniTime   :mappingするfieldを取得するtimeFolder
                  計算開始時のtimeFolderを指定する。
                  (省略時は、firstTime)
-m, --mapTime   :mappingの元dataを取得するtimeFolder
                  (省略時は、latestTime)
-h, --help      :helpを表示
```

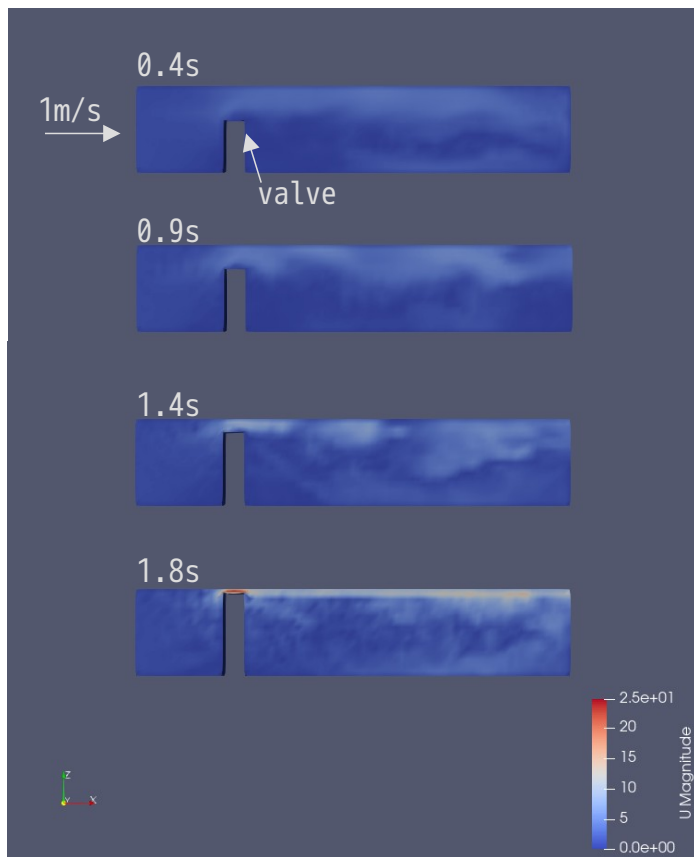
<使用例>

```
remeshAndMapping.py
:引数省略時は、「../<currDirのbasename>_mesh」をtempCaseとして設定する。
remeshAndMapping.py ../test_mesh
:「../test_mesh」をtempCaseとして設定。
remeshAndMapping.py -t ../test_mesh
:「../test_mesh」をtempCaseとして設定。
```

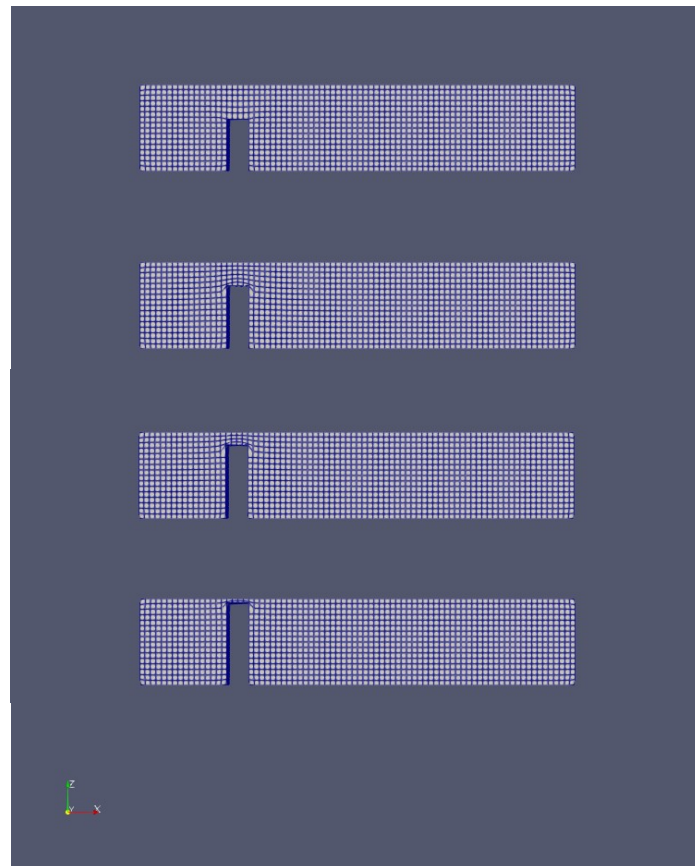
1-2. コマンド実行例

tutorials内の「movingCone」をアレンジしてcaseを作成。（詳細は、操作マニュアル「9-3-2項」参照）
valveを上昇させて流路を狭める。リメッシュしながら1.8sまで計算を継続。

<モデル> 流入速度:1m/s, valve上昇速度:0.01m/s

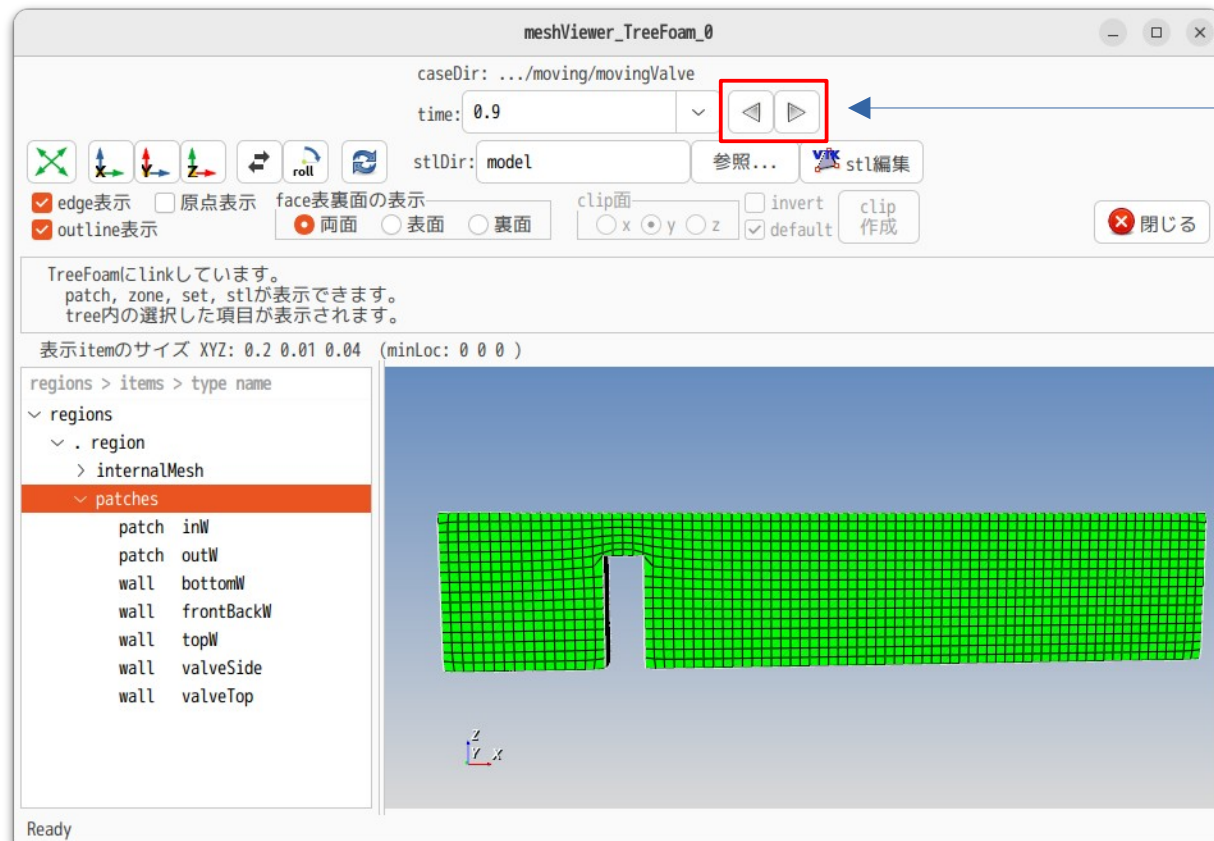


<メッシュ>



2. meshViewerに時間進む、戻るボタンを追加

dynamicMeshを使った解析で、メッシュの変化が容易に確認できるように
「時間進める」「時間戻す」ボタンを追加。



クリックする事で、
1stepの時間を進める、戻す
事ができる。

3. tutorials表示のバグ修正

OF-13対応でTreeFoamを修正したが、この修正によって、OF-13以外でエラーが発生。

OF-13は、system/controlDict内に「application」の項目が無くなった。

TreeFoamでは、application項目の内容を確認し、solverを起動している為、

applicationが未設定の場合、folder構成やcontrolDictの内容からapplicationを設定するように修正した。

OF-13以外では、tutorials内の全caseにおいて、application未設定のcaseは、存在しないと思っていたが、tutorials/mesh内のcaseには、application未設定のcaseが存在する。

このcase読み取り時に、エラーが発生していた。→ バグ

上記内容を修正した。

4. まとめ

「remeshAndMapping.py」 コマンドは、
流体 - 構造連成解析 (FSI) において、
メッシュが潰れても、リメッシュとマッピングを繰り返しながら、計算が継続できないか
検討している時にできあがった。

現状のFSIは、motionSolverとして「displacementLaplacian」を使って計算している。
field「pointDisplacement (変位)」の値を構造側から受け取って流体側に反映している。
変位は、初期位置からの移動量になるので、途中でリメッシュすると、初期位置が変わってしまう。
→ トライしたが、リメッシュは難しい。

今回のcaseでは、motionSolverとして「velocityComponentLaplacian(velocityLaplacian)」を使っている。
field「pointMotionUz (速度)」を使って、meshを移動させている。
初期位置は、関係なくその時々でmeshの座標を変化させている。
→ 今回の様にリメッシュが容易。

solver : pimpleFoam、motionSolver : velocityLaplacianの組み合わせで、FSI解析トライしてみたい。