APPENDICES

Numerical study of flow through a Savonius wind turbine

June 12th, 2015

Grau en Enginyeria en Vehicles Aeroespacials
Fluids Mechanics Department
ETSEIAT-UPC

Pere Antoni Martorell Pol

Tutor: **Daniel García Almiñana**

Director: **Robert Castilla López**

Co-director: **Pedro Javier Gámez Montero**

Escola Tècnica Superior d'Enginyeries
Industrial i Aeronàutica de Terrassa
UNIVERSITAT POLITÈCNICA DE CATALUNYA

Contents

List of Tables

List of Figures

Appendix A

Codes used in the development

The codes of the of the cases are extracted from the tutorials available on openFOAM. However, all the bash scripts as well as the way to generate the directories are fully designed and typed by the author of the project.

A.1 Cylinder

A.1.1 Summary

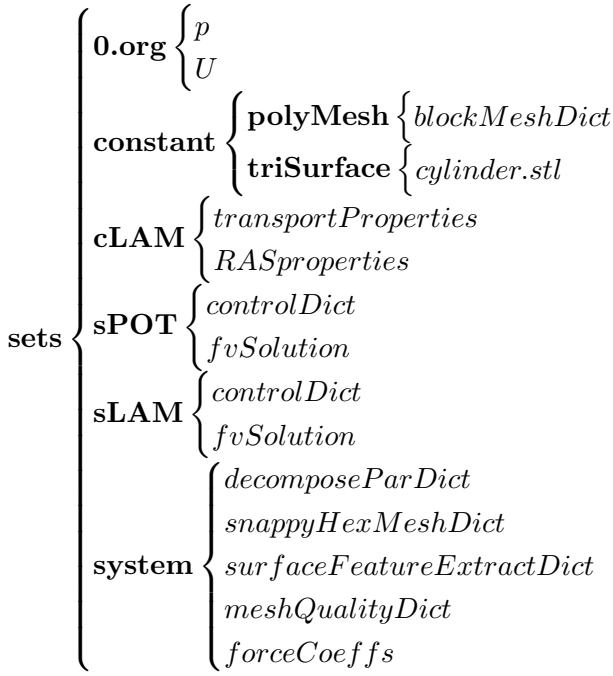
Directory tree

The potential and laminar case of the cylinder is composed by the following folders:

- **CYLINDER**
  - **0**
    - p
    - U
  - **constant**
    - **polyMesh**
      - $blockMeshDict$
    - **triSurface**
      - $cylinder.stl$
    - $transportProperties$
    - $RASproperties$
  - **system**
    - $controlDict$
    - $fvSolution$
    - $fvSchemes$
    - $decomposeParDict$
    - $snappyHexMeshDict$
    - $surfaceFeatureExtractDict$

Directory creation

In order to run several cases only changing some parameters has been created a folder called *org* that contains the *hostfile*, the bash scripts *RUN*, *potential* and *SOLVE*, and the folder *sets* that contain all the information of the case. The structure of this foldes is as the follows.



Bash scripts

The *Folders* bash file create a folder for each Reynolds:

```
#!/bin/bash
for i in 1 20 25 50 100 250 300 1000 1000;
do
        f=Re$i
        cp Re0.1 $f -r
done
```

Then the ν has been changed in order to obtain each Reynolds number desired, is ran *Execute*.

```
#!/bin/bash
for i in Re*;
do
        cd $i
                ./SOLVE
        cd ..
done
```

The bash script *RUN* is:

```
#!/bin/bash
mkdir POT LAM
#Potential
cp sets/constant sets/0.org sets/system POT -R
cp sets/sPOT/controlDict sets/sPOT/fvSchemes sets/sPOT/fvSolution POT/system
```

```
#LAMINAR
cp sets/system LAM -R
cp sets/sLAM/controlDict sets/sLAM/fvSchemes sets/sLAM/fvSolution LAM/system
cp sets/cLAM/RASProperties sets/cLAM/transportProperties LAM/constant
```

The bash script *potential* is:

```
#!/bin/bash
cd POT
rm 0 -r
cp 0.org 0 -R
blockMesh > log.blockMesh
checkMesh > log.checkMesh
surfaceFeatureExtract > log.surfaceFeatureExtract
decomposePar >log.decomposeParMesh
mpirun -np 4 snappyHexMesh -parallel -overwrite> log.snappyHexMesh
reconstructParMesh -constant -mergeTol 1e-6 >log.reconstructParMesh
rm -R processor* 0
cp 0.org 0 -R
checkMesh -allGeometry > log.checkMeshAll
potentialFoam > log.potentialFoam
paraFoam -builtin -touch
cd ..
```

The bash script *SOLVE* is:

```
#!/bin/bash
cd LAM
decomposePar >log.decomposePar
mpirun -np 4 simpleFoam -parallel > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
```

A.1.2 0

The boundary conditions defined are U:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
```

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (10 0 0);

boundaryField
{

    inlet
        {
                type                          freestream;
                freestreamValue         uniform (10 0 0);
        }
    outlet
    {
        type                           freestream;
                freestreamValue         uniform (10 0 0);
    }

         upperWall
        {
                type                          slip;
        }

        lowerWall
        {
                type                          slip;
        }

        frontAndBack
        {
                type                          slip;
        }


   cylinder
    {
```

```
        type            fixedValue;
        value           uniform (0 0 0);
    }
}
```

```
// ************************************************************************* //
```

The pressure is defined in p.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{

    inlet
    {
        type            freestreamPressure;
    }

    outlet
    {
        type            freestreamPressure;
    }
        upperWall
        {
                type                        slip;
```

```
        }
        lowerWall
        {
                type                        slip;
        }

        frontAndBack
        {
                type                        slip;
        }

    cylinder
    {
        type            zeroGradient;
    }


}

// ************************************************************************* //
```

A.1.3 constant

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1;

vertices
(
    (-2 -1.5 -1.5)
    (4 -1.5 -1.5)
    (4 1.5 -1.5)
    (-2 1.5 -1.5)
```

```
    (-2 -1.5 1.5)
    (4 -1.5 1.5)
    (4 1.5 1.5)
    (-2 1.5 1.5)
);


blocks
(
    hex (0 1 2 3 4 5 6 7) (50 25 25) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    frontAndBack
    {
        type patch;
        faces
        (
            (3 7 6 2)
            (1 5 4 0)
        );
    }
    inlet
    {
        type patch;
        faces
        (
            (0 4 7 3)
        );
    }
    outlet
    {
        type patch;
        faces
        (
            (2 6 5 1)
        );
    }
    lowerWall
    {
        type patch;
        faces
```

```
        (
            (0 3 2 1)
        );
    }
    upperWall
    {
        type patch;
        faces
        (
            (4 5 6 7)
        );
    }
);

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 2; //It depend on the Reynolds number

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
```

```
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            laminar;

turbulence          off;

printCoeffs         on;


// ************************************************************************* //
```

A.1.4 system

Potential

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     potentialFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         2;

deltaT          1;
```

```
writeControl    timeStep;

writeInterval   1;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    difference
    {
        // Load the library containing the 'coded' functionObject
        functionObjectLibs ("libutilityFunctionObjects.so");
        type coded;
        // Name of on-the-fly generated functionObject
        redirectType error;
        code
        #{
            // Lookup U
            Info<< "Looking up field U\n" << endl;
            const volVectorField& U = mesh().lookupObject<volVectorField>("U");

            Info<< "Reading inlet velocity  uInfX\n" << endl;

            scalar ULeft = 0.0;
            label leftI = mesh().boundaryMesh().findPatchID("inlet");
            const fvPatchVectorField& fvp = U.boundaryField()[leftI];
            if (fvp.size())
            {
                ULeft = fvp[0].x();
            }
            reduce(ULeft, maxOp<scalar>());

            dimensionedScalar uInfX
            (
                "uInfx",
```

```
    dimensionSet(0, 1, -1, 0, 0),
    ULeft
);

Info << "U at inlet = " << uInfX.value() << " m/s" << endl;


scalar magCylinder = 0.0;
label cylI = mesh().boundaryMesh().findPatchID("cylinder");
const fvPatchVectorField& cylFvp = mesh().C().boundaryField()[cylI];
if (cylFvp.size())
{
    magCylinder = mag(cylFvp[0]);
}
reduce(magCylinder, maxOp<scalar>());

dimensionedScalar radius
(
    "radius",
    dimensionSet(0, 1, 0, 0, 0),
    magCylinder
);

Info << "Cylinder radius = " << radius.value() << " m" << endl;

volVectorField UA
(
    IOobject
    (
        "UA",
        mesh().time().timeName(),
        U.mesh(),
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    U
);

Info<< "\nEvaluating analytical solution" << endl;

const volVectorField& centres = UA.mesh().C();
volScalarField magCentres(mag(centres));
volScalarField theta(acos((centres & vector(1,0,0))/magCentres));

volVectorField cs2theta
(
    cos(2*theta)*vector(1,0,0)
```

```
              + sin(2*theta)*vector(0,1,0)
            );

            UA = uInfX*(dimensionedVector(vector(1,0,0))
              - pow((radius/magCentres),2)*cs2theta);

            // Force writing of UA (since time has not changed)
            UA.write();

            volScalarField error("error", mag(U-UA)/mag(UA));

            Info<<"Writing relative error in U to " << error.objectPath()
                << endl;

            error.write();
        #};
    }
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         leastSquares;
}
```

```
divSchemes
{
    default         none;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
```

```
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }
}

potentialFlow
{
    nNonOrthogonalCorrectors 3;
        pRefPoint        (0.2546565 0.54651354 -0.15466496);
        pRefValue        0;
}


// ************************************************************************* //
```

Laminar

```
/*---------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs
(
    "libOpenFOAM.so"
    "libincompressibleTurbulenceModel.so"
    "libincompressibleRASModels.so"
);

application     simpleFoam;

startTime       0;
```

```
stopAt          endTime;

endTime         0.1;

deltaT          1e-4;

writeControl    adjustableRunTime;

writeInterval   0.01;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;


//- Uncomment to have regular (every 2 hours of run time) restart files
//secondaryWriteControl    cpuTime; // runtime
//secondaryWriteInterval   7200;    // seconds
//secondaryPurgeWrite      1;       // keep all but last dump


writeFormat     binary;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;
```

```
functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         Gauss linear;
    grad(U)         cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss linearUpwindV grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,omega)  bounded Gauss upwind;
           div((nuEff*dev(T(grad(U))))) Gauss linear;
}
```

```
laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p;
}

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-7;
        relTol          0.01;
        smoother        GaussSeidel;
```

```
        nPreSweeps       0;
        nPostSweeps      2;
        cacheAgglomeration on;
        agglomerator     faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels      1;
    }

    U
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
        relTol           0.1;
        nSweeps          1;
    }

    k
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
        relTol           0.1;
        nSweeps          1;
    }

    omega
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
        relTol           0.1;
        nSweeps          1;
    }


}
SIMPLE
{
    nNonOrthogonalCorrectors 0;
        pRefPoint        (0.215648 0.364514 -0.2651545);
        pRefValue        0;

}

potentialFlow
{
```

```
    nNonOrthogonalCorrectors 10;
}

relaxationFactors
{
    fields
    {
        p               0.3;
    }
    equations
    {
        U               0.7;
        k               0.7;
                omega           0.7;
    }
}

cache
{
    grad(U);
}

// ************************************************************************* //
```

Both

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      decomposeParDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

numberOfSubdomains 4;

method          hierarchical;
// method          ptscotch;
```

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

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

manualCoeffs
{
    dataFile        "cellDecomposition";
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      snappyHexMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Which of the steps to run
castellatedMesh true;
snap            true;
addLayers       true;


// Geometry. Definition of all surfaces. All surfaces are of class
// searchableSurface.
// Surfaces are used
// - to specify refinement for any mesh cell intersecting it
// - to specify refinement for any mesh cell inside/outside/near
```

```
// - to 'snap' the mesh boundary to the surface
geometry
{
    cylinder.stl
    {
        type triSurfaceMesh;
        name cylinder;
    }
    refinementBox
    {
        type searchableBox;
        min (-0.5 -0.5 -0.5);
        max ( 4  0.5 0.5);
    }
};



// Settings for the castellatedMesh generation.
castellatedMeshControls
{

    // Refinement parameters
    // ~~~~~~~~~~~~~~~~~~~~

    // If local number of cells is >= maxLocalCells on any processor
    // switches from from refinement followed by balancing
    // (current method) to (weighted) balancing before refinement.
    maxLocalCells 1000000;

    // Overall cell limit (approximately). Refinement will stop immediately
    // upon reaching this number so a refinement level might not complete.
    // Note that this is the number of cells before removing the part which
    // is not 'visible' from the keepPoint. The final number of cells might
    // actually be a lot less.
    maxGlobalCells 200000000;

    // The surface refinement loop might spend lots of iterations refining just a
    // few cells. This setting will cause refinement to stop if <= minimumRefine
    // are selected for refinement. Note: it will at least do one iteration
    // (unless the number of cells to refine is 0)
    minRefinementCells 10;

    // Allow a certain level of imbalance during refining
    // (since balancing is quite expensive)
    // Expressed as fraction of perfect balance (= overall number of cells /
    // nProcs). 0=balance always.
```

```
maxLoadUnbalance 0.10;


// Number of buffer layers between different levels.
// 1 means normal 2:1 refinement restriction, larger means slower
// refinement.
nCellsBetweenLevels 2;



// Explicit feature edge refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies a level for any cell intersected by its edges.
// This is a featureEdgeMesh, read from constant/triSurface for now.
features
(
    {
      file "cylinder.eMesh";
      level 5;
   }
);



// Surface based refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies two levels for every surface. The first is the minimum level,
// every cell intersecting a surface gets refined up to the minimum level.
// The second level is the maximum level. Cells that 'see' multiple
// intersections where the intersections make an
// angle > resolveFeatureAngle get refined up to the maximum level.

refinementSurfaces
{
    cylinder
    {
        // Surface-wise min and max refinement level
        level (2 4);

        // Optional specification of patch type (default is wall). No
        // constraint types (cyclic, symmetry) etc. are allowed.
       // patchInfo
        //{
          //  type wall;
            //inGroups (savoniusGroup);
```

```
            //}
        }
    }

    // Resolve sharp angles
    resolveFeatureAngle 30;


    // Region-wise refinement
    // ~~~~~~~~~~~~~~~~~~~~~~

    // Specifies refinement level for cells in relation to a surface. One of
    // three modes
    // - distance. 'levels' specifies per distance to the surface the
    //   wanted refinement level. The distances need to be specified in
    //   descending order.
    // - inside. 'levels' is only one entry and only the level is used. All
    //   cells inside the surface get refined up to the level. The surface
    //   needs to be closed for this to be possible.
    // - outside. Same but cells outside.

    refinementRegions
    {
       refinementBox
        {
            mode inside;
            levels ((4 4));
        }
    }


    // Mesh selection
    // ~~~~~~~~~~~~~~

    // After refinement patches get added for all refinementSurfaces and
    // all cells intersecting the surfaces get put into these patches. The
    // section reachable from the locationInMesh is kept.
    // NOTE: This point should never be on a face, always inside a cell, even
    // after refinement.
    locationInMesh (1.123573824 0.000100005468436 -1.12345698754);


    // Whether any faceZones (as specified in the refinementSurfaces)
    // are only on the boundary of corresponding cellZones or also allow
    // free-standing zone faces. Not used if there are no faceZones.
    allowFreeStandingZoneFaces true;
}
```

```
// Settings for the snapping.
snapControls
{
    //- Number of patch smoothing iterations before finding correspondence
    //  to surface
    nSmoothPatch 3;

    //- Relative distance for points to be attracted by surface feature point
    //  or edge. True distance is this factor times local
    //  maximum edge length.
    tolerance 2.0;

    //- Number of mesh displacement relaxation iterations.
    nSolveIter 30;

    //- Maximum number of snapping relaxation iterations. Should stop
    //  before upon reaching a correct mesh.
    nRelaxIter 5;

    // Feature snapping

        //- Number of feature edge snapping iterations.
        //  Leave out altogether to disable.
        nFeatureSnapIter 10;

        //- Detect (geometric only) features by sampling the surface
        //  (default=false).
        implicitFeatureSnap true;

        //- Use castellatedMeshControls::features (default = true)
        explicitFeatureSnap false;

        //- Detect points on multiple surfaces (only for explicitFeatureSnap)
        multiRegionFeatureSnap false;
}



// Settings for the layer addition.
addLayersControls
{
    // Are the thickness parameters below relative to the undistorted
    // size of the refined cell outside layer (true) or absolute sizes (false).
    relativeSizes true;
```

```
// Per final patch (so not geometry!) the layer information
layers
{
    cylinder
    {
        nSurfaceLayers 5;
    }
}

// Expansion factor for layer mesh
expansionRatio 1.1;

// Wanted thickness of final added cell layer. If multiple layers
// is the
// thickness of the layer furthest away from the wall.
// Relative to undistorted size of cell outside layer.
// is the thickness of the layer furthest away from the wall.
// See relativeSizes parameter.
finalLayerThickness 0.5;

// Minimum thickness of cell layer. If for any reason layer
// cannot be above minThickness do not add layer.
// Relative to undistorted size of cell outside layer.
minThickness 0.05;

// If points get not extruded do nGrow layers of connected faces that are
// also not grown. This helps convergence of the layer addition process
// close to features.
// Note: changed(corrected) w.r.t 17x! (didn't do anything in 17x)
nGrow 0;

// Advanced settings

// When not to extrude surface. 0 is flat surface, 90 is when two faces
// are perpendicular
featureAngle 60;

// At non-patched sides allow mesh to slip if extrusion direction makes
// angle larger than slipFeatureAngle.
slipFeatureAngle 30;

// Maximum number of snapping relaxation iterations. Should stop
// before upon reaching a correct mesh.
nRelaxIter 3;

// Number of smoothing iterations of surface normals
```

```
    nSmoothSurfaceNormals 1;

    // Number of smoothing iterations of interior mesh movement direction
    nSmoothNormals 3;

    // Smooth layer thickness over surface patches
    nSmoothThickness 10;

    // Stop layer growth on highly warped cells
    maxFaceThicknessRatio 0.5;

    // Reduce layer growth where ratio thickness to medial
    // distance is large
    maxThicknessToMedialRatio 0.3;

    // Angle used to pick up medial axis points
    // Note: changed(corrected) w.r.t 17x! 90 degrees corresponds to 130 in 17x.
    minMedianAxisAngle 90;


    // Create buffer region for new layer terminations
    nBufferCellsNoExtrude 0;


    // Overall max number of layer addition iterations. The mesher will exit
    // if it reaches this number of iterations; possibly with an illegal
    // mesh.
    nLayerIter 50;
}



// Generic mesh quality settings. At any undoable phase these determine
// where to undo.
meshQualityControls
{
    #include "meshQualityDict"


    // Advanced

    //- Number of error distribution iterations
    nSmoothScale 4;
    //- amount to scale back displacement at error points
    errorReduction 0.75;
}
```

```
// Advanced

// Write flags
writeFlags
(
    scalarLevels
    layerSets
    layerFields     // write volScalarField for layer coverage
);


// Merge tolerance. Is fraction of overall bounding box of initial mesh.
// Note: the write tolerance needs to be higher than this.
mergeTolerance 1e-6;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      surfaceFeatureExtractDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

cylinder.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   150;
    }
```

```
    subsetFeatures
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;

        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      meshQualityDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Include defaults parameters from master dictionary
#include "$WM_PROJECT_DIR/etc/caseDicts/meshQualityDict"

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


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
```

```
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/

forceCoeffs1
{
    type        forceCoeffs;

    functionObjectLibs ( "libforces.so" );

    outputControl   timeStep;
    timeInterval    ;

    log         yes;

    patches     ( cylinder );
    rhoName     rhoInf;      // Indicates incompressible
    rhoInf      1;           // Redundant for incompressible
    liftDir     (0 0 1);
    dragDir     (1 0 0);
    CofR        (0.72 0 0);  // Axle midpoint on ground
    pitchAxis   (0 1 0);
    magUInf     6;
    lRef        0.2;        // Wheelbase length
    Aref        0.028;        // Estimated
/*
    binData
    {
        nBin        20;          // output data into 20 bins
        direction   (1 0 0);     // bin direction
        cumulative  yes;
    }
*/
}


// ************************************************************************* //
```

A.2 Savonius 2D

A.2.1 Summary

In order to run several cases only changing some parameters has been created a folder called *org* that contains the *hostfile*, the bash scripts to set and solve the case. The structure of this foldes is as the follows.

- **sets**
  - **0.org**
    - p
    - U
  - **0RAS**
    - nut
    - $nuTilda$
  - **constant**
    - **polyMesh**
      - $blockMeshDict$
    - **triSurface**
      - $cylinder.stl$
  - **cLAM**
    - $transportProperties$
    - $RASproperties$
  - **cRAS**
    - $transportProperties$
    - $RASproperties$
  - **sPOT**
    - $controlDict$
    - $fvSolution$
  - **sLAM**
    - $controlDict$
    - $fvSolution$
  - **sRAS**
    - $controlDict$
    - $fvSolution$
  - **system**
    - $decomposeParDict$
    - $snappyHexMeshDict$
    - $surfaceFeatureExtractDict$
    - $meshQualityDict$
    - $forceCoeffs$

Bash scripts

The *Execute* bash file create the mesh for each angular position.

```
#!/bin/bash
savonius=sets/constant/triSurface/savonius.stl
#rm *degrees -r
for i in  {0..5};
do
        a=$(($i*30))
        f=$a'degrees'
        cp -r org $f
        cd $f
        surfaceTransformPoints -rollPitchYaw '(0 '$a' 0)' $savonius $savonius
```

```
        ./RUN
        cd ..
done
for j in LAM RAS;
do
        for i in {0..5};
        do
                a=$(($i*30))
                f=$a'degrees'
                cd $f
                ./$j
                cd ..
        done
done
```

Then the cases are solved with the script *Solve*.

```
#!/bin/bash
for j in Lam Ras;
do
        for i in {0..5};
        do
                a=$(($i*30))
                f=$a'degrees'
                cd $f
                ./$j
                cd ..
        done
done
```

The script **Lam** called by the the script are.

```
#!/bin/bash
#Laminar
cp sets/cLAM/RASProperties sets/cLAM/transportProperties LAM/constant
cd LAM
decomposePar >log.decomposePar
mpirun -np 6 simpleFoam -parallel > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
```

And *Ras*

```
#!/bin/bash
#RAS
cp sets/cRAS/RASProperties sets/cRAS/transportProperties RAS/constant
cp sets/0RAS/nut sets/0RAS/nuTilda RAS/0
cd RAS
decomposePar >log.decomposePar
mpirun -np 6 simpleFoam -parallel > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
yPlusRAS > yPlus
paraFoam -builtin -touch
cd ..
```

The script *RUN* called by *Execute* prepare each case.

```
#!/bin/bash
#LAM
cd LAM
decomposePar >log.decomposePar
mpirun -np 4 simpleFoam -parallel > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
#RAS
cd RAS
decomposePar >log.decomposePar
mpirun -np 4 simpleFoam -parallel > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
yPlusRAS > yPlus
paraFoam -builtin -touch
cd ..
```

In order to run the cases with rotation the script *Execute*, in another folder uses the same mesh than the fix case.

```
#!/bin/bash
for i in {0..5};
do
        a=$(($i*30))
        f=$a'degrees'
        cp -r org $f
        cd $f
        ./RUN
        cp -r ../../FIX/$f/POT/constant POT
        ./SOLVE
        cd ..
done
```

Where *RUN* is:

```
#!/bin/bash
rm -R POT LAM RAS
mkdir POT LAM RAS
#Potential
cp sets/0.org sets/system POT -R
cp sets/sPOT/controlDict sets/sPOT/fvSchemes sets/sPOT/fvSolution POT/system
```

and *SOLVE*

```
#!/bin/bash
#FOLDER SETTING
#Potential
cd POT
cp 0.org 0 -r
potentialFoam > log.potentialFoam
paraFoam -builtin -touch
cd ..
cp POT/constant POT/0 LAM -R
cp POT/constant POT/0 RAS -R

#Laminar
cp sets/system LAM -R
cp sets/sLAM/controlDict sets/sLAM/fvSchemes sets/sLAM/fvSolution LAM/system
cp sets/cLAM/RASProperties sets/cLAM/transportProperties LAM/constant
#RAS
cp sets/system RAS -R
cp sets/sRAS/controlDict sets/sRAS/fvSchemes sets/sRAS/fvSolution RAS/system
cp sets/cRAS/RASProperties sets/cRAS/transportProperties RAS/constant
cp sets/0RAS/nut sets/0RAS/nuTilda RAS/0

#RUNNING
#Laminar
cp sets/cLAM/RASProperties sets/cLAM/transportProperties LAM/constant
cd LAM
decomposePar >log.decomposePar
foamJob -p -s simpleFoam> log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
#RAS
cp sets/cRAS/RASProperties sets/cRAS/transportProperties RAS/constant
cp sets/0RAS/nut sets/0RAS/nuTilda RAS/0
cd RAS
decomposePar >log.decomposePar
foamJob -p -s simpleFoam> log.simpleFoam
reconstructPar > log.reconstructPar
```

```
rm -R processor*
yPlusRAS > yPlus
paraFoam -builtin -touch
cd ..
```

A.2.2 0

The boundary conditions defined are U:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (6 0 0);

boundaryField
{

    inlet
        {
                type                        freestream;
                freestreamValue        uniform (6 0 0);
        }
    outlet
    {
        type                        freestream;
                freestreamValue        uniform (6 0 0);
    }

         up
        {
                type                        slip;
```

```
        }

        down
        {
                type                        slip;
        }

        front
        {
                type                        empty;
        }

        back
        {
                type                        empty;
        }

   savonius
    {

                type                        rotatingWallVelocity;
                origin                        ( 0 0 0 );
                axis                        ( 0 1 0 );
                omega                       20;

    }
}
```

```
// ************************************************************************* //
```

The pressure is defined in p.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
```

```
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{

    inlet
    {
        type            freestreamPressure;
    }

    outlet
    {
        type            freestreamPressure;
    }
        up
        {
                type                        slip;
        }
        down
        {
                type                        slip;
        }

        back
        {
                type                        empty;
        }
           front
        {
                type                        empty;
        }

    savonius
    {
        type            zeroGradient;
    }


}

// ************************************************************************* //
```

A.2.3 constant

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1;

vertices
(
    (-2 0 -1.5)
    (4 0 -1.5)
    (4 0.005 -1.5)
    (-2 0.005 -1.5)
    (-2 0 1.5)
    (4 0 1.5)
    (4 0.005 1.5)
    (-2 0.005 1.5)
);


blocks
(
    hex (0 1 2 3 4 5 6 7) (200 1 100) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    front
    {
        type empty;
```

```
        faces
        (
            (1 5 4 0)
        );
    }
        back
        {
        type empty;
        faces
        (
            (3 7 6 2)
        );
    }

    inlet
    {
        type patch;
        faces
        (
            (0 4 7 3)
        );
    }
    outlet
    {
        type patch;
        faces
        (
            (2 6 5 1)
        );
    }
           down
    {
        type patch;
        faces
        (
            (0 3 2 1)
        );
    }
    up
    {
        type patch;
        faces
        (
            (4 5 6 7)
        );
    }
);
```

```
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            laminar;

turbulence          off;

printCoeffs         on;
```

```
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            SpalartAllmaras;

turbulence          on;

printCoeffs         on;


// ************************************************************************* //
```

A.2.4 system

Potential

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs
(
    "libOpenFOAM.so"
    "libincompressibleTurbulenceModel.so"
    "libincompressibleRASModels.so"
);

application     simpleFoam;

startTime       0;

stopAt          endTime;

endTime         0.1;

deltaT          1e-4;

writeControl    adjustableRunTime;

writeInterval   0.01;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;
```

```
timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;


//- Uncomment to have regular (every 2 hours of run time) restart files
//secondaryWriteControl    cpuTime; // runtime
//secondaryWriteInterval   7200;    // seconds
//secondaryPurgeWrite      1;       // keep all but last dump


writeFormat     binary;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
```

```
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         leastSquares;
}

divSchemes
{
    default         none;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}
```

```
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }
}

potentialFlow
{
    nNonOrthogonalCorrectors 3;
        pRefPoint        (0.2546565 0.000054651354 -0.15466496);
        pRefValue        0;
}


// ************************************************************************* //
```

Laminar

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
```

```
|    \\/     M anipulation  |                                                       |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs
(
    "libOpenFOAM.so"
    "libincompressibleTurbulenceModel.so"
    "libincompressibleRASModels.so"
);

application     simpleFoam;

startTime       0;

stopAt          endTime;

endTime         0.1;

deltaT          1e-4;

writeControl    adjustableRunTime;

writeInterval   0.01;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;
```

```
maxCo           1;


//- Uncomment to have regular (every 2 hours of run time) restart files
//secondaryWriteControl    cpuTime; // runtime
//secondaryWriteInterval   7200;    // seconds
//secondaryPurgeWrite      1;       // keep all but last dump


writeFormat     binary;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSchemes;
}
```

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         Gauss linear;
    grad(U)         cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss linearUpwindV grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,omega)  bounded Gauss upwind;
           div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p;
}

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
```

```
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver           GAMG;
        tolerance        1e-7;
        relTol           0.01;
        smoother         GaussSeidel;
        nPreSweeps       0;
        nPostSweeps      2;
        cacheAgglomeration on;
        agglomerator     faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels      1;
    }

    U
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
        relTol           0.1;
        nSweeps          1;
    }

    k
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
        relTol           0.1;
        nSweeps          1;
    }
```

```
    omega
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
        relTol           0.1;
        nSweeps          1;
    }


}
SIMPLE
{
    nNonOrthogonalCorrectors 0;
        pRefPoint        (0.215648 0.0000364514 -0.2651545);
        pRefValue        0;

}

potentialFlow
{
    nNonOrthogonalCorrectors 10;
}

relaxationFactors
{
    fields
    {
        p               0.3;
    }
    equations
    {
        U               0.7;
        k               0.7;
               omega           0.7;
    }
}

cache
{
    grad(U);
}

// ************************************************************************* //
```

Laminar

```
/*---------------------------------*- C++ -*----------------------------------*\
```

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs
(
    "libOpenFOAM.so"
    "libincompressibleTurbulenceModel.so"
    "libincompressibleRASModels.so"
);

application     simpleFoam;

startTime       0;

stopAt          endTime;

endTime         0.1;

deltaT          1e-4;

writeControl    adjustableRunTime;

writeInterval   0.01;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;
```

```
runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;


//- Uncomment to have regular (every 2 hours of run time) restart files
//secondaryWriteControl    cpuTime; // runtime
//secondaryWriteInterval   7200;    // seconds
//secondaryPurgeWrite      1;       // keep all but last dump


writeFormat     binary;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
```

```
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         Gauss linear;
    grad(U)         cellLimited Gauss linear 1;
    grad(nuTilda)   cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss linearUpwindV grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,omega)  bounded Gauss upwind;
    div((nuEff*dev(T(grad(U))))) Gauss linear;
    div(phi,nuTilda) bounded Gauss upwind;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
```

```
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-07;
        relTol          0.1;
        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;
        cacheAgglomeration on;
        agglomerator    faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels     1;
    }

    pFinal
    {
        $p;
        tolerance       1e-6;
        relTol          0;
    };

    "(U|nuTilda)"
    {
```

```
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-08;
        relTol          0.1;
    }
}

SIMPLE
{
    nNonOrthogonalCorrectors 0;
        pRefPoint        (0.2546565 0.000054651354 -0.15466496);
        pRefValue        0;
}

potentialFlow
{
    nNonOrthogonalCorrectors 10;
}

relaxationFactors
{
    p               0.3;
    U               0.5;
    nuTilda         0.5;
}

cache
{
    grad(U);
}


// ************************************************************************* //
```

Both

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
```

```
    object      decomposeParDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

numberOfSubdomains 4;

method          hierarchical;
// method          ptscotch;

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

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

manualCoeffs
{
    dataFile        "cellDecomposition";
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      snappyHexMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Which of the steps to run
```

```
castellatedMesh true;
snap            true;
addLayers       true;


// Geometry. Definition of all surfaces. All surfaces are of class
// searchableSurface.
// Surfaces are used
// - to specify refinement for any mesh cell intersecting it
// - to specify refinement for any mesh cell inside/outside/near
// - to 'snap' the mesh boundary to the surface
geometry
{
    savonius.stl
    {
        type triSurfaceMesh;
        name savonius;
    }
    refinementBox
    {
        type searchableBox;
        min (-0.5 0 -0.5);
        max ( 4  0.005 0.5);
    }
};



// Settings for the castellatedMesh generation.
castellatedMeshControls
{

    // Refinement parameters
    // ~~~~~~~~~~~~~~~~~~~~~

    // If local number of cells is >= maxLocalCells on any processor
    // switches from from refinement followed by balancing
    // (current method) to (weighted) balancing before refinement.
    maxLocalCells 100000000;

    // Overall cell limit (approximately). Refinement will stop immediately
    // upon reaching this number so a refinement level might not complete.
    // Note that this is the number of cells before removing the part which
    // is not 'visible' from the keepPoint. The final number of cells might
    // actually be a lot less.
    maxGlobalCells 2000000000;
```

```
// The surface refinement loop might spend lots of iterations refining just a
// few cells. This setting will cause refinement to stop if <= minimumRefine
// are selected for refinement. Note: it will at least do one iteration
// (unless the number of cells to refine is 0)
minRefinementCells 10;

// Allow a certain level of imbalance during refining
// (since balancing is quite expensive)
// Expressed as fraction of perfect balance (= overall number of cells /
// nProcs). 0=balance always.
maxLoadUnbalance 0.10;


// Number of buffer layers between different levels.
// 1 means normal 2:1 refinement restriction, larger means slower
// refinement.
nCellsBetweenLevels 2;



// Explicit feature edge refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies a level for any cell intersected by its edges.
// This is a featureEdgeMesh, read from constant/triSurface for now.
features
(
    {
      file "savonius.eMesh";
      level 6;
   }
);



// Surface based refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies two levels for every surface. The first is the minimum level,
// every cell intersecting a surface gets refined up to the minimum level.
// The second level is the maximum level. Cells that 'see' multiple
// intersections where the intersections make an
// angle > resolveFeatureAngle get refined up to the maximum level.

refinementSurfaces
{
    savonius
```

```
    {
        // Surface-wise min and max refinement level
        level (1 5);

        // Optional specification of patch type (default is wall). No
        // constraint types (cyclic, symmetry) etc. are allowed.
       // patchInfo
        //{
          //  type wall;
            //inGroups (savoniusGroup);
        //}
    }
}

// Resolve sharp angles
resolveFeatureAngle 30;


// Region-wise refinement
// ~~~~~~~~~~~~~~~~~~~~~~

// Specifies refinement level for cells in relation to a surface. One of
// three modes
// - distance. 'levels' specifies per distance to the surface the
//   wanted refinement level. The distances need to be specified in
//   descending order.
// - inside. 'levels' is only one entry and only the level is used. All
//   cells inside the surface get refined up to the level. The surface
//   needs to be closed for this to be possible.
// - outside. Same but cells outside.

refinementRegions
{
   refinementBox
    {
        mode inside;
        levels ((3 3));
    }
}


// Mesh selection
// ~~~~~~~~~~~~~~

// After refinement patches get added for all refinementSurfaces and
// all cells intersecting the surfaces get put into these patches. The
// section reachable from the locationInMesh is kept.
```

```
    // NOTE: This point should never be on a face, always inside a cell, even
    // after refinement.
    locationInMesh (1.123573824 0.00100005468436 -1.12345698754);


    // Whether any faceZones (as specified in the refinementSurfaces)
    // are only on the boundary of corresponding cellZones or also allow
    // free-standing zone faces. Not used if there are no faceZones.
    allowFreeStandingZoneFaces true;
}



// Settings for the snapping.
snapControls
{
    //- Number of patch smoothing iterations before finding correspondence
    //  to surface
    nSmoothPatch 3;

    //- Relative distance for points to be attracted by surface feature point
    //  or edge. True distance is this factor times local
    //  maximum edge length.
    tolerance 2.0;

    //- Number of mesh displacement relaxation iterations.
    nSolveIter 30;

    //- Maximum number of snapping relaxation iterations. Should stop
    //  before upon reaching a correct mesh.
    nRelaxIter 5;

    // Feature snapping

        //- Number of feature edge snapping iterations.
        //  Leave out altogether to disable.
        nFeatureSnapIter 10;

        //- Detect (geometric only) features by sampling the surface
        //  (default=false).
        implicitFeatureSnap true;

        //- Use castellatedMeshControls::features (default = true)
        explicitFeatureSnap false;

        //- Detect points on multiple surfaces (only for explicitFeatureSnap)
        multiRegionFeatureSnap false;
```

```
}


// Settings for the layer addition.
addLayersControls
{
    // Are the thickness parameters below relative to the undistorted
    // size of the refined cell outside layer (true) or absolute sizes (false).
    relativeSizes true;

    // Per final patch (so not geometry!) the layer information
    layers
    {
        savonius
        {
            nSurfaceLayers 15;
        }
    }

    // Expansion factor for layer mesh
    expansionRatio 1.1;

    // Wanted thickness of final added cell layer. If multiple layers
    // is the
    // thickness of the layer furthest away from the wall.
    // Relative to undistorted size of cell outside layer.
    // is the thickness of the layer furthest away from the wall.
    // See relativeSizes parameter.
    finalLayerThickness 0.2;

    // Minimum thickness of cell layer. If for any reason layer
    // cannot be above minThickness do not add layer.
    // Relative to undistorted size of cell outside layer.
    minThickness 0.01;

    // If points get not extruded do nGrow layers of connected faces that are
    // also not grown. This helps convergence of the layer addition process
    // close to features.
    // Note: changed(corrected) w.r.t 17x! (didn't do anything in 17x)
    nGrow 0;

    // Advanced settings

    // When not to extrude surface. 0 is flat surface, 90 is when two faces
    // are perpendicular
    featureAngle 90;
```

```
    // At non-patched sides allow mesh to slip if extrusion direction makes
    // angle larger than slipFeatureAngle.
    slipFeatureAngle 20;

    // Maximum number of snapping relaxation iterations. Should stop
    // before upon reaching a correct mesh.
    nRelaxIter 3;

    // Number of smoothing iterations of surface normals
    nSmoothSurfaceNormals 1;

    // Number of smoothing iterations of interior mesh movement direction
    nSmoothNormals 3;

    // Smooth layer thickness over surface patches
    nSmoothThickness 10;

    // Stop layer growth on highly warped cells
    maxFaceThicknessRatio 0.7;

    // Reduce layer growth where ratio thickness to medial
    // distance is large
    maxThicknessToMedialRatio 0.3;

    // Angle used to pick up medial axis points
    // Note: changed(corrected) w.r.t 17x! 90 degrees corresponds to 130 in 17x.
    minMedianAxisAngle 90;


    // Create buffer region for new layer terminations
    nBufferCellsNoExtrude 0;


    // Overall max number of layer addition iterations. The mesher will exit
    // if it reaches this number of iterations; possibly with an illegal
    // mesh.
    nLayerIter 50;
}



// Generic mesh quality settings. At any undoable phase these determine
// where to undo.
meshQualityControls
{
    #include "meshQualityDict"
```

```
    // Advanced

    //- Number of error distribution iterations
    nSmoothScale 4;
    //- amount to scale back displacement at error points
    errorReduction 0.75;
}


// Advanced

// Write flags
writeFlags
(
    scalarLevels
    layerSets
    layerFields     // write volScalarField for layer coverage
);


// Merge tolerance. Is fraction of overall bounding box of initial mesh.
// Note: the write tolerance needs to be higher than this.
mergeTolerance 1e-6;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      surfaceFeatureExtractDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

savonius.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   170;
    }

    subsetFeatures
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;

        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      meshQualityDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Include defaults parameters from master dictionary
#include "$WM_PROJECT_DIR/etc/caseDicts/meshQualityDict"
```

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


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/

forceCoeffs1
{
    type        forceCoeffs;

    functionObjectLibs ( "libforces.so" );

    outputControl   timeStep;
    timeInterval    ;

    log         yes;

    patches     ( savonius);
    rhoName     rhoInf;                     // Indicates incompressible
    rhoInf      1;                          // Redundant for incompressible
    liftDir     (0 0 1);
    dragDir     (1 0 0);
    CofR        (0 0 0);                        // Axle midpoint on ground
    pitchAxis   (0 1 0);
    magUInf     6;
    lRef        0.09;                       // Radius
    Aref        4.915229e-3;        // Estimated
/*
    binData
    {
        nBin        20;          // output data into 20 bins
        direction   (1 0 0);     // bin direction
        cumulative  yes;
    }
*/
}


// ************************************************************************* //
```

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      extrudeMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

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

//constructFrom mesh;
constructFrom patch;
//constructFrom surface;

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

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

//- 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             1;

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       0.01;
}

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;


// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

A.3 Savonius 3D

A.3.1 Summary

In order to run several cases only changing some parameters has been created a folder called *org* that contains the *hostfile*, the bash scripts to set and solve the case. The structure of this folders is as the follows.

- **sets**
  - **0.org**
    - p
    - U
  - **0RAS**
    - nut
    - $nuTilda$
  - **0LES**
    - k
    - $nuSgs$
  - **constant**
    - **polyMesh**
      - $blockMeshDict$
    - **triSurface**
      - $savonius.stl$
  - **cLAM**
    - $transportProperties$
    - $RASproperties$
  - **cRAS**
    - $transportProperties$
    - $RASproperties$
  - **cLES**
    - $transportProperties$
    - $RASproperties$
  - **sPOT**
    - $controlDict$
    - $fvSolution$
  - **sLAM**
    - $controlDict$
    - $fvSolution$
  - **sRAS**
    - $controlDict$
    - $fvSolution$
  - **sLES**
    - $controlDict$
    - $fvSolution$
  - **system**
    - $decomposeParDict$
    - $snappyHexMeshDict$
    - $surfaceFeatureExtractDict$
    - $meshQualityDict$
    - $forceCoeffs$

Bash scripts

The *Execute* bash generate the mesh of each angular position:

```
#!/bin/bash
savonius=sets/constant/triSurface/savonius.stl
rm *degrees -r
```

```
for i in  {0..5};
do
        a=$(($i*30))
        f=$a'degrees'
        cp -r org $f
        cd $f
        surfaceTransformPoints -rollPitchYaw '(0 '$a' 0)' $savonius $savonius
        ./RUN
        cd ..
done
```

Then the script *LAM* run the laminar cases:

```
#!/bin/bash
for i in {0..5};
        do
        a=$(($i*30))
        f=$a'degrees'
        cd $f
        ./Lam
        cd ..
done
```

And *RAS* run the turbulent cases:

```
#!/bin/bash
for i in {0..5};
        do
        a=$(($i*30))
        f=$a'degrees'
        cd $f
        ./Ras
        cd ..
done
```

Also *LES* should run the LES turbulent cases:

```
#!/bin/bash
for i in {0..5};
        do
        a=$(($i*30))
        f=$a'degrees'
        cd $f
        ./Les
        cd ..
done
```

In these there are *Lam*.

```
#!/bin/bash
```

```
cd LAM
decomposePar >log.decomposePar
foamJob -s -p simpleFoam > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor
paraFoam -builtin -touch
Co> log.courant
cd ..
```

Ras :

```
#!/bin/bash
cd RAS
decomposePar >log.decomposePar
foamJob -s -p simpleFoam > log.simpleFoam
reconstructPar > log.reconstructPar
rm -R processor*
yPlusRAS > log.yPlus
paraFoam -builtin -touch
Co> log.courant
cd ..
```

Les:

```
#!/bin/bash
cd LES
decomposePar >log.decomposePar
foamJob -s -p pisoFoam > log.pisoFoam
reconstructPar > log.reconstructPar
rm -R processor*
yPlusLES > log.yPlus
paraFoam -builtin -touch
Co> log.courant
cd ..
```

A.3.2 0

The boundary conditions defined are U:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
```

```
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (6 0 0);

boundaryField
{

    inlet
        {
                type                        freestream;
                freestreamValue        uniform (6 0 0);
        }
    outlet
    {
        type                          freestream;
                freestreamValue        uniform (6 0 0);
    }

         up
        {
                type                        slip;
        }

        down
        {
                type                        slip;
        }

        front
        {
                type                        empty;
        }

        back
        {
                type                        empty;
        }

   savonius
    {
```

```
                type                            rotatingWallVelocity;
                origin                            ( 0 0 0 );
                axis                            ( 0 1 0 );
                omega                           20;

    }
}



// ************************************************************************* //
```

The pressure is defined in p.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{

    inlet
    {
        type            freestreamPressure;
    }

    outlet
    {
        type            freestreamPressure;
```

```
    }
        up
        {
                type                        slip;
        }
        down
        {
                type                        slip;
        }

        back
        {
                type                        empty;
        }
           front
        {
                type                        empty;
        }

    savonius
    {
        type            zeroGradient;
    }


}

// ************************************************************************* //
```

A.3.3 constant

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

```
convertToMeters 1;

vertices
(
    (-2 -1.5 -1.5)
    (4 -1.5 -1.5)
    (4 1.5 -1.5)
    (-2 1.5 -1.5)
    (-2 -1.5 1.5)
    (4 -1.5 1.5)
    (4 1.5 1.5)
    (-2 1.5 1.5)
);

blocks
(
    hex (0 1 2 3 4 5 6 7) (150 75 75) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    frontAndBack
    {
        type patch;
        faces
        (
            (3 7 6 2)
            (1 5 4 0)
        );
    }
    inlet
    {
        type patch;
        faces
        (
            (0 4 7 3)
        );
    }
    outlet
    {
        type patch;
        faces
        (
```

```
            (2 6 5 1)
        );
    }
    lowerWall
    {
        type patch;
        faces
        (
            (0 3 2 1)
        );
    }
    upperWall
    {
        type patch;
        faces
        (
            (4 5 6 7)
        );
    }
);

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
```

```
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            laminar;

turbulence          off;

printCoeffs         on;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
```

```
|   \\  /    A nd           | Web:      www.OpenFOAM.org                    |
|    \\/     M anipulation  |                                               |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel        SpalartAllmaras;

turbulence      on;

printCoeffs     on;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
```

```
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      LESProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

LESModel            SpalartAllmarasDDES;

delta               cubeRootVol;

printCoeffs         on;

cubeRootVolCoeffs
{
    deltaCoeff      1;
}

PrandtlCoeffs
{
    delta           cubeRootVol;
    cubeRootVolCoeffs
    {
        deltaCoeff      1;
    }
    smoothCoeffs
    {
        delta           cubeRootVol;
        cubeRootVolCoeffs
        {
            deltaCoeff      1;
        }
        maxDeltaRatio   1.1;
    }
    Cdelta           0.158;
}

vanDriestCoeffs
{
    delta           cubeRootVol;
    cubeRootVolCoeffs
```

```
    {
        deltaCoeff      1;
    }
    smoothCoeffs
    {
        delta           cubeRootVol;
        cubeRootVolCoeffs
        {
            deltaCoeff      1;
        }
        maxDeltaRatio   1.1;
    }
    Aplus            26;
    Cdelta           0.158;
}

smoothCoeffs
{
    delta           cubeRootVol;
    cubeRootVolCoeffs
    {
        deltaCoeff      1;
    }
    maxDeltaRatio   1.1;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      turbulenceProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

simulationType  LESModel;
```

```
// ************************************************************************* //
```

A.3.4 system

Potential

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     potentialFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         1;

deltaT          1;

writeControl    timeStep;

writeInterval   1;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;
```

```
timePrecision   6;

runTimeModifiable true;

functions
{
    difference
    {
        // Load the library containing the 'coded' functionObject
        functionObjectLibs ("libutilityFunctionObjects.so");
        type coded;
        // Name of on-the-fly generated functionObject
        redirectType error;
        code
        #{
            // Lookup U
            Info<< "Looking up field U\n" << endl;
            const volVectorField& U = mesh().lookupObject<volVectorField>("U");

            Info<< "Reading inlet velocity  uInfX\n" << endl;

            scalar ULeft = 0.0;
            label leftI = mesh().boundaryMesh().findPatchID("inlet");
            const fvPatchVectorField& fvp = U.boundaryField()[leftI];
            if (fvp.size())
            {
                ULeft = fvp[0].x();
            }
            reduce(ULeft, maxOp<scalar>());

            dimensionedScalar uInfX
            (
                "uInfx",
                dimensionSet(0, 1, -1, 0, 0),
                ULeft
            );

            Info << "U at inlet = " << uInfX.value() << " m/s" << endl;


            scalar magCylinder = 0.0;
            label cylI = mesh().boundaryMesh().findPatchID("savonius");
            const fvPatchVectorField& cylFvp = mesh().C().boundaryField()[cylI];
            if (cylFvp.size())
            {
                magCylinder = mag(cylFvp[0]);
```

```
}
reduce(magCylinder, maxOp<scalar>());

dimensionedScalar radius
(
    "radius",
    dimensionSet(0, 1, 0, 0, 0),
    magCylinder
);

Info << "Cylinder radius = " << radius.value() << " m" << endl;

volVectorField UA
(
    IOobject
    (
        "UA",
        mesh().time().timeName(),
        U.mesh(),
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    U
);

Info<< "\nEvaluating analytical solution" << endl;

const volVectorField& centres = UA.mesh().C();
volScalarField magCentres(mag(centres));
volScalarField theta(acos((centres & vector(1,0,0))/magCentres));

volVectorField cs2theta
(
    cos(2*theta)*vector(1,0,0)
  + sin(2*theta)*vector(0,1,0)
);

UA = uInfX*(dimensionedVector(vector(1,0,0))
  - pow((radius/magCentres),2)*cs2theta);

// Force writing of UA (since time has not changed)
UA.write();

volScalarField error("error", mag(U-UA)/mag(UA));

Info<<"Writing relative error in U to " << error.objectPath()
    << endl;
```

```
            error.write();
        #};
    }
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         leastSquares;
}

divSchemes
{
    default         none;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
```

```
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }
}

potentialFlow
{
    nNonOrthogonalCorrectors 3;
```

```
        pRefPoint        (0.2546565 0.54651354 -0.15466496);
        pRefValue        0;
}


// ************************************************************************* //
```

Laminar

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs
(
    "libOpenFOAM.so"
    "libincompressibleTurbulenceModel.so"
    "libincompressibleRASModels.so"
);

application     simpleFoam;

startTime              latestTime;

stopAt          endTime;

endTime         2.5e-3;

deltaT          5e-6;

writeControl    timeStep;

writeInterval   500;

purgeWrite      0;
```

```
writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;


//- Uncomment to have regular (every 2 hours of run time) restart files
//secondaryWriteControl    cpuTime; // runtime
//secondaryWriteInterval   7200;    // seconds
//secondaryPurgeWrite      1;       // keep all but last dump


writeFormat     binary;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*---------------------------------*- C++ -*----------------------------------*\
```

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         Gauss linear;
    grad(U)         cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss linearUpwindV grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,omega)  bounded Gauss upwind;
           div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
```

```
    default         corrected;
}

fluxRequired
{
    default         no;
    p;
}

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver           GAMG;
        tolerance        1e-7;
        relTol           0.01;
        smoother         GaussSeidel;
        nPreSweeps       0;
        nPostSweeps      2;
        cacheAgglomeration on;
        agglomerator     faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels      1;
    }

    U
    {
        solver           smoothSolver;
        smoother         GaussSeidel;
        tolerance        1e-8;
```

```
        relTol          0.1;
        nSweeps         1;
    }

    k
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-8;
        relTol          0.1;
        nSweeps         1;
    }

    omega
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-8;
        relTol          0.1;
        nSweeps         1;
    }


}
SIMPLE
{
residualControl
 {
        p               1e-4;
        U               1e-5;

 }
        correctPhi          yes;
    nNonOrthogonalCorrectors 5;
        pRefPoint        (0.215648 0.364514 -0.2651545);
        pRefValue        0;

}

potentialFlow
{
    nNonOrthogonalCorrectors 10;
}

relaxationFactors
{
    fields
```

```
    {
        p               0.3;
    }
    equations
    {
        U               0.7;
        k               0.7;
                omega           0.7;
    }
}

cache
{
    grad(U);
}

// ************************************************************************* //
```

RAS

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs
(
    "libOpenFOAM.so"
    "libincompressibleTurbulenceModel.so"
    "libincompressibleRASModels.so"
);

application     simpleFoam;

startTime               latestTime;

stopAt          endTime;
```

```
endTime         2.5e-3;

deltaT          5e-6;

writeControl    timeStep;

writeInterval   500;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;


//- Uncomment to have regular (every 2 hours of run time) restart files
//secondaryWriteControl    cpuTime; // runtime
//secondaryWriteInterval   7200;    // seconds
//secondaryPurgeWrite      1;       // keep all but last dump


writeFormat     binary;

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
```

```
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         Gauss linear;
    grad(U)         cellLimited Gauss linear 1;
    grad(nuTilda)   cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      bounded Gauss linearUpwindV grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,omega)  bounded Gauss upwind;
    div((nuEff*dev(T(grad(U))))) Gauss linear;
    div(phi,nuTilda) bounded Gauss upwind;
```

```
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-7;
```

```
        relTol          0.01;
        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;
        cacheAgglomeration on;
        agglomerator    faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels     1;
    }

    U
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-8;
        relTol          0.1;
        nSweeps         1;
    }
    nuTilda
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        nSweeps         2;
        tolerance       1e-08;
        relTol          0.1;
    }



}
SIMPLE
{

    nNonOrthogonalCorrectors 3;
        pRefPoint       (0.215648 0.364514 -0.2651545);
        pRefValue       0;
    residualControl
    {
        p               1e-5;
        U               1e-5;
        nuTilda         1e-5;
    }

}

potentialFlow
{
```

```
    nNonOrthogonalCorrectors 10;
}

relaxationFactors
{
    fields
    {
        p               0.3;
    }
    equations
    {
        U               0.7;
        k               0.7;
                omega           0.7;
    }
}

cache
{
    grad(U);
}

// ************************************************************************* //
```

LES

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

libs            ("libOpenFOAM.so" "libfieldFunctionObjects.so");

application     pisoFoam;

startFrom       startTime;
```

```
startTime       0;

stopAt          endTime;

endTime         0.01;

deltaT          1e-5;

writeControl    timeStep;

writeInterval   100;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression compressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;


functions
{
    #include "readFields"
    #include "cuttingPlane"
    #include "streamLines"
    #include "forceCoeffs"
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
```

```
    format      ascii;
    class       dictionary;
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default backward;
}

d2dt2Schemes
{
}

gradSchemes
{
    default         Gauss linear;

    grad(nuTilda)   cellLimited Gauss linear 1;
    grad(U)         cellLimited Gauss linear 1;
}

divSchemes
{
    default         none;

    div(phi,U)      Gauss LUST unlimitedGrad(U);
    //div(phi,U)      Gauss linearUpwind unlimitedGrad(U);
    div(phi,k)      Gauss limitedLinear 1;
    div(phi,nuTilda) Gauss limitedLinear 1;

    div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear limited corrected 0.33;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
```

```
    default         limited corrected 0.33;
}

fluxRequired
{
    default         no;
    p;
}

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          GAMG;
        tolerance       1e-6;
        relTol          0.1;

        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;

        cacheAgglomeration true;

        nCellsInCoarsestLevel 50;//10;
        agglomerator    faceAreaPair;
        mergeLevels     1;
    };

    pFinal
    {
```

```
        $p;
        tolerance       1e-6;
        relTol          0;
    };

    "(U|k|B|nuTilda)"
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-07;
        relTol          0;
    };
}

PISO
{
    nCorrectors     2;
    nNonOrthogonalCorrectors 1;
         pRefPoint        (0.2546565 0.54651354 -0.15466496);
        pRefValue        0;
}

relaxationFactors
{
    "U.*"               1;
    "nuTilda.*"         1;
}


// ************************************************************************* //
```

Both

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      decomposeParDict;
}
```

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


numberOfSubdomains 16;

method          scotch;

distributed     no;

roots           ( );


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      snappyHexMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Which of the steps to run
castellatedMesh true;
snap            true;
addLayers       true;


// Geometry. Definition of all surfaces. All surfaces are of class
// searchableSurface.
// Surfaces are used
// - to specify refinement for any mesh cell intersecting it
// - to specify refinement for any mesh cell inside/outside/near
// - to 'snap' the mesh boundary to the surface
geometry
{
    savonius.stl
    {
        type triSurfaceMesh;
        name savonius;
```

```
    }

    refinementBox1
    {
        type searchableBox;
        min (-0.5 -0.5 -0.5);
        max ( 4  0.5 0.5);
    }
    refinementBox2
    {
        type searchableBox;
        min (-0.2 -0.2 -0.2);
        max ( 0.5  0.2 0.2);
    }
};



// Settings for the castellatedMesh generation.
castellatedMeshControls
{

    // Refinement parameters
    // ~~~~~~~~~~~~~~~~~~~~~

    // If local number of cells is >= maxLocalCells on any processor
    // switches from from refinement followed by balancing
    // (current method) to (weighted) balancing before refinement.
    maxLocalCells 10000000;

    // Overall cell limit (approximately). Refinement will stop immediately
    // upon reaching this number so a refinement level might not complete.
    // Note that this is the number of cells before removing the part which
    // is not 'visible' from the keepPoint. The final number of cells might
    // actually be a lot less.
    maxGlobalCells 12000000;

    // The surface refinement loop might spend lots of iterations refining just a
    // few cells. This setting will cause refinement to stop if <= minimumRefine
    // are selected for refinement. Note: it will at least do one iteration
    // (unless the number of cells to refine is 0)
    minRefinementCells 10;

    // Allow a certain level of imbalance during refining
    // (since balancing is quite expensive)
    // Expressed as fraction of perfect balance (= overall number of cells /
    // nProcs). 0=balance always.
```

```
maxLoadUnbalance 0.10;


// Number of buffer layers between different levels.
// 1 means normal 2:1 refinement restriction, larger means slower
// refinement.
nCellsBetweenLevels 2;



// Explicit feature edge refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies a level for any cell intersected by its edges.
// This is a featureEdgeMesh, read from constant/triSurface for now.
features
(
    {
      file "savonius.eMesh";
      level 6;
   }
);



// Surface based refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies two levels for every surface. The first is the minimum level,
// every cell intersecting a surface gets refined up to the minimum level.
// The second level is the maximum level. Cells that 'see' multiple
// intersections where the intersections make an
// angle > resolveFeatureAngle get refined up to the maximum level.

refinementSurfaces
{
    savonius
    {
        // Surface-wise min and max refinement level
        level (1 5);

        // Optional specification of patch type (default is wall). No
        // constraint types (cyclic, symmetry) etc. are allowed.
        //patchInfo
        //{
        //    type wall;
        //    inGroups (savoniusGroup);
```

```
        //}
    }
}

// Resolve sharp angles
resolveFeatureAngle 30;


// Region-wise refinement
// ~~~~~~~~~~~~~~~~~~~~~~

// Specifies refinement level for cells in relation to a surface. One of
// three modes
// - distance. 'levels' specifies per distance to the surface the
//   wanted refinement level. The distances need to be specified in
//   descending order.
// - inside. 'levels' is only one entry and only the level is used. All
//   cells inside the surface get refined up to the level. The surface
//   needs to be closed for this to be possible.
// - outside. Same but cells outside.

refinementRegions
{
   refinementBox1
    {
        mode inside;
        levels ((2 2));
    }
   refinementBox2
    {
        mode inside;
        levels ((3 3));
    }
}


// Mesh selection
// ~~~~~~~~~~~~~~

// After refinement patches get added for all refinementSurfaces and
// all cells intersecting the surfaces get put into these patches. The
// section reachable from the locationInMesh is kept.
// NOTE: This point should never be on a face, always inside a cell, even
// after refinement.
locationInMesh (1.123573824 1.156786348421 -1.12345698754);
```

```
    // Whether any faceZones (as specified in the refinementSurfaces)
    // are only on the boundary of corresponding cellZones or also allow
    // free-standing zone faces. Not used if there are no faceZones.
    allowFreeStandingZoneFaces true;
}



// Settings for the snapping.
snapControls
{
    //- Number of patch smoothing iterations before finding correspondence
    //  to surface
    nSmoothPatch 3;

    //- Relative distance for points to be attracted by surface feature point
    //  or edge. True distance is this factor times local
    //  maximum edge length.
    tolerance 2.0;

    //- Number of mesh displacement relaxation iterations.
    nSolveIter 30;

    //- Maximum number of snapping relaxation iterations. Should stop
    //  before upon reaching a correct mesh.
    nRelaxIter 5;

    // Feature snapping

        //- Number of feature edge snapping iterations.
        //  Leave out altogether to disable.
        nFeatureSnapIter 10;

        //- Detect (geometric only) features by sampling the surface
        //  (default=false).
        implicitFeatureSnap false;

        //- Use castellatedMeshControls::features (default = true)
        explicitFeatureSnap true;

        //- Detect points on multiple surfaces (only for explicitFeatureSnap)
        multiRegionFeatureSnap false;
}



// Settings for the layer addition.
```

```
addLayersControls
{
    // Are the thickness parameters below relative to the undistorted
    // size of the refined cell outside layer (true) or absolute sizes (false).
    relativeSizes true;

    // Per final patch (so not geometry!) the layer information
    layers
    {
        savonius
        {
            nSurfaceLayers 5;
        }
    }

    // Expansion factor for layer mesh
    expansionRatio 1.1;

    // Wanted thickness of final added cell layer. If multiple layers
    // is the
    // thickness of the layer furthest away from the wall.
    // Relative to undistorted size of cell outside layer.
    // is the thickness of the layer furthest away from the wall.
    // See relativeSizes parameter.
    finalLayerThickness 0.3;

    // Minimum thickness of cell layer. If for any reason layer
    // cannot be above minThickness do not add layer.
    // Relative to undistorted size of cell outside layer.
    minThickness 0.1;

    // If points get not extruded do nGrow layers of connected faces that are
    // also not grown. This helps convergence of the layer addition process
    // close to features.
    // Note: changed(corrected) w.r.t 17x! (didn't do anything in 17x)
    nGrow 0;

    // Advanced settings

    // When not to extrude surface. 0 is flat surface, 90 is when two faces
    // are perpendicular
    featureAngle 90;//60

    // At non-patched sides allow mesh to slip if extrusion direction makes
    // angle larger than slipFeatureAngle.
    slipFeatureAngle 30;//30
```

```
    // Maximum number of snapping relaxation iterations. Should stop
    // before upon reaching a correct mesh.
    nRelaxIter 3;

    // Number of smoothing iterations of surface normals

    nSmoothSurfaceNormals 1;
    // Number of smoothing iterations of interior mesh movement direction
    nSmoothNormals 3;

    // Smooth layer thickness over surface patches
    nSmoothThickness 10;

    // Stop layer growth on highly warped cells
    maxFaceThicknessRatio 0.5;

    // Reduce layer growth where ratio thickness to medial
    // distance is large
    maxThicknessToMedialRatio 0.3;

    // Angle used to pick up medial axis points
    // Note: changed(corrected) w.r.t 17x! 90 degrees corresponds to 130 in 17x.
    minMedianAxisAngle 90;


    // Create buffer region for new layer terminations
    nBufferCellsNoExtrude 0;


    // Overall max number of layer addition iterations. The mesher will exit
    // if it reaches this number of iterations; possibly with an illegal
    // mesh.
    nLayerIter 50;
}



// Generic mesh quality settings. At any undoable phase these determine
// where to undo.
meshQualityControls
{
    #include "meshQualityDict"


    // Advanced

    //- Number of error distribution iterations
```

```
    nSmoothScale 4;
    //- amount to scale back displacement at error points
    errorReduction 0.75;
}


// Advanced

// Write flags
writeFlags
(
    scalarLevels
    layerSets
    layerFields     // write volScalarField for layer coverage
);


// Merge tolerance. Is fraction of overall bounding box of initial mesh.
// Note: the write tolerance needs to be higher than this.
mergeTolerance 1e-6;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      surfaceFeatureExtractDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

savonius.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   180;
    }

    subsetFeatures
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;

        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      meshQualityDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Include defaults parameters from master dictionary
#include "$WM_PROJECT_DIR/etc/caseDicts/meshQualityDict"

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

```
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/

forceCoeffs1
{
    type        forceCoeffs;

    functionObjectLibs ( "libforces.so" );

    outputControl   timeStep;
    timeInterval    ;

    log         yes;

    patches     ( savonius);
    rhoName     rhoInf;      // Indicates incompressible
    rhoInf      1;           // Redundant for incompressible
    liftDir     (0 0 1);
    dragDir     (1 0 0);
    CofR        (0 0 0);  // Axle midpoint on ground
    pitchAxis   (0 1 0);
    magUInf     9;
    lRef        0.09091;        // Wheelbase length
    Aref        0.05091;        // Estimated
/*
    binData
    {
        nBin        20;          // output data into 20 bins
        direction   (1 0 0);     // bin direction
        cumulative  yes;
    }
*/
}


// ************************************************************************* //
```

A.4 AMI 2D

A.4.1 Summary

In order to run several cases only changing some parameters has been created a folder called *org* that contains the *hostfile*, the bash scripts to set and solve the case. The structure of this folders is as the follows.

- **sets**
  - **0.org**
    - p
    - U
  - **0RAS**
    - nut
    - $nuTilda$
  - **constant**
    - **polyMesh**
      - $blockMeshDict$
    - **triSurface**
      - $savonius.stl$
  - **cLAM**
    - $transportProperties$
    - $RASproperties$
  - **cRAS**
    - $transportProperties$
    - $turbulenceProperties$
    - $RASproperties$
  - **sPOT**
    - $controlDict$
    - $fvSolution$
  - **sLAM**
    - $controlDict$
    - $fvSolution$
  - **sRAS**
    - $controlDict$
    - $fvSolution$
  - **system**
    - $decomposeParDict$
    - $snappyHexMeshDict$
    - $surfaceFeatureExtractDict$
    - $meshQualityDict$
    - $forceCoeffs$
    - $createPatchDict$
    - $extruceMeshDict$
    - $topoSetDict$

Bash scripts

The script *RUN* prepare the case:

```
#!/bin/bash
rm -r POT LAM RAS
mkdir POT LAM RAS
#Potential
cp sets/constant sets/0.org sets/system POT -R
```

```
cp sets/sPOT/controlDict sets/sPOT/fvSchemes sets/sPOT/fvSolution POT/system
./potential
cp POT/constant POT/0 LAM -R
cp POT/constant POT/0 RAS -R

#LAMINAR
cp sets/system LAM -R
cp sets/sLAM/controlDict sets/sLAM/fvSchemes sets/sLAM/fvSolution LAM/system
cp sets/cLAM/RASProperties sets/cLAM/transportProperties  sets/cLAM/turbulenceProper

#RAS
cp sets/system RAS -R
cp sets/sRAS/controlDict sets/sRAS/fvSchemes sets/sRAS/fvSolution RAS/system
cp sets/cRAS/RASProperties sets/cRAS/transportProperties RAS/constant
cp sets/0RAS/nut sets/0RAS/nuTilda RAS/0

#./SOLVE
```

potential

```
#!/bin/bash
cd POT
rm 0 -r
cp 0.org 0 -R
blockMesh > log.blockMesh
checkMesh > log.checkMesh
surfaceFeatureExtract > log.surfaceFeatureExtract
snappyHexMesh -overwrite> log.snappyHexMesh
extrudeMesh > log.extrudeMesh
createPatch -overwrite> log.createPatch
topoSet > log.topoSet
checkMesh -allGeometry -allTopology > log.checkMeshAll
rm -R 0
cp 0.org 0 -R
potentialFoam > log.potentialFoam
paraFoam -builtin -touch
cd ..
```

SOLVE

```
#!/bin/bash
#LAMINAR
cd LAM
decomposePar >log.decomposePar
mpirun -np 6 pimpleDyMFoam -parallel > log.pimpleDyMFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
```

```
#RAS TURBULENCE
cd RAS
decomposePar >log.decomposePar
mpirun -np 6 pimpleDyMFoam -parallel > log.pimpleDyMFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
```

A.4.2 0

The boundary conditions defined are U:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (9 0 0);

boundaryField
{

    inlet
        {
                type                          freestream;
                freestreamValue        uniform (9 0 0);
        }
    outlet
    {
        type                        freestream;
```

```
                freestreamValue        uniform (9 0 0);
    }

         up
        {
                type                           slip;
        }

        down
        {
                type                           slip;
        }

        front
        {
                type                           empty;
        }

        back
        {
                type                           empty;
        }

   savonius
    {
        type            movingWallVelocity; // fixedValue;
        value           uniform (0 0 0);

    }
    AMI1
    {
        type            cyclicAMI;
        value           $internalField;
    }

    AMI2
    {
        type            cyclicAMI;
        value           $internalField;
    }
}


// ************************************************************************* //
```

The pressure is defined in p.

```
/*--------------------------------*- C++ -*----------------------------------*\
```

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{

    inlet
    {
        type            freestreamPressure;
    }

    outlet
    {
        type            freestreamPressure;
    }
        up
        {
                type                        slip;
        }
        down
        {
                type                        slip;
        }

        back
        {
                type                        empty;
        }
           front
        {
```

```
                type                            empty;
        }

    savonius
    {
        type            zeroGradient;
    }
    AMI1
    {
        type            cyclicAMI;
        value           $internalField;
    }

    AMI2
    {
        type            cyclicAMI;
        value           $internalField;
    }
}

// ************************************************************************* //
```

A.4.3 constant

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1;

vertices
(
    (-2 0 -1.5)
    (4 0 -1.5)
    (4 0.006 -1.5)
```

```
    (-2 0.006 -1.5)
    (-2 0 1.5)
    (4 0 1.5)
    (4 0.006 1.5)
    (-2 0.006 1.5)
);


blocks
(
    hex (0 1 2 3 4 5 6 7) (200 1 100) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    front
    {
        type empty;
        faces
        (
            (1 5 4 0)
        );
    }
        back
        {
        type empty;
        faces
        (
            (3 7 6 2)
        );
    }

    inlet
    {
        type patch;
        faces
        (
            (0 4 7 3)
        );
    }
    outlet
    {
        type patch;
```

```
        faces
        (
            (2 6 5 1)
        );
    }
           down
    {
        type patch;
        faces
        (
            (0 3 2 1)
        );
    }
    up
    {
        type patch;
        faces
        (
            (4 5 6 7)
        );
    }
);

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      dynamicMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dynamicFvMesh   solidBodyMotionFvMesh;

motionSolverLibs ( "libfvMotionSolvers.so" );

solidBodyMotionFvMeshCoeffs
{
```

```
    cellZone        cylinder;

    solidBodyMotionFunction  rotatingMotion;
    rotatingMotionCoeffs
    {
        origin        (0 0 0);
        axis          (0 1 0);
        omega         79.2; // rad/s
    }
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
```

```
    location    "constant";
    object      turbulenceProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

simulationType  RASModel;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            laminar;

turbulence          off;

printCoeffs         on;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
```

```
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            SpalartAllmaras;

turbulence          on;

printCoeffs         on;


// ************************************************************************* //
```

A.4.4 system

Potential

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     potentialFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         1;

deltaT          1;

writeControl    timeStep;

writeInterval   1;
```

```
purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    difference
    {
        // Load the library containing the 'coded' functionObject
        functionObjectLibs ("libutilityFunctionObjects.so");
        type coded;
        // Name of on-the-fly generated functionObject
        redirectType error;
        code
        #{
            // Lookup U
            Info<< "Looking up field U\n" << endl;
            const volVectorField& U = mesh().lookupObject<volVectorField>("U");

            Info<< "Reading inlet velocity  uInfX\n" << endl;

            scalar ULeft = 0.0;
            label leftI = mesh().boundaryMesh().findPatchID("inlet");
            const fvPatchVectorField& fvp = U.boundaryField()[leftI];
            if (fvp.size())
            {
                ULeft = fvp[0].x();
            }
            reduce(ULeft, maxOp<scalar>());

            dimensionedScalar uInfX
            (
                "uInfx",
                dimensionSet(0, 1, -1, 0, 0),
                ULeft
            );
```

```
Info << "U at inlet = " << uInfX.value() << " m/s" << endl;


scalar magCylinder = 0.0;
label cylI = mesh().boundaryMesh().findPatchID("savonius");
const fvPatchVectorField& cylFvp = mesh().C().boundaryField()[cylI];
if (cylFvp.size())
{
    magCylinder = mag(cylFvp[0]);
}
reduce(magCylinder, maxOp<scalar>());

dimensionedScalar radius
(
    "radius",
    dimensionSet(0, 1, 0, 0, 0),
    magCylinder
);

Info << "Cylinder radius = " << radius.value() << " m" << endl;

volVectorField UA
(
    IOobject
    (
        "UA",
        mesh().time().timeName(),
        U.mesh(),
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    U
);

Info<< "\nEvaluating analytical solution" << endl;

const volVectorField& centres = UA.mesh().C();
volScalarField magCentres(mag(centres));
volScalarField theta(acos((centres & vector(1,0,0))/magCentres));

volVectorField cs2theta
(
    cos(2*theta)*vector(1,0,0)
  + sin(2*theta)*vector(0,1,0)
);
```

```
            UA = uInfX*(dimensionedVector(vector(1,0,0))
              - pow((radius/magCentres),2)*cs2theta);

            // Force writing of UA (since time has not changed)
            UA.write();

            volScalarField error("error", mag(U-UA)/mag(UA));

            Info<<"Writing relative error in U to " << error.objectPath()
                << endl;

            error.write();
        #};
    }
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         leastSquares;
}

divSchemes
{
```

```
    default         none;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
```

```
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }
}

potentialFlow
{
    nNonOrthogonalCorrectors 2;
        pRefPoint        (0.2546565 0.000054651354 -0.15466496);
        pRefValue        0;
}


// ************************************************************************* //
```

Laminar

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     pimpleDyMFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         1;

deltaT          1e-4;
```

```
writeControl    adjustableRunTime;

writeInterval   0.005;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;
functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

```
ddtSchemes
{
    default         Euler;
}

gradSchemes
{
    default         Gauss linear;
    grad(p)         Gauss linear;
    grad(U)         Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      Gauss linearUpwind grad(U);
    div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
    interpolate(HbyA) linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    pcorr           ;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
```

```
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    pcorr
    {
        solver          GAMG;
        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;
        cacheAgglomeration off;
        agglomerator    faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels      1;

        tolerance        0.02;
        relTol           0;
    }

    p
    {
        $pcorr;
        tolerance       1e-06;
        relTol          0.01;
    }

    pFinal
    {
        $p;
        tolerance       1e-06;
        relTol          0;
    }

    U
    {
```

```
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-05;
        relTol          0.01;
    }

    UFinal
    {
        $U;
        tolerance       1e-06;
        relTol          0;
    }

    cellMotionUx
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-08;
        relTol          0;
    }
}

PIMPLE
{
    correctPhi          yes;
    nOuterCorrectors    2;
    nCorrectors         1;
    nNonOrthogonalCorrectors 2;

    pRefCell            0;
    pRefValue           0;
}

relaxationFactors
{
    fields
    {
    }
    equations
    {
        "U.*"           1;
    }
}

// ************************************************************************* //
```

RAS

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     pimpleDyMFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         1;

deltaT          1e-4;

writeControl    adjustableRunTime;

writeInterval   0.005;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;
```

```
adjustTimeStep  yes;

maxCo           1;
functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         Euler;
}

gradSchemes
{
    default         Gauss linear;
    grad(p)         Gauss linear;
    grad(U)         Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      Gauss linearUpwind grad(U);
    div(phi,k)      bounded Gauss upwind;
    div(phi,omega)  bounded Gauss upwind;
```

```
    div((nuEff*dev(T(grad(U))))) Gauss linear;
    div(phi,nuTilda) bounded Gauss upwind;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
    interpolate(HbyA) linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    pcorr           ;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
```

```
{
    pcorr
    {
        solver          GAMG;
        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;
        cacheAgglomeration off;
        agglomerator    faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels      1;

        tolerance        0.02;
        relTol           0;
    }

    p
    {
        $pcorr;
        tolerance       1e-06;
        relTol          0.01;
    }

    pFinal
    {
        $p;
        tolerance       1e-06;
        relTol          0;
    }

    "(U|nuTilda)"
    {
        solver          smoothSolver;
        smoother        GaussSeidel;
        tolerance       1e-05;
        relTol          0.01;
    }

    "(UFinal|nuTildaFinal)"
    {
        $U;
        tolerance       1e-06;
        relTol          0;
    }

    cellMotionUx
    {
```

```
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-08;
        relTol          0;
    }
}

PIMPLE
{
    correctPhi          yes;
    nOuterCorrectors    2;
    nCorrectors         1;
    nNonOrthogonalCorrectors 2;

    pRefCell            0;
    pRefValue           0;
}

relaxationFactors
{
    fields
    {
    }
    equations
    {
        "U.*"           1;
    }
}

// ************************************************************************* //
```

Both

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      decomposeParDict;
}
```

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//- Force AMI to be on single processor. Can cause imbalance with some
//  decomposers.
singleProcessorFaceSets ((AMI -1));

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

numberOfSubdomains 8;

method          scotch;

distributed     no;

roots           ( );


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      snappyHexMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Which of the steps to run
castellatedMesh true;
snap            true;
addLayers       true;


// Geometry. Definition of all surfaces. All surfaces are of class
// searchableSurface.
// Surfaces are used
// - to specify refinement for any mesh cell intersecting it
// - to specify refinement for any mesh cell inside/outside/near
```

```
// - to 'snap' the mesh boundary to the surface
geometry
{
    savonius.stl
    {
        type triSurfaceMesh;
        name savonius;
    }
         /*cylinder.stl
    {
        type triSurfaceMesh;
        name cylinder;
    }*/
    refinementBox1
    {
        type searchableBox;
        min (-0.75 0 -0.75);
        max ( 4  0.005 0.75);
    }
    refinementBox2
    {
        type searchableBox;
        min (-0.2 0 -0.2);
        max ( 0.5  0.005 0.2);
    }
        cylinder
    {
        type searchableCylinder;
        point1 (0 0 0);
            point2 (0 0.005 0);
        radius 0.15;
    }
};



// Settings for the castellatedMesh generation.
castellatedMeshControls
{

    // Refinement parameters
    // ~~~~~~~~~~~~~~~~~~~~~

    // If local number of cells is >= maxLocalCells on any processor
    // switches from from refinement followed by balancing
    // (current method) to (weighted) balancing before refinement.
    maxLocalCells 1000000000;
```

```
// Overall cell limit (approximately). Refinement will stop immediately
// upon reaching this number so a refinement level might not complete.
// Note that this is the number of cells before removing the part which
// is not 'visible' from the keepPoint. The final number of cells might
// actually be a lot less.
maxGlobalCells 2000000000;

// The surface refinement loop might spend lots of iterations refining just a
// few cells. This setting will cause refinement to stop if <= minimumRefine
// are selected for refinement. Note: it will at least do one iteration
// (unless the number of cells to refine is 0)
minRefinementCells 10;

// Allow a certain level of imbalance during refining
// (since balancing is quite expensive)
// Expressed as fraction of perfect balance (= overall number of cells /
// nProcs). 0=balance always.
maxLoadUnbalance 0.10;


// Number of buffer layers between different levels.
// 1 means normal 2:1 refinement restriction, larger means slower
// refinement.
nCellsBetweenLevels 2;



// Explicit feature edge refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies a level for any cell intersected by its edges.
// This is a featureEdgeMesh, read from constant/triSurface for now.
features
(
    {
      file "savonius.eMesh";
      level 6;
   }/*
            {
      file "cylinder.eMesh";
      level 5;
   }*/
);
```

```
// Surface based refinement
// ~~~~~~~~~~~~~~~~~~~~~~~

// Specifies two levels for every surface. The first is the minimum level,
// every cell intersecting a surface gets refined up to the minimum level.
// The second level is the maximum level. Cells that 'see' multiple
// intersections where the intersections make an
// angle > resolveFeatureAngle get refined up to the maximum level.

refinementSurfaces
{
    savonius
    {
        // Surface-wise min and max refinement level
        level (1 5);

        // Optional specification of patch type (default is wall). No
        // constraint types (cyclic, symmetry) etc. are allowed.
       // patchInfo
        //{
          //  type wall;
            //inGroups (savoniusGroup);
        //}
    }
            cylinder
            {
            level       (4 5);

        faceType    boundary;

        cellZone    cylinder;

        faceZone    cylinder;

        cellZoneInside  inside;

                /*level (4 4);

                    cellZone rotating;

                    faceZone rotating;

                    cellZoneInside insidePoint;

                    insidePoint (0 0.001 0.5);
                    /*
                    patchInfo
```

```
                    {
                            type cyclicAMI;
                            neigbourPatch cyclic*;
                            transform noOrdering;
                    }*/
            }
    }
// Resolve sharp angles
resolveFeatureAngle 30;


// Region-wise refinement
// ~~~~~~~~~~~~~~~~~~~~~~

// Specifies refinement level for cells in relation to a surface. One of
// three modes
// - distance. 'levels' specifies per distance to the surface the
//   wanted refinement level. The distances need to be specified in
//   descending order.
// - inside. 'levels' is only one entry and only the level is used. All
//   cells inside the surface get refined up to the level. The surface
//   needs to be closed for this to be possible.
// - outside. Same but cells outside.

refinementRegions
{
   refinementBox1
    {
        mode inside;
        levels ((3 3));
    }
   refinementBox2
    {
        mode inside;
        levels ((3 3));
    }
            cylinder
            {
                    mode inside;
                levels ((4 4));
            }
}


// Mesh selection
// ~~~~~~~~~~~~~~
```

```
    // After refinement patches get added for all refinementSurfaces and
    // all cells intersecting the surfaces get put into these patches. The
    // section reachable from the locationInMesh is kept.
    // NOTE: This point should never be on a face, always inside a cell, even
    // after refinement.
    locationInMesh (1.123573824 0.00100005468436 -1.12345698754);


    // Whether any faceZones (as specified in the refinementSurfaces)
    // are only on the boundary of corresponding cellZones or also allow
    // free-standing zone faces. Not used if there are no faceZones.
    allowFreeStandingZoneFaces true;
}



// Settings for the snapping.
snapControls
{
    //- Number of patch smoothing iterations before finding correspondence
    //  to surface
    nSmoothPatch 3;

    //- Relative distance for points to be attracted by surface feature point
    //  or edge. True distance is this factor times local
    //  maximum edge length.
    tolerance 2.0;

    //- Number of mesh displacement relaxation iterations.
    nSolveIter 30;

    //- Maximum number of snapping relaxation iterations. Should stop
    //  before upon reaching a correct mesh.
    nRelaxIter 5;

    // Feature snapping

        //- Number of feature edge snapping iterations.
        //  Leave out altogether to disable.
        nFeatureSnapIter 10;

        //- Detect (geometric only) features by sampling the surface
        //  (default=false).
        implicitFeatureSnap true;

        //- Use castellatedMeshControls::features (default = true)
        explicitFeatureSnap false;
```

```
        //- Detect points on multiple surfaces (only for explicitFeatureSnap)
        multiRegionFeatureSnap false;
}



// Settings for the layer addition.
addLayersControls
{
    // Are the thickness parameters below relative to the undistorted
    // size of the refined cell outside layer (true) or absolute sizes (false).
    relativeSizes true;

    // Per final patch (so not geometry!) the layer information
    layers
    {
        savonius
        {
            nSurfaceLayers 15;
        }
    }

    // Expansion factor for layer mesh
    expansionRatio 1.1;

    // Wanted thickness of final added cell layer. If multiple layers
    // is the
    // thickness of the layer furthest away from the wall.
    // Relative to undistorted size of cell outside layer.
    // is the thickness of the layer furthest away from the wall.
    // See relativeSizes parameter.
    finalLayerThickness 0.2;

    // Minimum thickness of cell layer. If for any reason layer
    // cannot be above minThickness do not add layer.
    // Relative to undistorted size of cell outside layer.
    minThickness 0.005;

    // If points get not extruded do nGrow layers of connected faces that are
    // also not grown. This helps convergence of the layer addition process
    // close to features.
    // Note: changed(corrected) w.r.t 17x! (didn't do anything in 17x)
    nGrow 0;

    // Advanced settings
```

```
    // When not to extrude surface. 0 is flat surface, 90 is when two faces
    // are perpendicular
    featureAngle 90;

    // At non-patched sides allow mesh to slip if extrusion direction makes
    // angle larger than slipFeatureAngle.
    slipFeatureAngle 30;

    // Maximum number of snapping relaxation iterations. Should stop
    // before upon reaching a correct mesh.
    nRelaxIter 3;

    // Number of smoothing iterations of surface normals
    nSmoothSurfaceNormals 1;

    // Number of smoothing iterations of interior mesh movement direction
    nSmoothNormals 3;

    // Smooth layer thickness over surface patches
    nSmoothThickness 10;

    // Stop layer growth on highly warped cells
    maxFaceThicknessRatio 1;

    // Reduce layer growth where ratio thickness to medial
    // distance is large
    maxThicknessToMedialRatio 0.3;

    // Angle used to pick up medial axis points
    // Note: changed(corrected) w.r.t 17x! 90 degrees corresponds to 130 in 17x.
    minMedianAxisAngle 90;


    // Create buffer region for new layer terminations
    nBufferCellsNoExtrude 0;


    // Overall max number of layer addition iterations. The mesher will exit
    // if it reaches this number of iterations; possibly with an illegal
    // mesh.
    nLayerIter 50;
}



// Generic mesh quality settings. At any undoable phase these determine
// where to undo.
```

```
meshQualityControls
{
    #include "meshQualityDict"


    // Advanced

    //- Number of error distribution iterations
    nSmoothScale 4;
    //- amount to scale back displacement at error points
    errorReduction 0.75;
}


// Advanced

// Write flags
writeFlags
(
    scalarLevels
    layerSets
    layerFields     // write volScalarField for layer coverage
);


// Merge tolerance. Is fraction of overall bounding box of initial mesh.
// Note: the write tolerance needs to be higher than this.
mergeTolerance 1e-6;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      surfaceFeatureExtractDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
```

```
savonius.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   170;
    }

    subsetFeatures
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;

        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}
cylinder.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   170;
    }

    subsetFeatures
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;
```

```
        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      meshQualityDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Include defaults parameters from master dictionary
#include "$WM_PROJECT_DIR/etc/caseDicts/meshQualityDict"

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


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
```

```
forceCoeffs1
{
    type        forceCoeffs;

    functionObjectLibs ( "libforces.so" );

    outputControl   timeStep;
    timeInterval    ;

    log         yes;

    patches     ( savonius);
    rhoName     rhoInf;                     // Indicates incompressible
    rhoInf      1;                          // Redundant for incompressible
    liftDir     (0 0 1);
    dragDir     (1 0 0);
    CofR        (0 0 0);                        // Axle midpoint on ground
    pitchAxis   (0 1 0);
    magUInf     6;
    lRef        0.09;                       // Radius
    Aref        3.636e-3;        // Estimated
/*
    binData
    {
        nBin        20;          // output data into 20 bins
        direction   (1 0 0);     // bin direction
        cumulative  yes;
    }
*/
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      extrudeMeshDict;
}
```

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

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

//constructFrom mesh;
constructFrom patch;
//constructFrom surface;

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

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

//- 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             1;

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       0.01;
}

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;


// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

/*---------------------------------*- C++ -*----------------------------------*\
```

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      createPatchDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// 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
(
    {
        //- Master side patch
        name            AMI1;
        patchInfo
        {
            type            cyclicAMI;
            matchTolerance  0.0001;
            neighbourPatch  AMI2;
            transform       noOrdering;
        }
        constructFrom patches;
        patches (cylinder);
    }

    {
        //- Slave side patch
        name            AMI2;
        patchInfo
        {
            type            cyclicAMI;
            matchTolerance  0.0001;
            neighbourPatch  AMI1;
            transform       noOrdering;
        }
```

```
        constructFrom patches;
        patches (cylinder_slave);
    }/*


    {
        name inlet;
        patchInfo
        {
            type            patch;
        }
        constructFrom set;
        set inletFaces;
    }
    {
        name outlet;
        patchInfo
        {
            type            patch;
        }
        constructFrom set;
        set outletFaces;
    }*/
);

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      topoSetDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

actions
(
    // Get both sides of ami
    // ~~~~~~~~~~~~~~~~~~~~
```

```
    // Get all faces in cellSet
    {
        name    AMI;
        type    faceSet;
        action  new;
        source  patchToFace;
        sourceInfo
        {
            name "AMI.*";
        }
    }
);

// ************************************************************************* //
```

A.5 AMI 3D

A.5.1 Summary

In order to run several cases only changing some parameters has been created a folder called *org* that contains the *hostfile*, the bash scripts *RUN*, *potential* and *SOLVE*, and the folder *sets* that contain all the information of the case. The structure of this foldes is as the follows.

- **sets**
  - **0.org**
    - p
    - U
  - **0RAS**
    - nut
    - $nuTilda$
  - **constant**
    - **polyMesh**
      - $blockMeshDict$
    - **triSurface**
      - $savonius.stl$
  - **cLAM**
    - $transportProperties$
    - $RASproperties$
  - **cRAS**
    - $transportProperties$
    - $turbulenceProperties$
    - $RASproperties$
  - **sPOT**
    - $controlDict$
    - $fvSolution$
  - **sLAM**
    - $controlDict$
    - $fvSolution$
  - **sRAS**
    - $controlDict$
    - $fvSolution$
  - **system**
    - $decomposeParDict$
    - $snappyHexMeshDict$
    - $surfaceFeatureExtractDict$
    - $meshQualityDict$
    - $forceCoeffs$
    - $createPatchDict$
    - $topoSetDict$

Bash scripts

The script *RUN* prepare the case:

```
#!/bin/bash
rm -r POT LAM RAS
mkdir POT LAM RAS
#Potential
cp sets/constant sets/0.org sets/system POT -R
cp sets/sPOT/controlDict sets/sPOT/fvSchemes sets/sPOT/fvSolution POT/system
cp hostfile POT
```

```
./potential
cp POT/constant POT/0 LAM -R
cp POT/constant POT/0 RAS -R

#LAMINAR
cp sets/system LAM -R
cp sets/sLAM/controlDict sets/sLAM/fvSchemes sets/sLAM/fvSolution LAM/system
cp sets/cLAM/RASProperties sets/cLAM/transportProperties  sets/cLAM/turbulenceProper

#RAS
cp sets/system RAS -R
cp sets/sRAS/controlDict sets/sRAS/fvSchemes sets/sRAS/fvSolution RAS/system
cp sets/cRAS/RASProperties sets/cRAS/transportProperties RAS/constant
cp sets/0RAS/nut sets/0RAS/nuTilda RAS/0

./SOLVE
```

potential

```
#!/bin/bash
cd POT
rm 0 -r
cp 0.org 0 -R
blockMesh > log.blockMesh
checkMesh > log.checkMesh
surfaceFeatureExtract > log.surfaceFeatureExtract
foamJob -s snappyHexMesh -overwrite> log.snappyHexMesh
#extrudeMesh > log.extrudeMesh
createPatch -overwrite> log.createPatch
topoSet > log.topoSet
checkMesh -allGeometry -allTopology > log.checkMeshAll
rm -R 0
cp 0.org 0 -R
potentialFoam > log.potentialFoam
paraFoam -builtin -touch
cd ..
```

SOLVE

```
#!/bin/bash
#LAMINAR
cd LAM
decomposePar >log.decomposePar
mpirun -np 8 pimpleDyMFoam -parallel > log.pimpleDyMFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
#RAS TURBULENCE
```

```
cd RAS
decomposePar >log.decomposePar
mpirun -np 8 pimpleDyMFoam -parallel > log.pimpleDyMFoam
reconstructPar > log.reconstructPar
rm -R processor*
paraFoam -builtin -touch
cd ..
```

A.5.2 0

The boundary conditions defined are U:

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    location    "0";
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (6 0 0);

boundaryField
{

    inlet
        {
                type                          freestream;
                freestreamValue        uniform (6 0 0);
        }
    outlet
    {
        type                         freestream;
                freestreamValue        uniform (6 0 0);
```

```
    }

        up
        {
                type                        slip;
        }

        down
        {
                type                        slip;
        }

        front
        {
                type                        slip;
        }

        back
        {
                type                        slip;
        }

   savonius
    {
        type            movingWallVelocity; // fixedValue;
        value           uniform (0 0 0);

    }
    AMI1
    {
        type            cyclicAMI;
        value           $internalField;
    }

    AMI2
    {
        type            cyclicAMI;
        value           $internalField;
    }
}


// ************************************************************************* //
```

The pressure is defined in p.

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
```

```
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      p;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //


dimensions      [0 2 -2 0 0 0 0];

internalField   uniform 0;

boundaryField
{

    inlet
    {
        type            freestreamPressure;
    }

    outlet
    {
        type            freestreamPressure;
    }
        up
        {
                type                        slip;
        }
        down
        {
                type                        slip;
        }

        back
        {
                type                        slip;
        }
           front
        {
                type                        slip;
```

```
        }

    savonius
    {
        type            zeroGradient;
    }
    AMI1
    {
        type            cyclicAMI;
        value           $internalField;
    }

    AMI2
    {
        type            cyclicAMI;
        value           $internalField;
    }
}

// ************************************************************************* //
```

A.5.3 constant

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1;

vertices
(
    (-2 -1.5 -1.5)
    (4 -1.5 -1.5)
    (4 1.5 -1.5)
    (-2 1.5 -1.5)
```

```
    (-2 -1.5 1.5)
    (4 -1.5 1.5)
    (4 1.5 1.5)
    (-2 1.5 1.5)
);


blocks
(
    hex (0 1 2 3 4 5 6 7) (50 25 25) simpleGrading (1 1 1)
);

edges
(
);

boundary
(
    front
    {
        type patch;
        faces
        (
            (1 5 4 0)
        );
    }
        back
        {
        type patch;
        faces
        (
            (3 7 6 2)
        );
    }

    inlet
    {
        type patch;
        faces
        (
            (0 4 7 3)
        );
    }
    outlet
    {
        type patch;
        faces
```

```
        (
            (2 6 5 1)
        );
    }
           down
    {
        type patch;
        faces
        (
            (0 3 2 1)
        );
    }
    up
    {
        type patch;
        faces
        (
            (4 5 6 7)
        );
    }
);

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      dynamicMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dynamicFvMesh   solidBodyMotionFvMesh;

motionSolverLibs ( "libfvMotionSolvers.so" );

solidBodyMotionFvMeshCoeffs
{
    cellZone        cylinder;
```

```
    solidBodyMotionFunction  rotatingMotion;
    rotatingMotionCoeffs
    {
        origin        (0 0 0);
        axis          (0 1 0);
        omega         39.6; // rad/s
    }
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

transportModel  Newtonian;

nu              nu [0 2 -1 0 0 0 0] 1.5e-05;

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
```

```
    object      turbulenceProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

simulationType  RASModel;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            laminar;

turbulence          off;

printCoeffs         on;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      RASProperties;
}
```

```
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

RASModel            SpalartAllmaras;

turbulence          on;

printCoeffs         on;


// ************************************************************************* //
```

A.5.4 system

Potential

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     potentialFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         1;

deltaT          1;

writeControl    timeStep;

writeInterval   1;
```

```
purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

functions
{
    difference
    {
        // Load the library containing the 'coded' functionObject
        functionObjectLibs ("libutilityFunctionObjects.so");
        type coded;
        // Name of on-the-fly generated functionObject
        redirectType error;
        code
        #{
            // Lookup U
            Info<< "Looking up field U\n" << endl;
            const volVectorField& U = mesh().lookupObject<volVectorField>("U");

            Info<< "Reading inlet velocity  uInfX\n" << endl;

            scalar ULeft = 0.0;
            label leftI = mesh().boundaryMesh().findPatchID("inlet");
            const fvPatchVectorField& fvp = U.boundaryField()[leftI];
            if (fvp.size())
            {
                ULeft = fvp[0].x();
            }
            reduce(ULeft, maxOp<scalar>());

            dimensionedScalar uInfX
            (
                "uInfx",
                dimensionSet(0, 1, -1, 0, 0),
                ULeft
            );
```

```
Info << "U at inlet = " << uInfX.value() << " m/s" << endl;


scalar magCylinder = 0.0;
label cylI = mesh().boundaryMesh().findPatchID("savonius");
const fvPatchVectorField& cylFvp = mesh().C().boundaryField()[cylI];
if (cylFvp.size())
{
    magCylinder = mag(cylFvp[0]);
}
reduce(magCylinder, maxOp<scalar>());

dimensionedScalar radius
(
    "radius",
    dimensionSet(0, 1, 0, 0, 0),
    magCylinder
);

Info << "Cylinder radius = " << radius.value() << " m" << endl;

volVectorField UA
(
    IOobject
    (
        "UA",
        mesh().time().timeName(),
        U.mesh(),
        IOobject::NO_READ,
        IOobject::AUTO_WRITE
    ),
    U
);

Info<< "\nEvaluating analytical solution" << endl;

const volVectorField& centres = UA.mesh().C();
volScalarField magCentres(mag(centres));
volScalarField theta(acos((centres & vector(1,0,0))/magCentres));

volVectorField cs2theta
(
    cos(2*theta)*vector(1,0,0)
  + sin(2*theta)*vector(0,1,0)
);

UA = uInfX*(dimensionedVector(vector(1,0,0))
```

```
              - pow((radius/magCentres),2)*cs2theta);

            // Force writing of UA (since time has not changed)
            UA.write();

            volScalarField error("error", mag(U-UA)/mag(UA));

            Info<<"Writing relative error in U to " << error.objectPath()
                << endl;

            error.write();
        #};
    }
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
    default         steadyState;
}

gradSchemes
{
    default         leastSquares;
}

divSchemes
{
    default         none;
```

```
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    p
    {
        solver          PCG;
```

```
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0;
    }
}

potentialFlow
{
    nNonOrthogonalCorrectors 2;
        pRefPoint        (0.2546565 0.000054651354 -0.15466496);
        pRefValue        0;
}


// ************************************************************************* //
```

Laminar

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     pimpleDyMFoam;

startFrom       startTime;

startTime       0;

stopAt          endTime;

endTime         2;

deltaT          1e-4;

writeControl    adjustableRunTime;
```

```
writeInterval   0.01;

purgeWrite      0;

writeFormat     binary;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;

adjustTimeStep  yes;

maxCo           1;
functions
{
    #include "readFields"
    #include "streamLines"
    #include "wallBoundedStreamLines"
    #include "cuttingPlane"
    #include "forceCoeffs"
}
// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
```

```
{
    default         Euler;
}

gradSchemes
{
    default         Gauss linear;
    grad(p)         Gauss linear;
    grad(U)         Gauss linear 1;
}

divSchemes
{
    default         none;
    div(phi,U)      Gauss linearUpwind grad(U);
    div((nuEff*dev(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
    interpolate(HbyA) linear;
}

snGradSchemes
{
    default         corrected;
}

fluxRequired
{
    default         no;
    pcorr           ;
    p               ;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
```

```
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

solvers
{
    pcorr
    {
        solver          GAMG;
        smoother        GaussSeidel;
        nPreSweeps      0;
        nPostSweeps     2;
        cacheAgglomeration off;
        agglomerator    faceAreaPair;
        nCellsInCoarsestLevel 10;
        mergeLevels      1;

        tolerance        0.02;
        relTol           0;
    }

    p
    {
        $pcorr;
        tolerance       1e-06;
        relTol          0.01;
    }

    pFinal
    {
        $p;
        tolerance       1e-06;
        relTol          0;
    }

    U
    {
        solver          smoothSolver;
```

```
        smoother        GaussSeidel;
        tolerance       1e-05;
        relTol          0.01;
    }

    UFinal
    {
        $U;
        tolerance       1e-06;
        relTol          0;
    }

    cellMotionUx
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-08;
        relTol          0;
    }
}

PIMPLE
{
    correctPhi          yes;
    nOuterCorrectors    2;
    nCorrectors         1;
    nNonOrthogonalCorrectors 2;

    pRefCell            0;
    pRefValue           0;
}

relaxationFactors
{
    fields
    {
    }
    equations
    {
        "U.*"           1;
    }
}

// ************************************************************************* //
```

Both

```
/*--------------------------------*- C++ -*----------------------------------*\
```

```
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      decomposeParDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//- Force AMI to be on single processor. Can cause imbalance with some
//  decomposers.
singleProcessorFaceSets ((AMI -1));

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

numberOfSubdomains 8;

method          scotch;

distributed     no;

roots           ( );


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      snappyHexMeshDict;
```

```
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Which of the steps to run
castellatedMesh true;
snap            true;
addLayers       true;


// Geometry. Definition of all surfaces. All surfaces are of class
// searchableSurface.
// Surfaces are used
// - to specify refinement for any mesh cell intersecting it
// - to specify refinement for any mesh cell inside/outside/near
// - to 'snap' the mesh boundary to the surface
geometry
{
    savonius.stl
    {
        type triSurfaceMesh;
        name savonius;
    }
         /*cylinder.stl
    {
        type triSurfaceMesh;
        name cylinder;
    }*/
    refinementBox
    {
        type searchableBox;
        min (-0.75 -0.75 -0.75);
        max ( 4  0.75 0.75);
    }
        cylinder
    {
        type searchableCylinder;
        point1 (0 -0.2 0);
            point2 (0 0.2 0);
        radius 0.15;
    }
};



// Settings for the castellatedMesh generation.
castellatedMeshControls
{
```

```
// Refinement parameters
// ~~~~~~~~~~~~~~~~~~~~

// If local number of cells is >= maxLocalCells on any processor
// switches from from refinement followed by balancing
// (current method) to (weighted) balancing before refinement.
maxLocalCells 100000000;

// Overall cell limit (approximately). Refinement will stop immediately
// upon reaching this number so a refinement level might not complete.
// Note that this is the number of cells before removing the part which
// is not 'visible' from the keepPoint. The final number of cells might
// actually be a lot less.
maxGlobalCells 200000000;

// The surface refinement loop might spend lots of iterations refining just a
// few cells. This setting will cause refinement to stop if <= minimumRefine
// are selected for refinement. Note: it will at least do one iteration
// (unless the number of cells to refine is 0)
minRefinementCells 10;

// Allow a certain level of imbalance during refining
// (since balancing is quite expensive)
// Expressed as fraction of perfect balance (= overall number of cells /
// nProcs). 0=balance always.
maxLoadUnbalance 0.10;


// Number of buffer layers between different levels.
// 1 means normal 2:1 refinement restriction, larger means slower
// refinement.
nCellsBetweenLevels 2;



// Explicit feature edge refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies a level for any cell intersected by its edges.
// This is a featureEdgeMesh, read from constant/triSurface for now.
features
(
    {
      file "savonius.eMesh";
      level 6;
   }/*
```

```
            {
      file "cylinder.eMesh";
      level 5;
   }*/
);



// Surface based refinement
// ~~~~~~~~~~~~~~~~~~~~~~~~

// Specifies two levels for every surface. The first is the minimum level,
// every cell intersecting a surface gets refined up to the minimum level.
// The second level is the maximum level. Cells that 'see' multiple
// intersections where the intersections make an
// angle > resolveFeatureAngle get refined up to the maximum level.

refinementSurfaces
{
    savonius
    {
        // Surface-wise min and max refinement level
        level (1 5);

        // Optional specification of patch type (default is wall). No
        // constraint types (cyclic, symmetry) etc. are allowed.
       // patchInfo
        //{
          //  type wall;
            //inGroups (savoniusGroup);
        //}
    }
            cylinder
            {
            level       (4 5);

        faceType    boundary;

        cellZone    cylinder;

        faceZone    cylinder;

        cellZoneInside  inside;

                /*level (4 4);

                    cellZone rotating;
```

```
                    faceZone rotating;

                    cellZoneInside insidePoint;

                    insidePoint (0 0.001 0.5);
                    /*
                    patchInfo
                    {
                            type cyclicAMI;
                            neigbourPatch cyclic*;
                            transform noOrdering;
                    }*/
            }
    }
// Resolve sharp angles
resolveFeatureAngle 30;


// Region-wise refinement
// ~~~~~~~~~~~~~~~~~~~~~~

// Specifies refinement level for cells in relation to a surface. One of
// three modes
// - distance. 'levels' specifies per distance to the surface the
//   wanted refinement level. The distances need to be specified in
//   descending order.
// - inside. 'levels' is only one entry and only the level is used. All
//   cells inside the surface get refined up to the level. The surface
//   needs to be closed for this to be possible.
// - outside. Same but cells outside.

refinementRegions
{
   refinementBox
    {
        mode inside;
        levels ((3 3));
    }
            cylinder
            {
                    mode inside;
                levels ((4 4));
            }
}
```

```
    // Mesh selection
    // ~~~~~~~~~~~~~~

    // After refinement patches get added for all refinementSurfaces and
    // all cells intersecting the surfaces get put into these patches. The
    // section reachable from the locationInMesh is kept.
    // NOTE: This point should never be on a face, always inside a cell, even
    // after refinement.
    locationInMesh (1.123573824 0.00100005468436 -1.12345698754);


    // Whether any faceZones (as specified in the refinementSurfaces)
    // are only on the boundary of corresponding cellZones or also allow
    // free-standing zone faces. Not used if there are no faceZones.
    allowFreeStandingZoneFaces true;
}



// Settings for the snapping.
snapControls
{
    //- Number of patch smoothing iterations before finding correspondence
    //  to surface
    nSmoothPatch 3;

    //- Relative distance for points to be attracted by surface feature point
    //  or edge. True distance is this factor times local
    //  maximum edge length.
    tolerance 2.0;

    //- Number of mesh displacement relaxation iterations.
    nSolveIter 30;

    //- Maximum number of snapping relaxation iterations. Should stop
    //  before upon reaching a correct mesh.
    nRelaxIter 5;

    // Feature snapping

        //- Number of feature edge snapping iterations.
        //  Leave out altogether to disable.
        nFeatureSnapIter 10;

        //- Detect (geometric only) features by sampling the surface
        //  (default=false).
        implicitFeatureSnap true;
```

```
        //- Use castellatedMeshControls::features (default = true)
        explicitFeatureSnap false;

        //- Detect points on multiple surfaces (only for explicitFeatureSnap)
        multiRegionFeatureSnap false;
}



// Settings for the layer addition.
addLayersControls
{
    // Are the thickness parameters below relative to the undistorted
    // size of the refined cell outside layer (true) or absolute sizes (false).
    relativeSizes true;

    // Per final patch (so not geometry!) the layer information
    layers
    {
        savonius
        {
            nSurfaceLayers 15;
        }
    }

    // Expansion factor for layer mesh
    expansionRatio 1.1;

    // Wanted thickness of final added cell layer. If multiple layers
    // is the
    // thickness of the layer furthest away from the wall.
    // Relative to undistorted size of cell outside layer.
    // is the thickness of the layer furthest away from the wall.
    // See relativeSizes parameter.
    finalLayerThickness 0.2;

    // Minimum thickness of cell layer. If for any reason layer
    // cannot be above minThickness do not add layer.
    // Relative to undistorted size of cell outside layer.
    minThickness 0.05;

    // If points get not extruded do nGrow layers of connected faces that are
    // also not grown. This helps convergence of the layer addition process
    // close to features.
    // Note: changed(corrected) w.r.t 17x! (didn't do anything in 17x)
    nGrow 0;
```

```
    // Advanced settings

    // When not to extrude surface. 0 is flat surface, 90 is when two faces
    // are perpendicular
    featureAngle 90;

    // At non-patched sides allow mesh to slip if extrusion direction makes
    // angle larger than slipFeatureAngle.
    slipFeatureAngle 30;

    // Maximum number of snapping relaxation iterations. Should stop
    // before upon reaching a correct mesh.
    nRelaxIter 3;

    // Number of smoothing iterations of surface normals
    nSmoothSurfaceNormals 1;

    // Number of smoothing iterations of interior mesh movement direction
    nSmoothNormals 3;

    // Smooth layer thickness over surface patches
    nSmoothThickness 10;

    // Stop layer growth on highly warped cells
    maxFaceThicknessRatio 1;

    // Reduce layer growth where ratio thickness to medial
    // distance is large
    maxThicknessToMedialRatio 0.3;

    // Angle used to pick up medial axis points
    // Note: changed(corrected) w.r.t 17x! 90 degrees corresponds to 130 in 17x.
    minMedianAxisAngle 90;


    // Create buffer region for new layer terminations
    nBufferCellsNoExtrude 0;


    // Overall max number of layer addition iterations. The mesher will exit
    // if it reaches this number of iterations; possibly with an illegal
    // mesh.
    nLayerIter 50;
}
```

```
// Generic mesh quality settings. At any undoable phase these determine
// where to undo.
meshQualityControls
{
    #include "meshQualityDict"


    // Advanced

    //- Number of error distribution iterations
    nSmoothScale 4;
    //- amount to scale back displacement at error points
    errorReduction 0.75;
}


// Advanced

// Write flags
writeFlags
(
    scalarLevels
    layerSets
    layerFields     // write volScalarField for layer coverage
);


// Merge tolerance. Is fraction of overall bounding box of initial mesh.
// Note: the write tolerance needs to be higher than this.
mergeTolerance 1e-6;


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      surfaceFeatureExtractDict;
```

```
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

savonius.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   170;
    }

    subsetFeatures
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;

        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}
cylinder.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   170;
    }

    subsetFeatures
```

```
    {
        // Keep nonManifold edges (edges with >2 connected faces)
        nonManifoldEdges       no;

        // Keep open edges (edges with 1 connected face)
        openEdges       yes;
    }


    // Write options

        // Write features to obj format for postprocessing
        writeObj                yes;
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      meshQualityDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Include defaults parameters from master dictionary
#include "$WM_PROJECT_DIR/etc/caseDicts/meshQualityDict"

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


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
```

```
|    \\/     M anipulation  |                                                 |
\*-----------------------------------------------------------------------------*/

forceCoeffs1
{
    type        forceCoeffs;

    functionObjectLibs ( "libforces.so" );

    outputControl   timeStep;
    timeInterval    ;

    log         yes;

    patches     ( savonius);
    rhoName     rhoInf;                     // Indicates incompressible
    rhoInf      1;                          // Redundant for incompressible
    liftDir     (0 0 1);
    dragDir     (1 0 0);
    CofR        (0 0 0);                        // Axle midpoint on ground
    pitchAxis   (0 1 0);
    magUInf     6;
    lRef        0.09;                       // Radius
    Aref        4.915229e-3;        // Estimated
/*
    binData
    {
        nBin        20;          // output data into 20 bins
        direction   (1 0 0);     // bin direction
        cumulative  yes;
    }
*/
}


// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*-----------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
```

```
    class       dictionary;
    object      createPatchDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// 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
(
    {
        //- Master side patch
        name            AMI1;
        patchInfo
        {
            type            cyclicAMI;
            matchTolerance  0.0001;
            neighbourPatch  AMI2;
            transform       noOrdering;
        }
        constructFrom patches;
        patches (cylinder);
    }

    {
        //- Slave side patch
        name            AMI2;
        patchInfo
        {
            type            cyclicAMI;
            matchTolerance  0.0001;
            neighbourPatch  AMI1;
            transform       noOrdering;
        }
        constructFrom patches;
        patches (cylinder_slave);
    }/*


    {
        name inlet;
        patchInfo
        {
            type            patch;
```

```
        }
        constructFrom set;
        set inletFaces;
    }
    {
        name outlet;
        patchInfo
        {
            type            patch;
        }
        constructFrom set;
        set outletFaces;
    }*/
);

// ************************************************************************* //

/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  2.3.1                                 |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      topoSetDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

actions
(
    // Get both sides of ami
    // ~~~~~~~~~~~~~~~~~~~~

    // Get all faces in cellSet
    {
        name    AMI;
        type    faceSet;
        action  new;
        source  patchToFace;
        sourceInfo
        {
            name "AMI.*";
        }
```

```
    }
);

// ************************************************************************* //
```

Appendix B

Convergence of the results

B.1 Octave scripts of the results convergence

B.1.1 General script

```
%Pere Antoni Martorell
%TFG Savonius
%Plot Convergence
clear
close all
%clc
Coeff=dlmread('postProcessing/forceCoeffs1/0/forceCoeffs.dat');
Time=Coeff(:,1);
tf=length(Time);
Time=Coeff(10:tf,1);
%Coeff=Coeff*2e-4/0.028;
Cm=Coeff(10:tf,2);
Cd=Coeff(10:tf,3);
Cl=Coeff(10:tf,4);
Cl_f=Coeff(10:tf,5);
Cl_r=Coeff(10:tf,6);

figure
plot(Time,Cm);
xlabel('Time')
ylabel('Cm')
print('Cm.png')

figure
plot(Time,Cd);
xlabel('Time')
ylabel('Cd')
print('Cd.png')

figure
plot(Time,Cl,'bk');
xlabel('Time')
ylabel('Cl')
print('Cl.png')

%%Average values
```

```
Tcon=0.004;
tf=length(Time);
t0=int16(tf*(Tcon/Time(tf)));
CD=mean(Cd(t0:tf))
CL=mean(Cl(t0:tf))
CM=mean(Cm(t0:tf))
save("Coefficients.mat",'CD','CL','CM');
%exit
```

B.1.2 Angular position searching

```
%Pere Antoni Martorell
%TFG Savonius
%Plot Coefficient
clear
close all
clc
for i=1:6
    a(i)=(i-1)*30;
    Coeff=dlmread(sprintf('%1.0fdegrees/RAS/postProcessing/forceCoeffs1/0/forceCoeffs.dat'
    %Coeff1=dlmread(sprintf('%1.0fdegrees/LAM/postProcessing/forceCoeffs1/0/forceCoeffs.da
    Time1=Coeff(:,1);
    tf=length(Time1);
    %Coeff=Coeff;Coeff1(10:tf,:);
    %Time1=Coeff1(:,1);
    %tf=length(Time1);

    Time(:,i)=Coeff(10:tf,1);
    Coeff=Coeff*2; %Surface reference mistake correction
    Cm(:,i)=Coeff(10:tf,2);
    Cd(:,i)=Coeff(10:tf,3);
    Cl(:,i)=Coeff(10:tf,4);
    Cl_f(:,i)=Coeff(10:tf,5);
    Cl_r(:,i)=Coeff(10:tf,6);
    %%Average values
    Tcon=0.002;
    tf=length(Time);
    t0=int16(tf*(Tcon/Time(tf)));
    CD(i)=mean(Cd(t0:tf,i));
    CL(i)=mean(Cl(t0:tf,i));
    CM(i)=mean(Cm(t0:tf,i));
end

a(7)=180;
CD(7)=CD(1);
CL(7)=CL(1);
CM(7)=CM(1);
CL=-CL;

figure
plot(a,CD, 'bk',a,CD,'bko')
ylabel('CD')
xlabel('Angular position [degrees]')
xlim([0 180])
set(gca,'XTick',0:30:180)
set(gca,'YTick',-2:0.1:3)
title('Drag coefficient')
```

```
grid on
print('Cd2DRROT.png')


figure
plot(a,CL,'bk', a,CL,'bko')
ylabel('CL')
xlabel('Angular position [degrees]')
set(gca,'XTick',0:30:180)
set(gca,'YTick',-2:0.1:2)
title('Lift coefficient')
xlim([0 180])
grid on
print('Cl2DRROT.png')

figure
plot(a,CM, 'bk',a,CM,'bko')
ylabel('CM')
xlabel('Angular position [degrees]')
set(gca,'XTick',0:30:180)
set(gca,'YTick',-2:0.05:2)
title('Static torque coefficient')
xlim([0 180])
grid on
print('Cm2DRROT.png')

dlmwrite('Coeff2DRROT.dat', [a' CD' CL' CM']);
```

B.1.3 AMI results processing

```
%Pere Antoni Martorell
%TFG Savonius
%Plot Convergence
clear
close all
%clc
Coeff=dlmread('postProcessing/forceCoeffs1/0/forceCoeffs.dat');
Time=Coeff(:,1);
tf=length(Time);
Time=Coeff(10:tf,1);
Coeff=Coeff*2*6^2/9^2;
%Coeff=Coeff*2e-4/0.028;
Cm=Coeff(10:tf,2)*6/9;
Cd=Coeff(10:tf,3);
Cl=Coeff(10:tf,4);
Cl_f=Coeff(10:tf,5);
Cl_r=Coeff(10:tf,6);

ANGLE=79.2*180/pi*Time;
Angle=ANGLE;
tf=length(Angle);
Ant=1;
j=1;
while max(Angle)>180
    for i=1:tf
        if Angle(i)>180
            Angle(i:tf)=Angle(i:tf)-180;
```

```
            %CM1(:,j)=Cm(Ant:i);
            %Ant=i;
            %j=j+1;
        end
        if ANGLE(i)<900.0001
            t1=i;
        end
        if ANGLE(i)<1080.0001
            t2=i;
        end
    end
end
CM1=Cm(t1:t2);
CL1=Cl(t1:t2);
CD1=Cd(t1:t2);
Angle1=ANGLE(t1:t2)-900;
figure
plot(Angle1,CM1,'bk');
xlabel('Anlge [degrees]')
ylabel('Ct')
set(gca,'XTick',0:30:10000)
set(gca,'YTick',-2:0.1:10)
xlim([0 180])
grid on
print('AMICm1.png')

figure
plot(Angle1,CD1,'bk');
xlabel('Anlge [degrees]')
ylabel('Cd')
set(gca,'XTick',0:30:10000)
set(gca,'YTick',-2:0.1:10)
xlim([0 180])
grid on
print('AMICd1.png')

figure
plot(Angle1,CL1,'bk');
xlabel('Anlge [degrees]')
ylabel('Cl')
set(gca,'XTick',0:30:10000)
set(gca,'YTick',-2:0.1:10)
xlim([0 180])
grid on
print('AMICl1.png')



figure
plot(ANGLE,Cm,'bk');
xlabel('Anlge [degrees]')
ylabel('Ct')
set(gca,'XTick',0:180:10000)
set(gca,'YTick',-2:0.1:10)
grid on
print('AMICm.png')

figure
```

```
plot(ANGLE,Cd,'bk');
xlabel('Anlge [degrees]')
ylabel('Cd')
set(gca,'XTick',0:180:10000)
set(gca,'YTick',-2:0.2:10)
grid on
print('AMICd.png')

figure
plot(ANGLE,Cl,'bk');
xlabel('Anlge [degrees]')
ylabel('Cl');
set(gca,'XTick',0:180:10000)
set(gca,'YTick',-2:0.2:10)
grid on
ylim([-2 4])
print('AMICl.png')

%%Average values
Tcon=0.15;
tf=length(Time);
t0=int16(tf*(Tcon/Time(tf)));
CD=mean(Cd(t0:tf))
CL=mean(Cl(t0:tf))
CM=mean(Cm(t0:tf))
save("Coefficients.mat",'CD','CL','CM');
%exit

dlmwrite('CoeffAMILAM.dat', [Angle1 CD1 CL1 CM1])
```

B.2 Mesh convergence table

In the Table B.1 are shown the parameters of the meshes tested.

C_t	BM discretization	BM cells	Max level	Box dim.	Box level	Box cell size	Min cell size	Layers	Layer (%)	Cells
0.336	50	31250	7	0.2	3	0.015	0.001	3.27	66.5	2415214
0.696	100	250000	6	0.2	4	0.004	0.001	4.33	88.6	2767732
0.384	50	31250	7	0.2	4	0.007	0.001	4.51	91.2	5214971
0.383	50	31250	7	0.2	4	0.007	0.001	4.08	83.2	3525990
0.372	50	31250	7	0.2	4	0.007	0.001	6.67	80.3	4087378
0.530	50	31250	7	0.2	5	0.004	0.001	4.51	91.2	14491449
0.179	100	250000	6	0.2	3	0.007	0.001	4.48	90.5	4835703
0.210	200	2000000	5	0.2	2	0.007	0.001	4.48	90.5	6538387
0.216	200	2000000	5	0.5	2	0.007	0.001	4.48	90.5	15477421
0.445	200	2000000	5	0.5	3	0.004	0.001	4.47	90.5	15606522
0.120	200	2000000	5	0.5	2	0.007	0.001	5.90	69.4	16062818
0.545	50	31250	7	0.2	6	0.002	0.001	4.48	90.5	22303163

Table B.1: Characteristics of the tested 3D meshes

B.3 Plots of the results convergence

B.3.1 2D

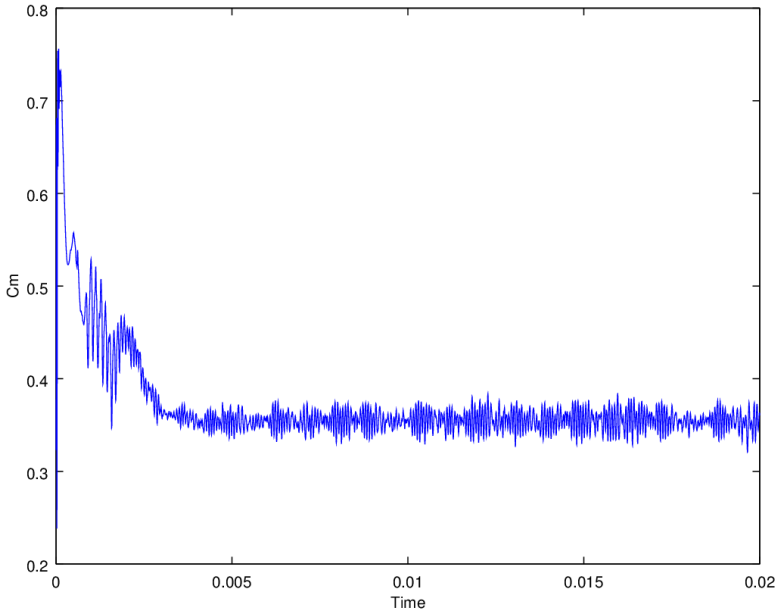


Figure B.1: Convergence of the LAM simulation at 0 degrees in a FIX boundary condition with the 2D mesh

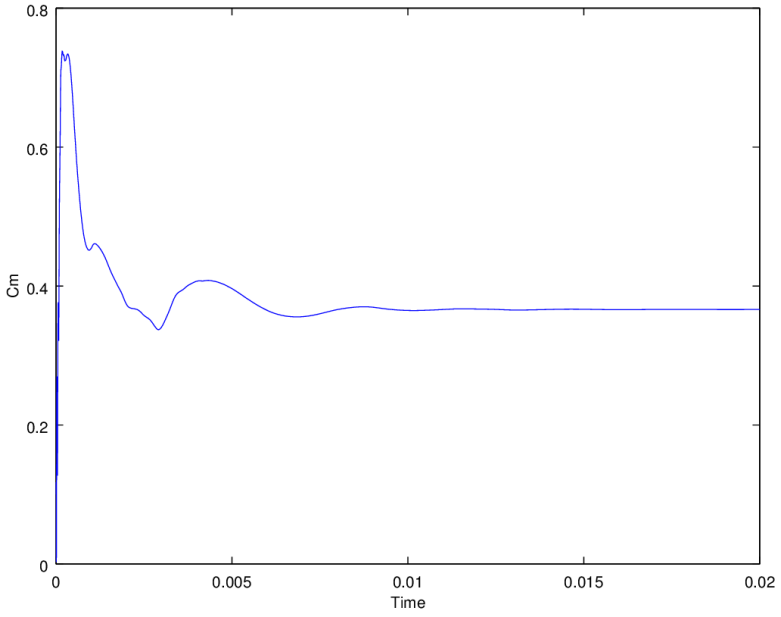


Figure B.2: Convergence of the RAS simulation at 0 degrees in a FIX boundary condition with the 2D mesh

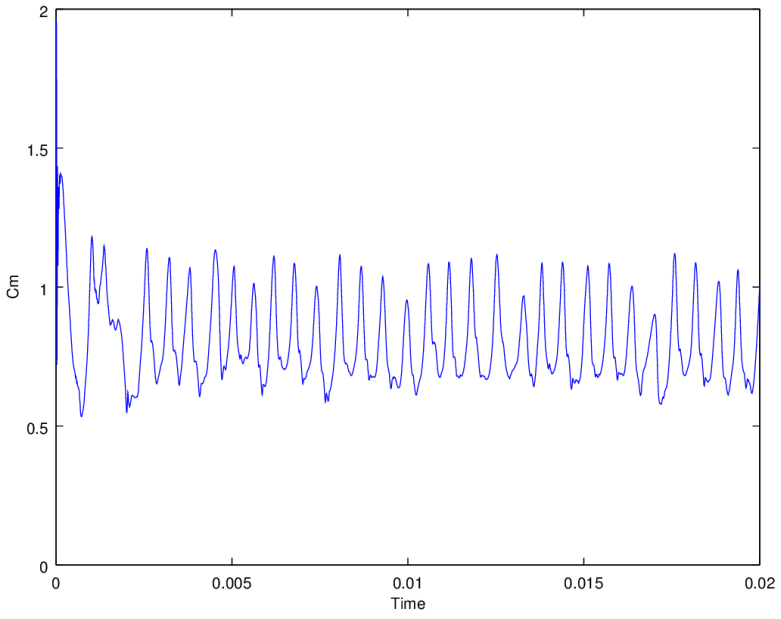


Figure B.3: Convergence of the LAM simulation at 30 degrees in a FIX boundary condition with the 2D mesh

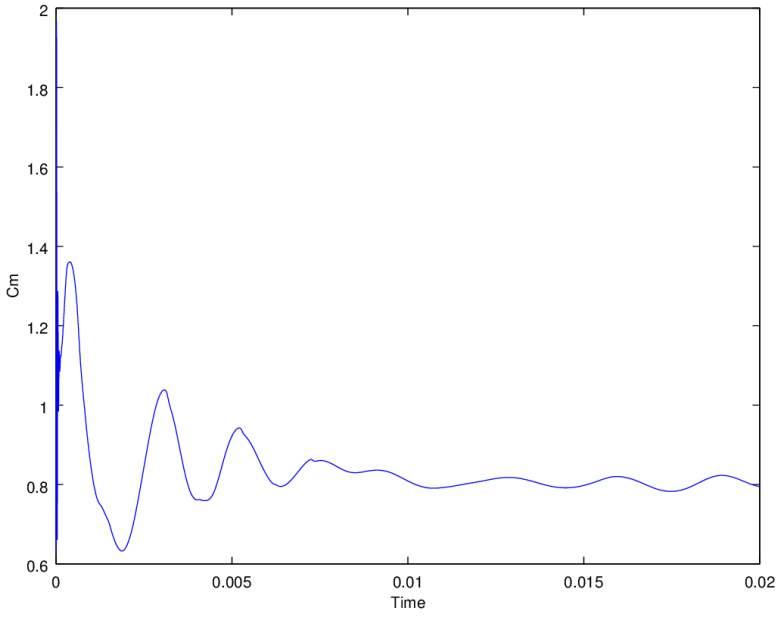


Figure B.4: Convergence of the RAS simulation at 30 degrees in a FIX boundary condition with the 2D mesh

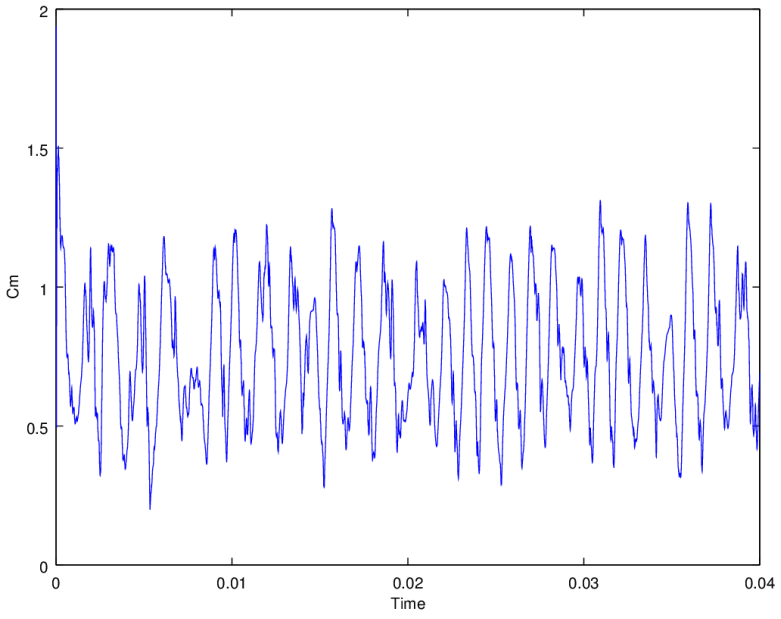


Figure B.5: Convergence of the LAM simulation at 60 degrees in a FIX boundary condition with the 2D mesh

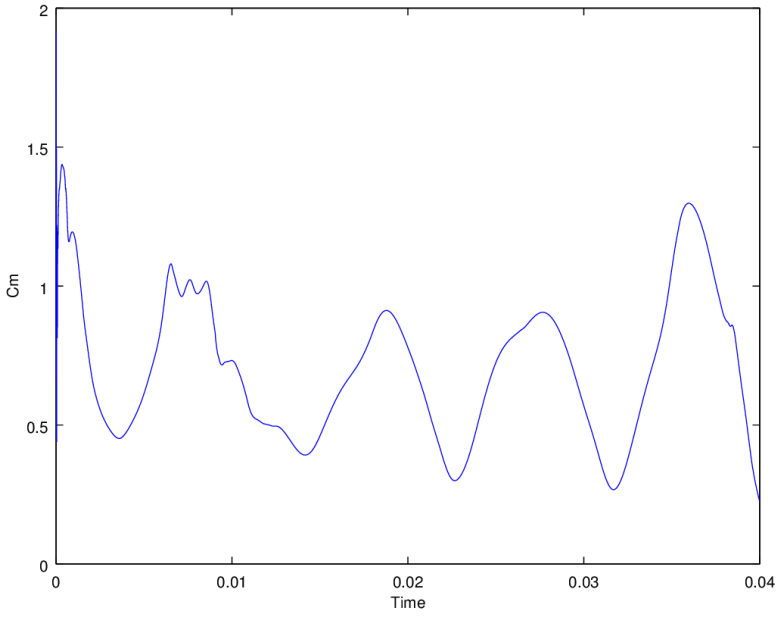


Figure B.6: Convergence of the RAS simulation at 60 degrees in a FIX boundary condition with the 2D mesh

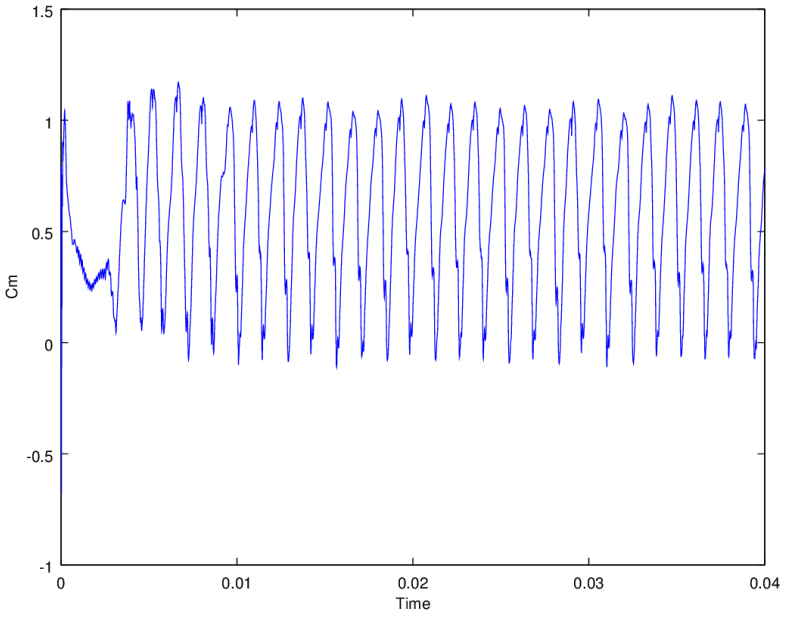


Figure B.7: Convergence of the LAM simulation at 90 degrees in a FIX boundary condition with the 2D mesh

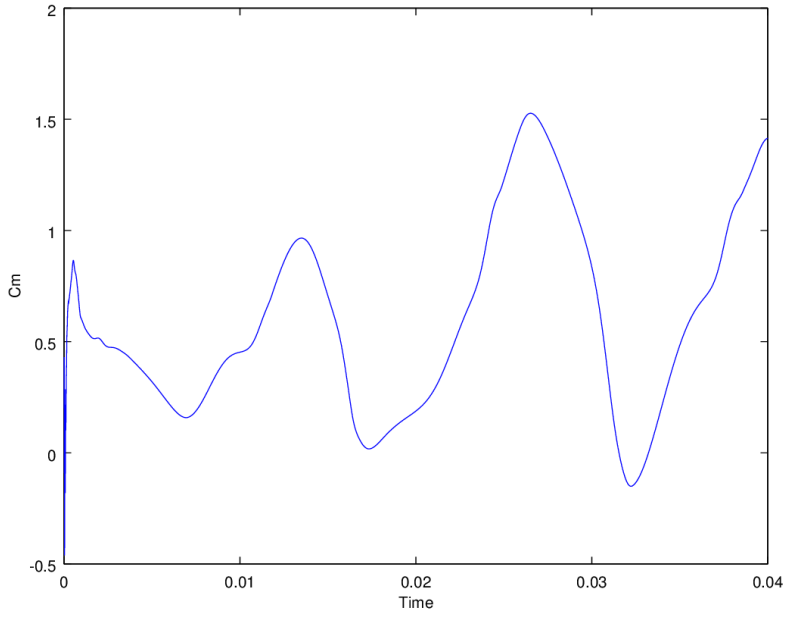


Figure B.8: Convergence of the RAS simulation at 90 degrees in a FIX boundary condition with the 2D mesh

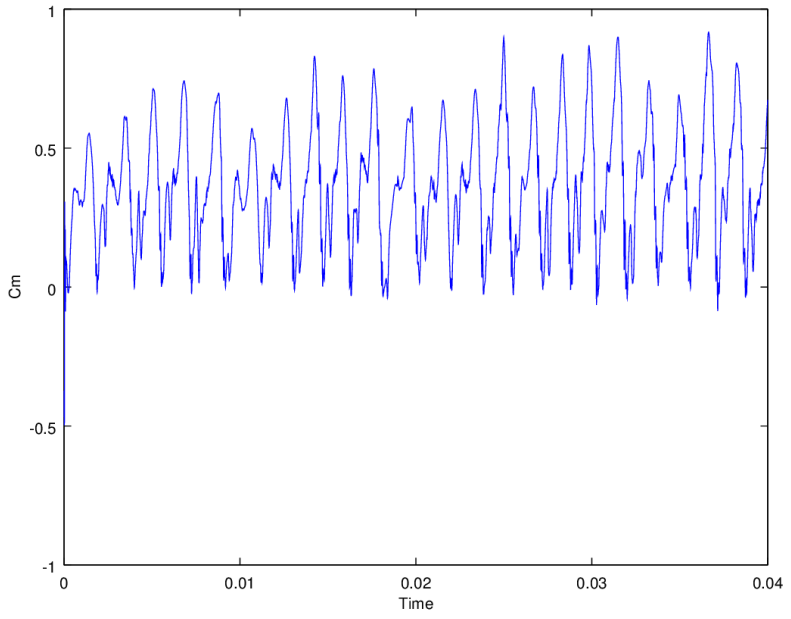


Figure B.9: Convergence of the LAM simulation at 120 degrees in a FIX boundary condition with the 2D mesh

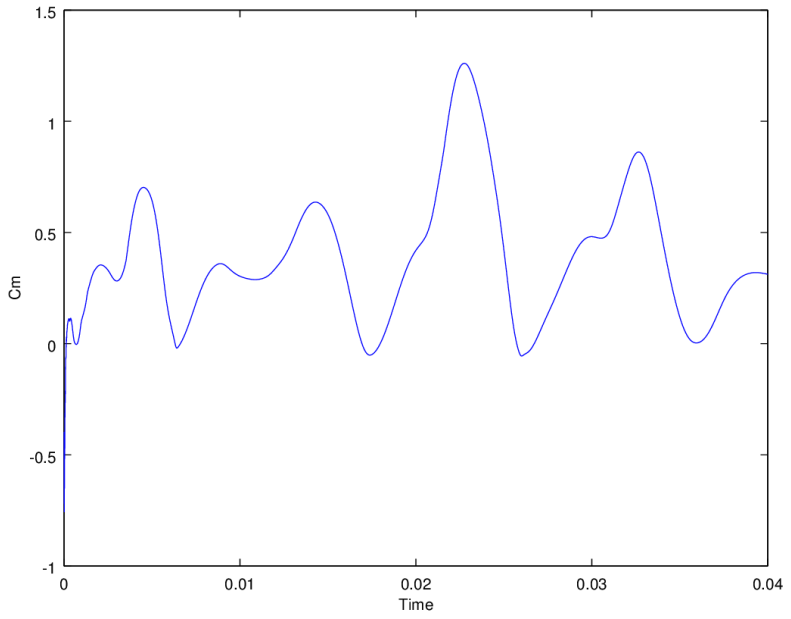


Figure B.10: Convergence of the RAS simulation at 120 degrees in a FIX boundary condition with the 2D mesh

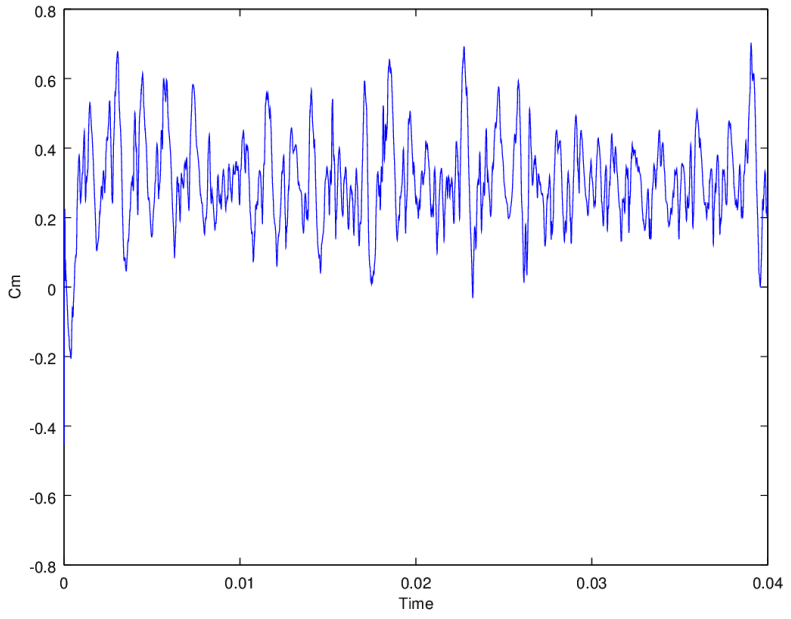


Figure B.11: Convergence of the LAM simulation at 150 degrees in a FIX boundary condition with the 2D mesh

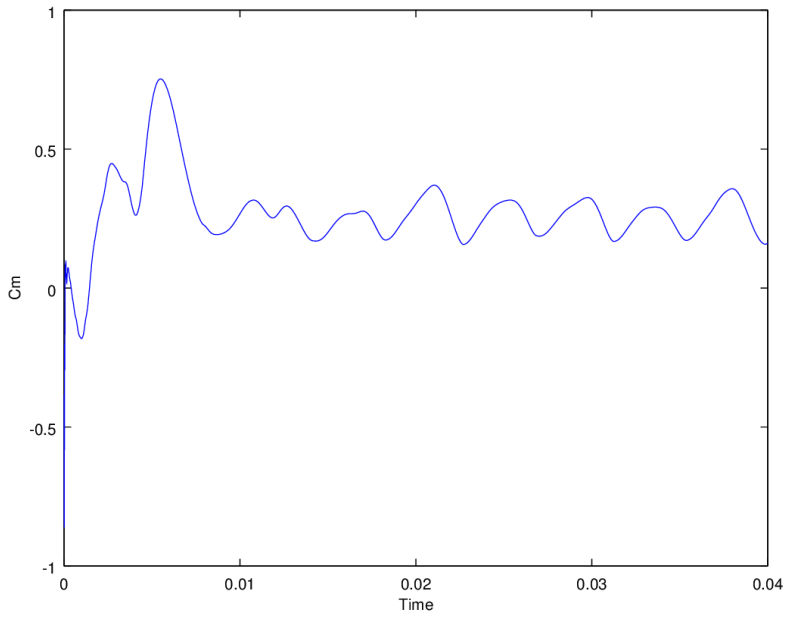


Figure B.12: Convergence of the RAS simulation at 150 degrees in a FIX boundary condition with the 2D mesh

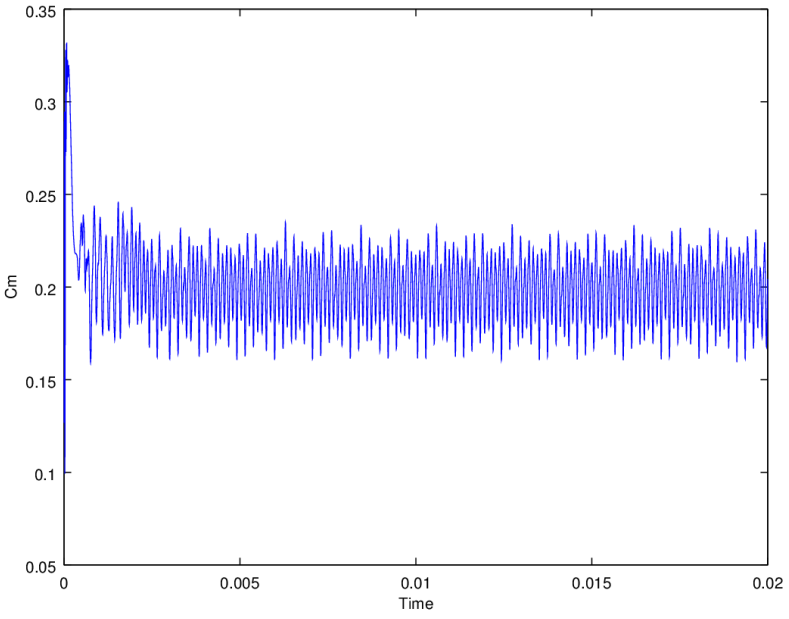


Figure B.13: Convergence of the LAM simulation at 0 degrees in a ROT boundary condition with the 2D mesh

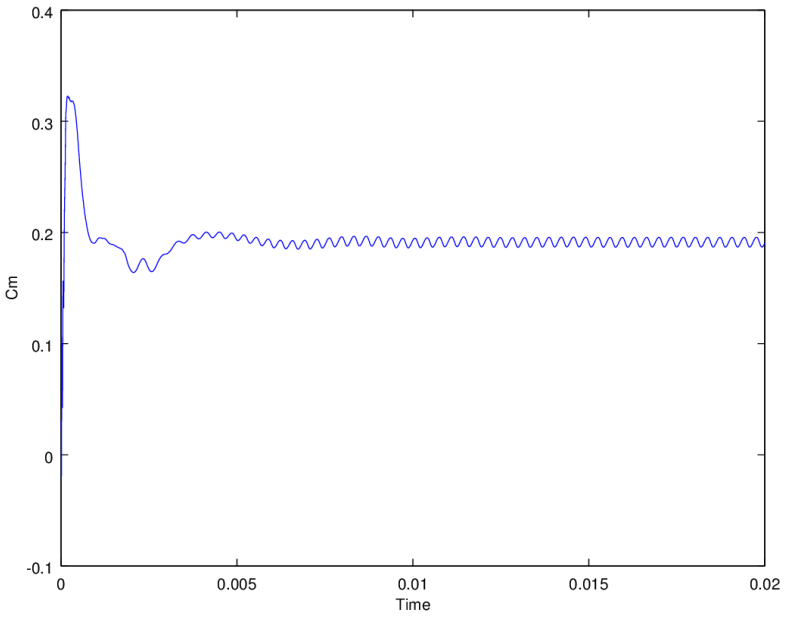


Figure B.14: Convergence of the RAS simulation at 0 degrees in a ROT boundary condition with the 2D mesh

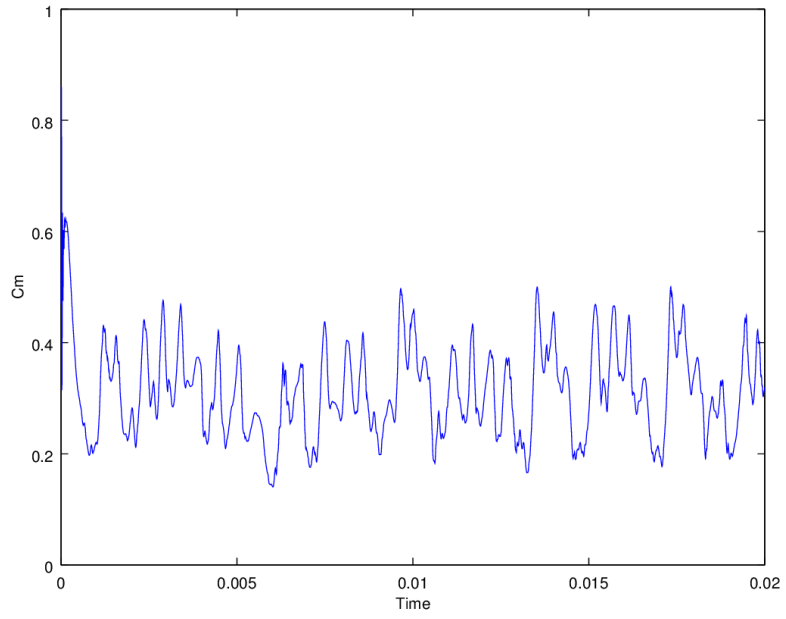


Figure B.15: Convergence of the LAM simulation at 30 degrees in a ROT boundary condition with the 2D mesh

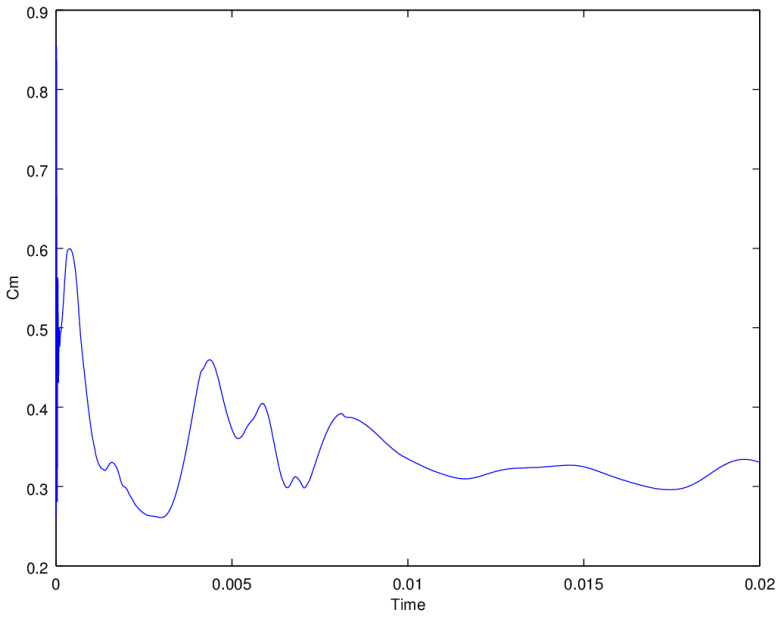


Figure B.16: Convergence of the RAS simulation at 30 degrees in a ROT boundary condition with the 2D mesh

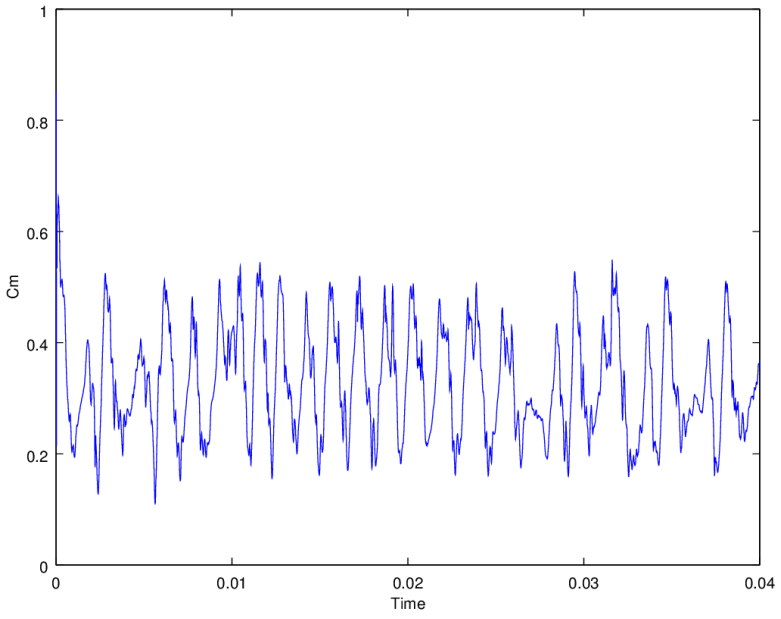


Figure B.17: Convergence of the LAM simulation at 60 degrees in a ROT boundary condition with the 2D mesh

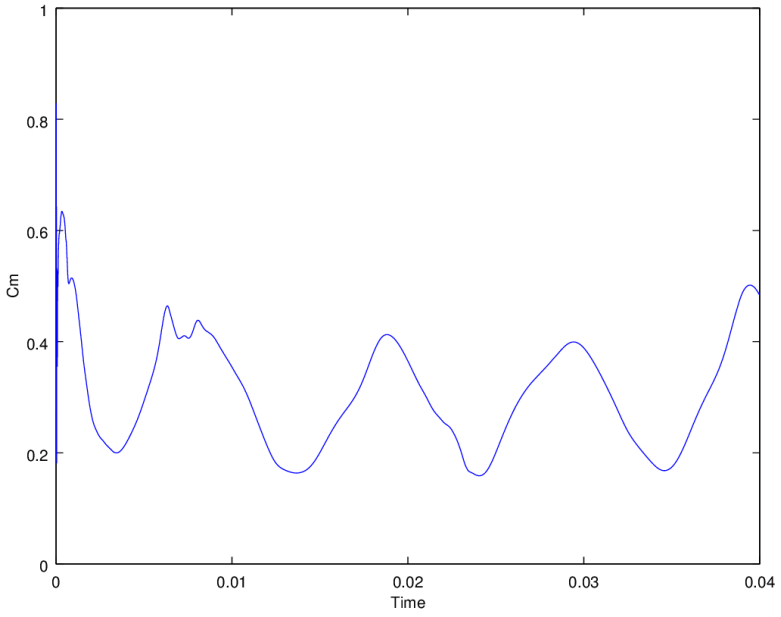


Figure B.18: Convergence of the RAS simulation at 60 degrees in a ROT boundary condition with the 2D mesh

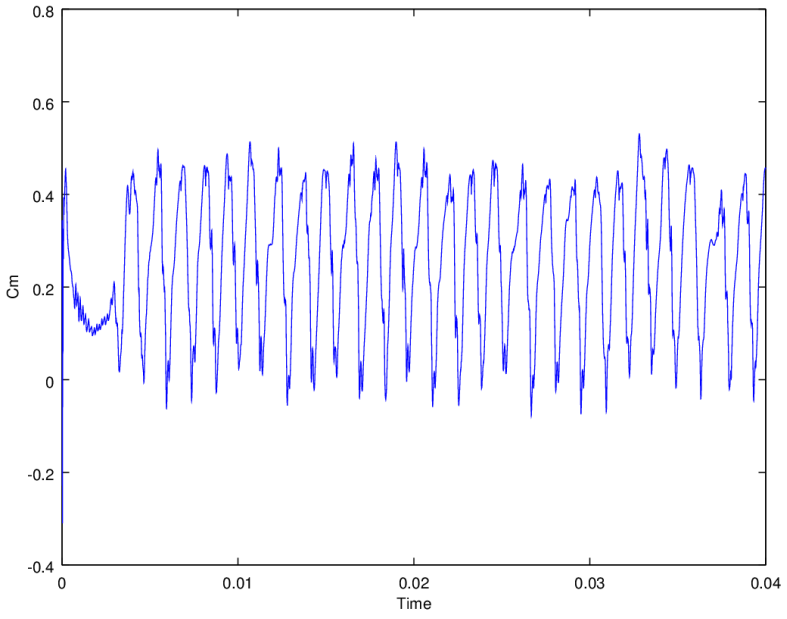


Figure B.19: Convergence of the LAM simulation at 90 degrees in a ROT boundary condition with the 2D mesh

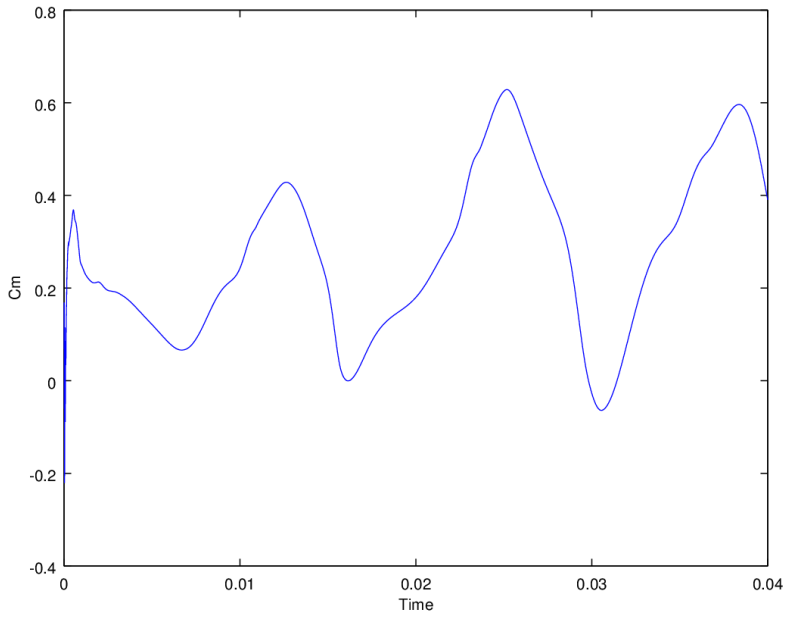


Figure B.20: Convergence of the RAS simulation at 90 degrees in a ROT boundary condition with the 2D mesh

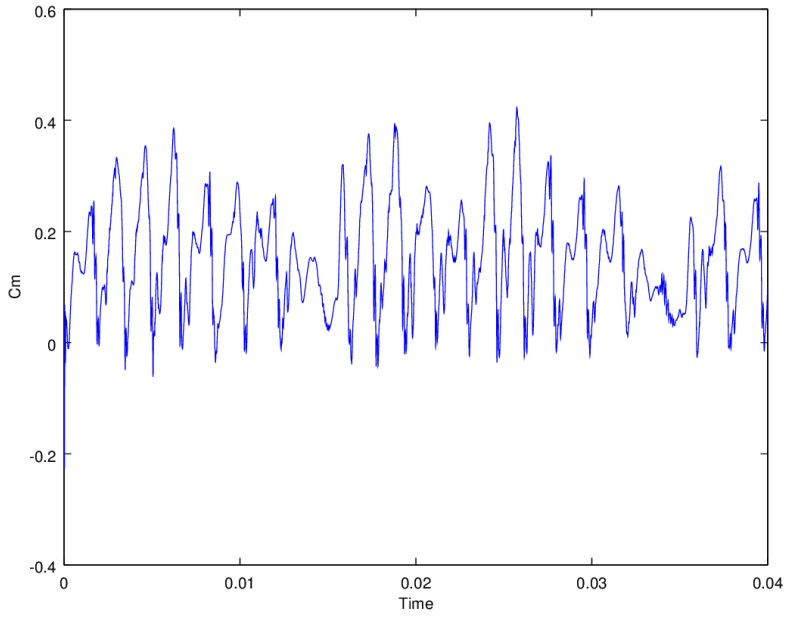


Figure B.21: Convergence of the LAM simulation at 120 degrees in a ROT boundary condition with the 2D mesh

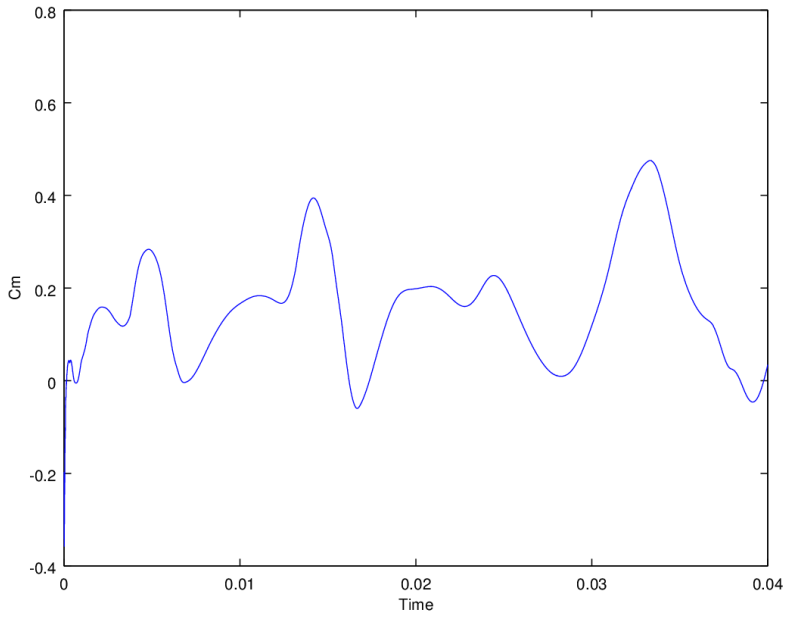


Figure B.22: Convergence of the RAS simulation at 120 degrees in a ROT boundary condition with the 2D mesh

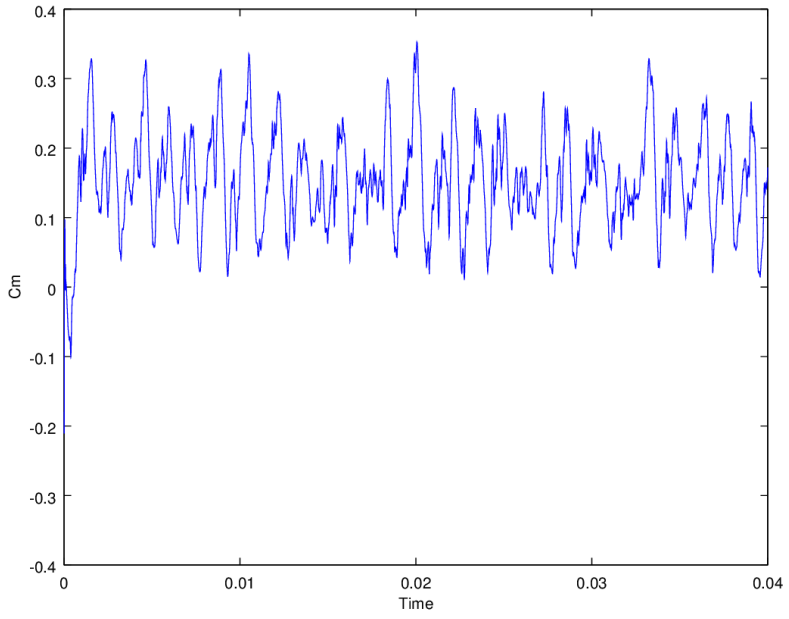


Figure B.23: Convergence of the LAM simulation at 150 degrees in a ROT boundary condition with the 2D mesh

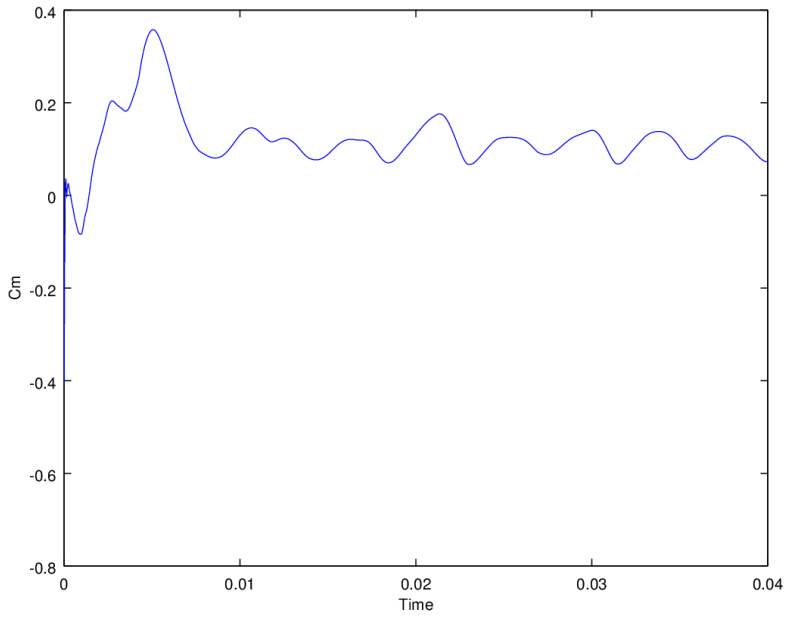


Figure B.24: Convergence of the RAS simulation at 150 degrees in a ROT boundary condition with the 2D mesh

B.3.2 3D

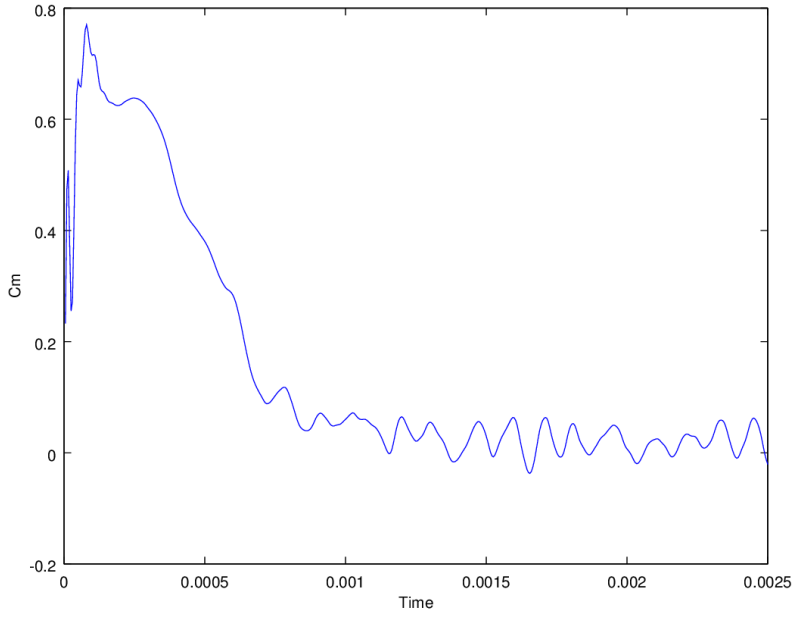


Figure B.25: Convergence of the LAM simulation at 0 degrees in a FIX boundary condition with the 3D mesh

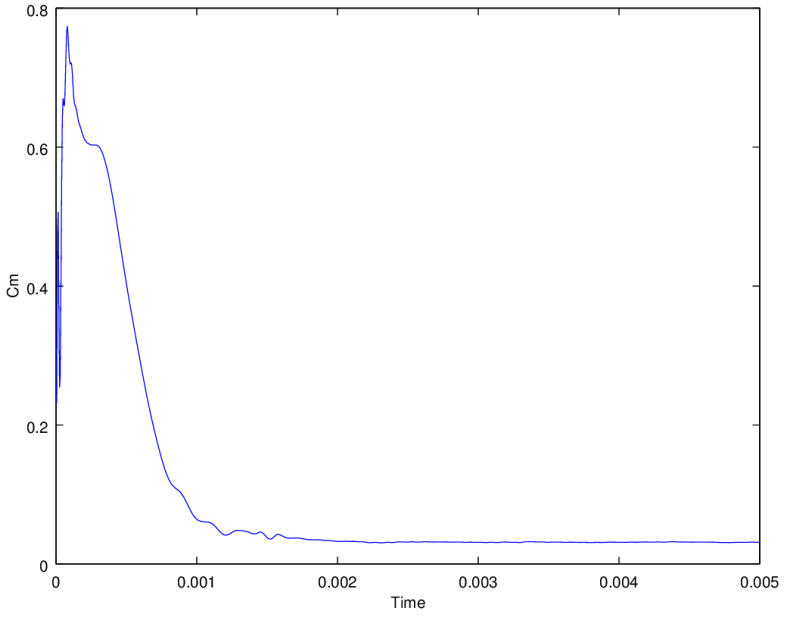


Figure B.26: Convergence of the RAS simulation at 0 degrees in a FIX boundary condition with the 3D mesh

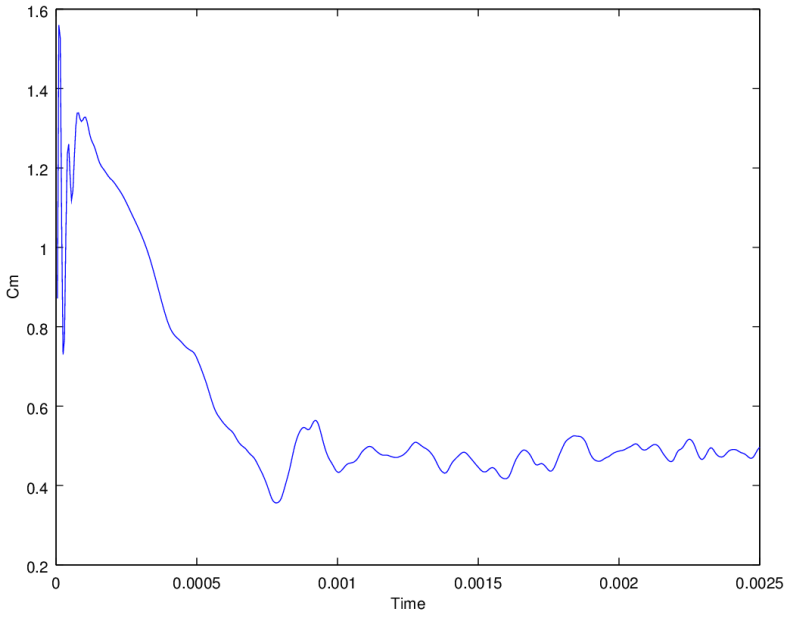


Figure B.27: Convergence of the LAM simulation at 30 degrees in a FIX boundary condition with the 3D mesh

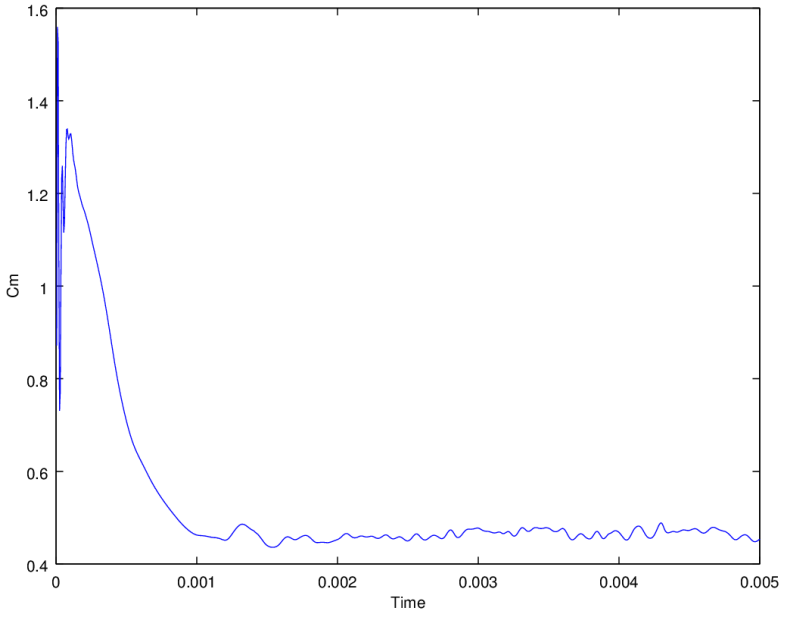


Figure B.28: Convergence of the RAS simulation at 30 degrees in a FIX boundary condition with the 3D mesh

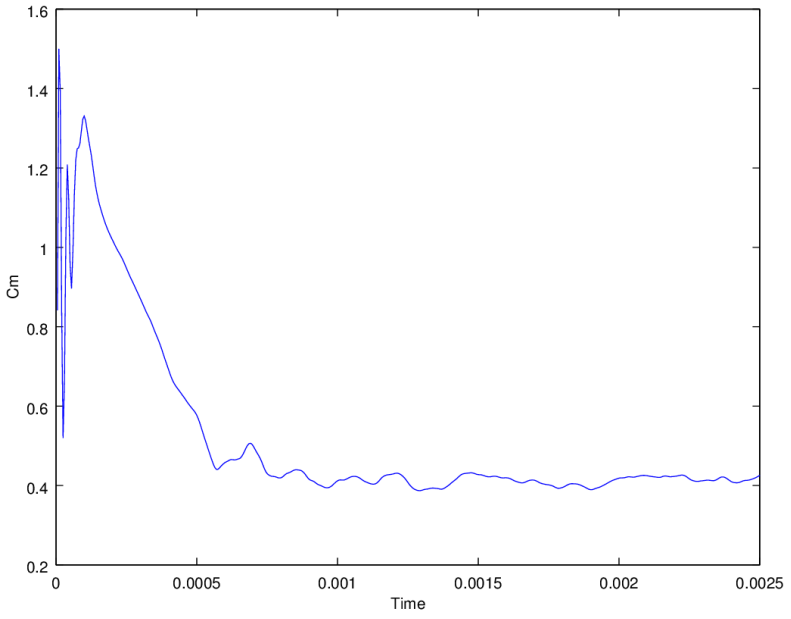


Figure B.29: Convergence of the LAM simulation at 60 degrees in a FIX boundary condition with the 3D mesh

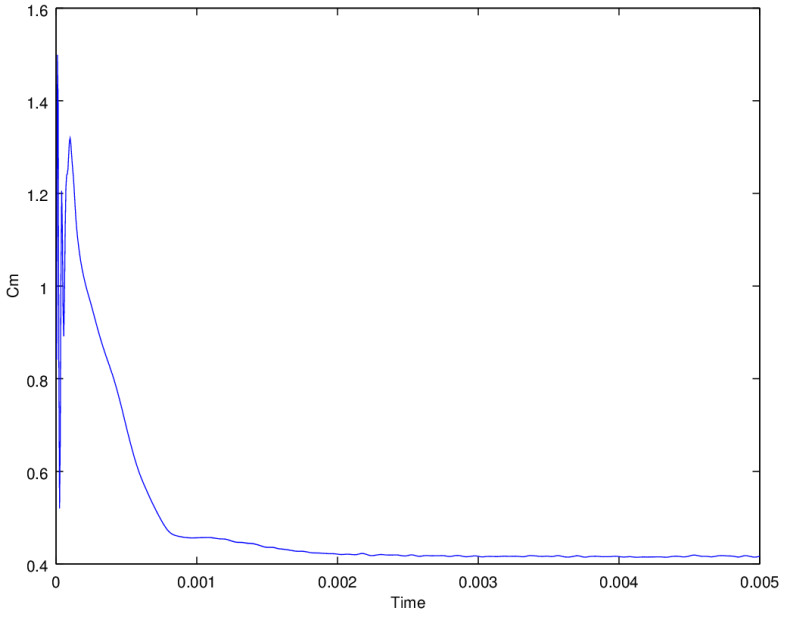


Figure B.30: Convergence of the RAS simulation at 60 degrees in a FIX boundary condition with the 3D mesh

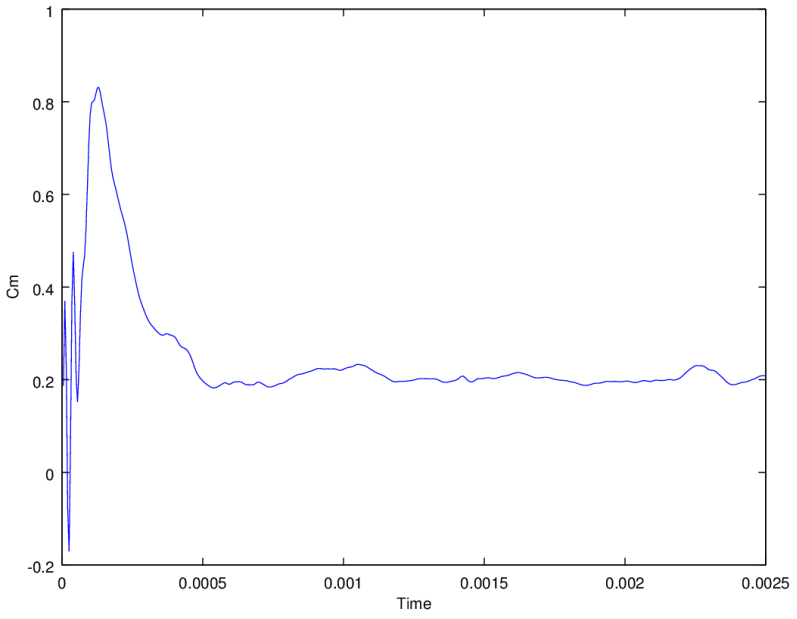


Figure B.31: Convergence of the LAM simulation at 90 degrees in a FIX boundary condition with the 3D mesh

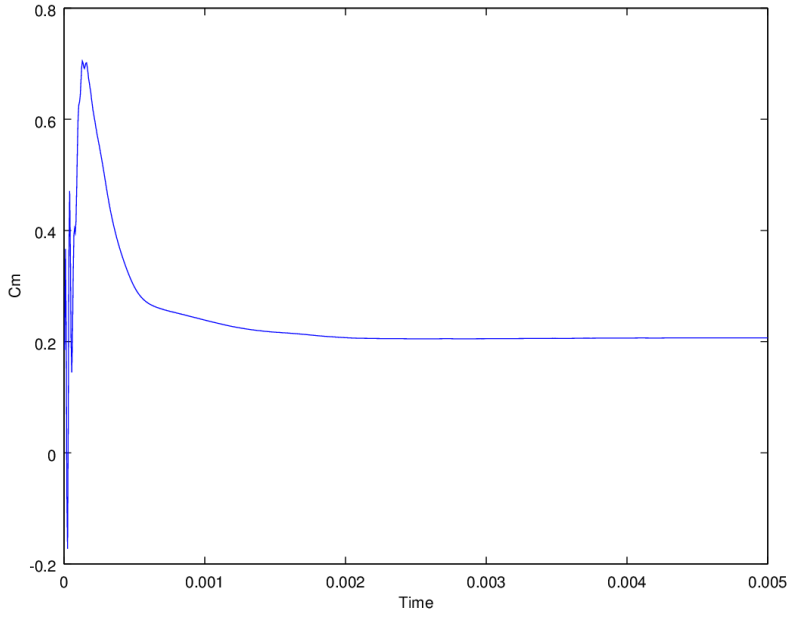


Figure B.32: Convergence of the RAS simulation at 90 degrees in a FIX boundary condition with the 3D mesh

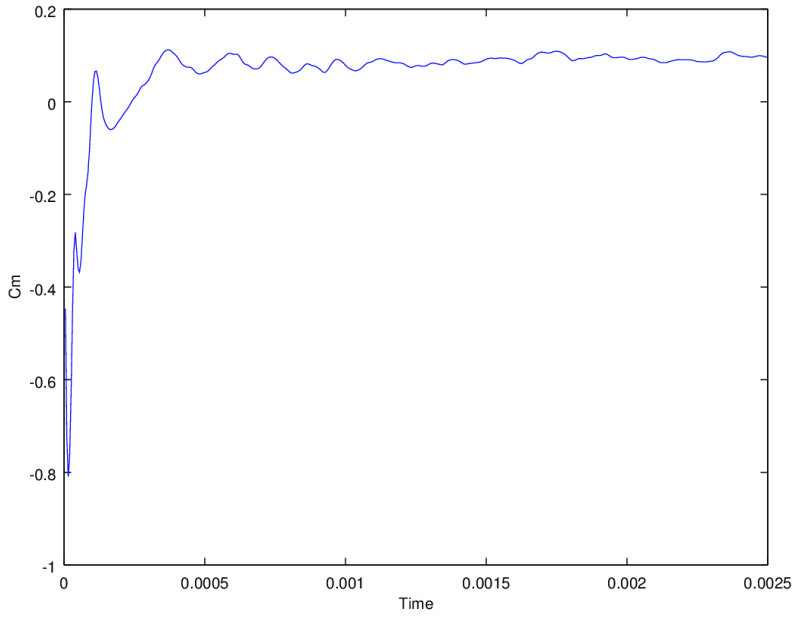


Figure B.33: Convergence of the LAM simulation at 120 degrees in a FIX boundary condition with the 3D mesh

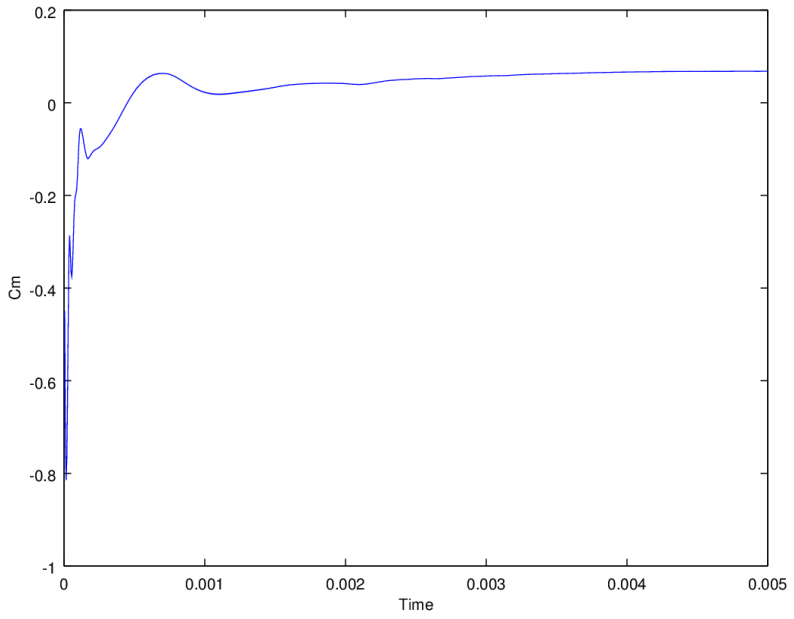


Figure B.34: Convergence of the RAS simulation at 120 degrees in a FIX boundary condition with the 3D mesh

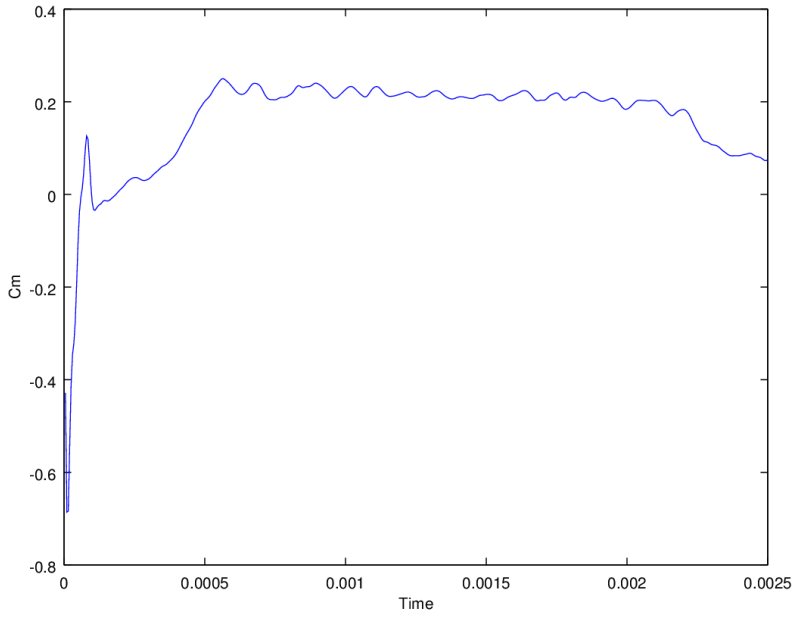


Figure B.35: Convergence of the LAM simulation at 150 degrees in a FIX boundary condition with the 3D mesh

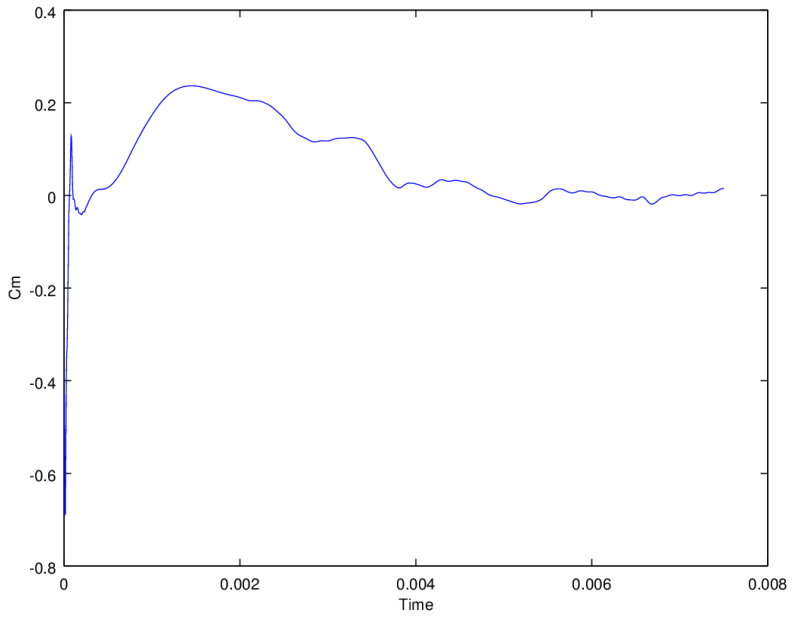


Figure B.36: Convergence of the RAS simulation at 150 degrees in a FIX boundary condition with the 3D mesh

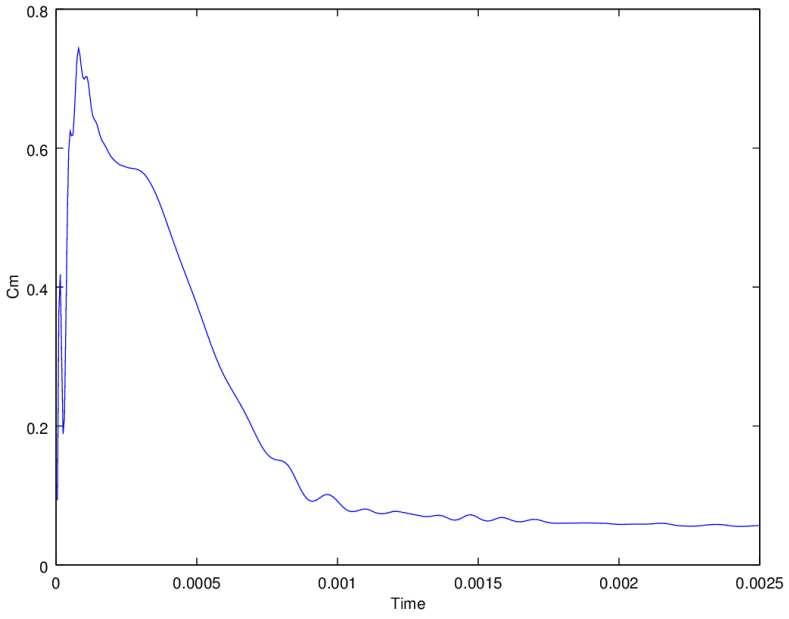


Figure B.37: Convergence of the RAS simulation at 0 degrees in a ROT boundary condition with the 3D mesh

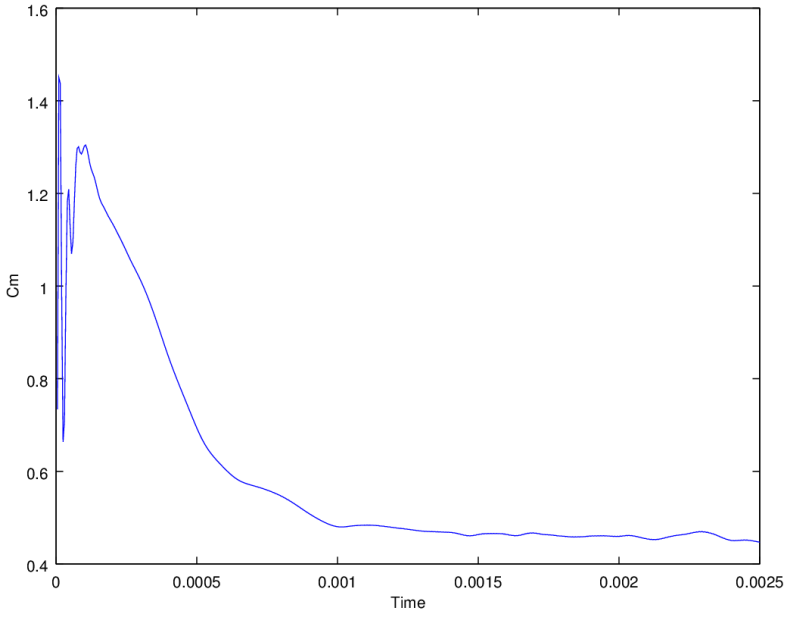


Figure B.38: Convergence of the RAS simulation at 30 degrees in a ROT boundary condition with the 3D mesh

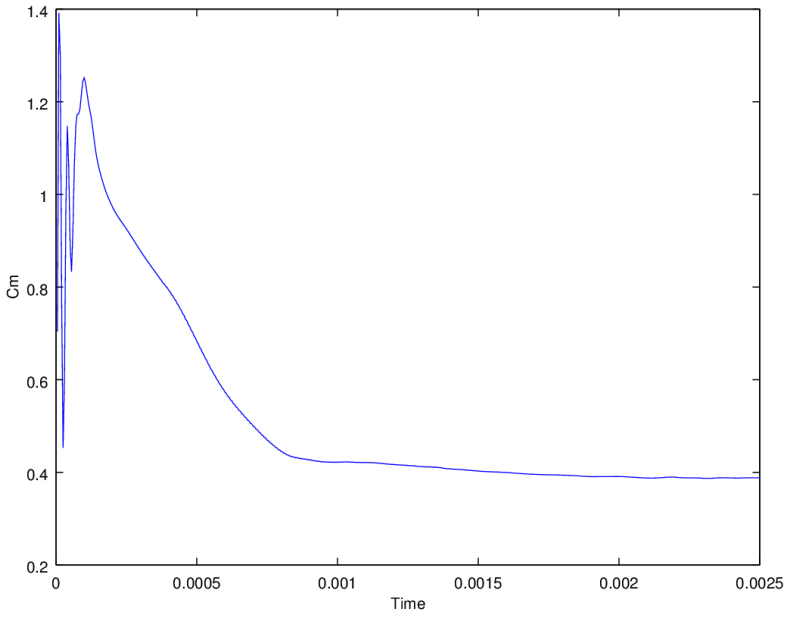


Figure B.39: Convergence of the RAS simulation at 60 degrees in a ROT boundary condition with the 3D mesh

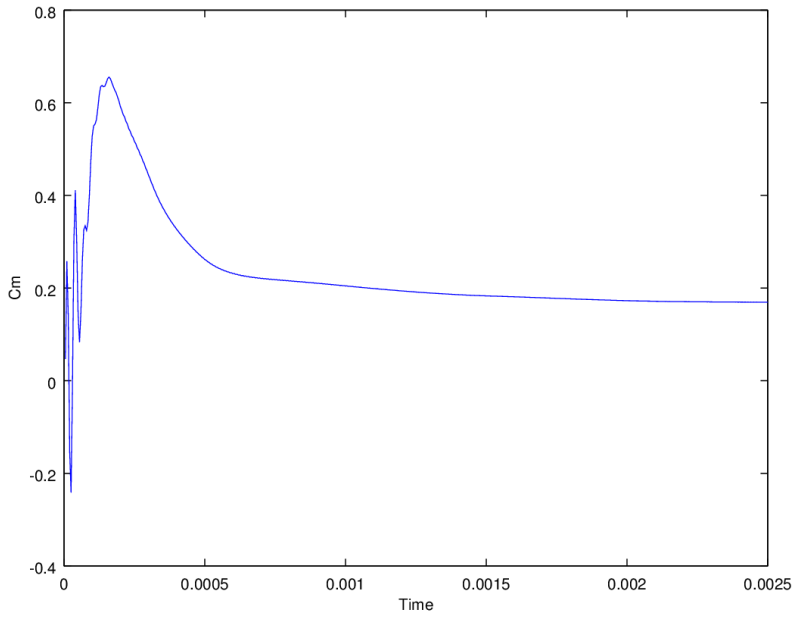


Figure B.40: Convergence of the RAS simulation at 90 degrees in a ROT boundary condition with the 3D mesh

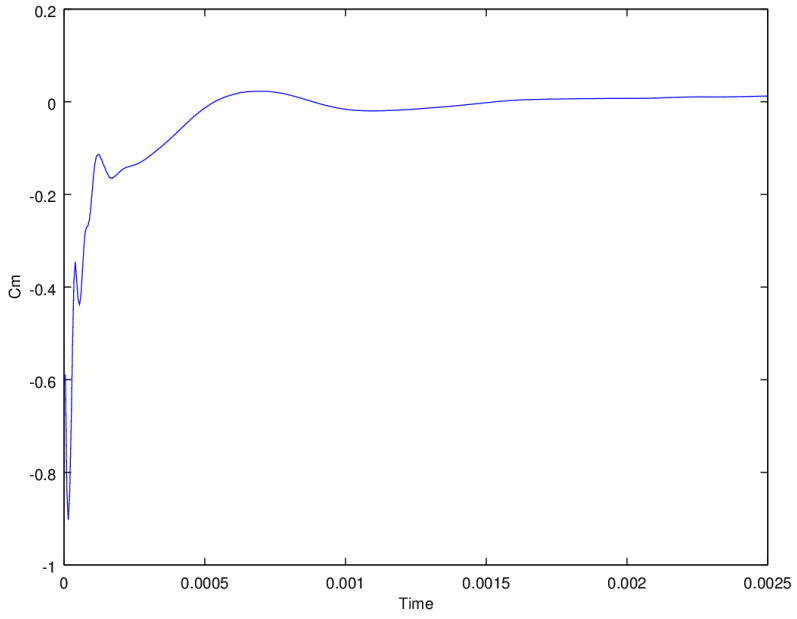


Figure B.41: Convergence of the RAS simulation at 120 degrees in a ROT boundary condition with the 3D mesh

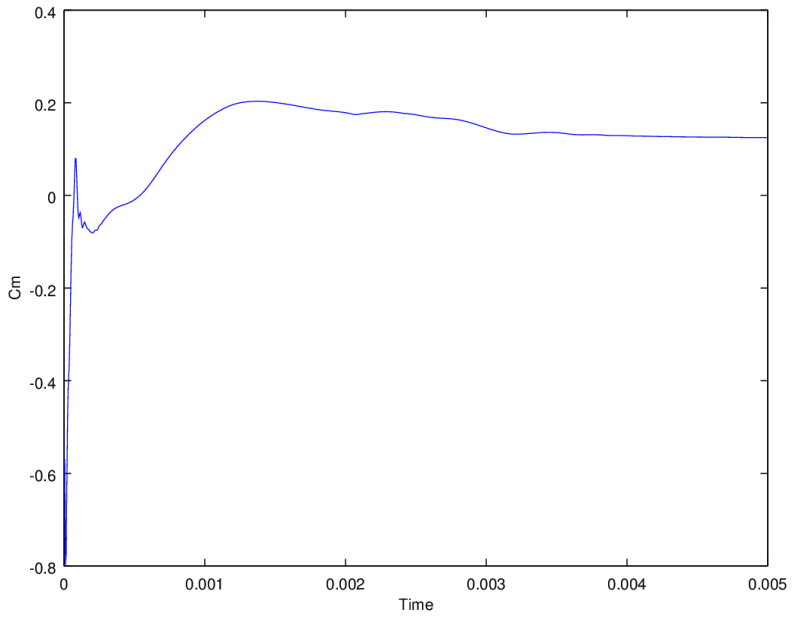


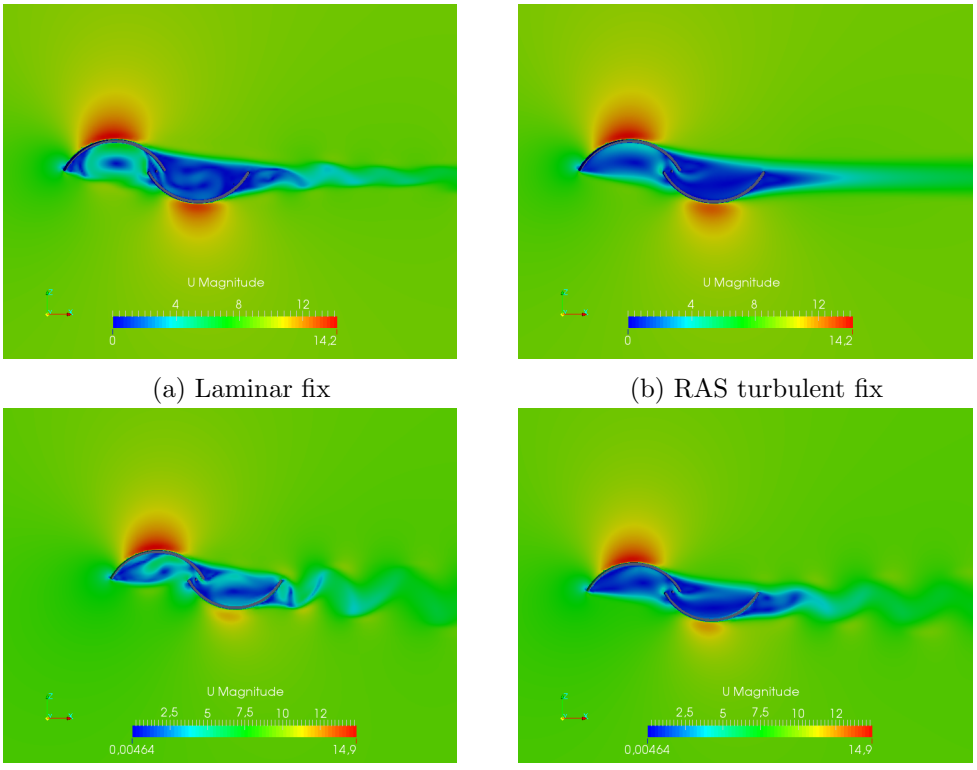
Figure B.42: Convergence of the RAS simulation at 150 degrees in a ROT boundary condition with the 3D mesh

Appendix C

Results

C.1 Cylinder

C.2 Savonius 2D



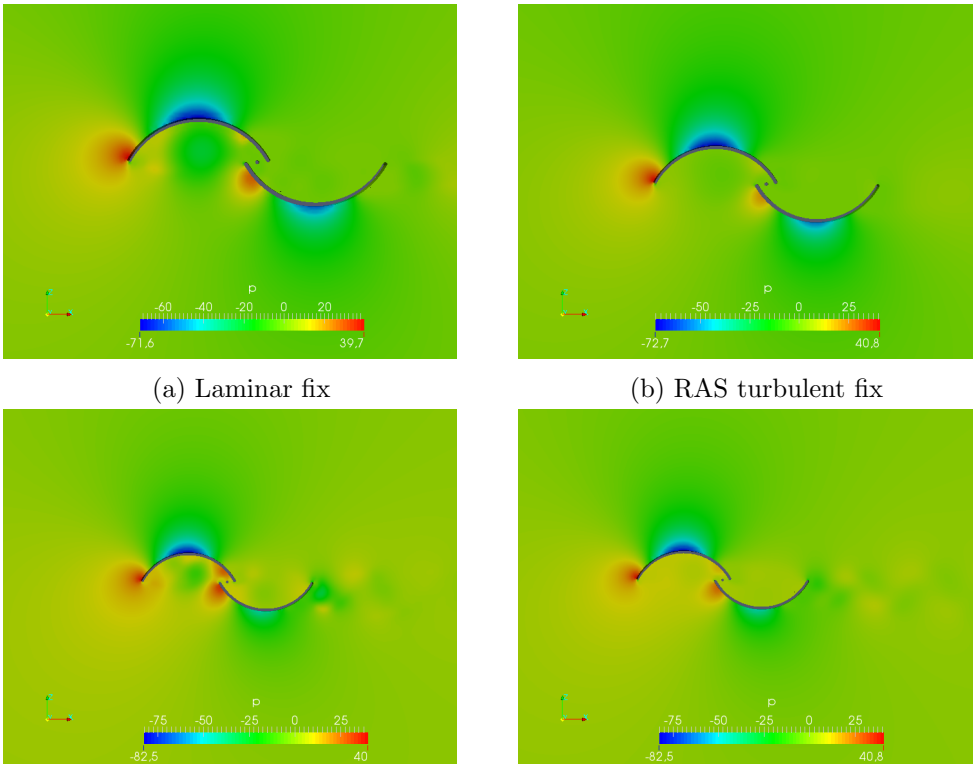
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

Figure C.1: Velocity fields at 0degrees with the 2D mesh



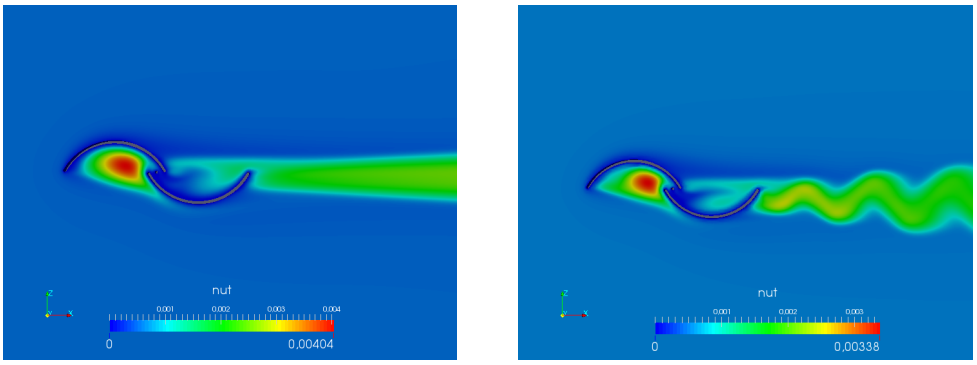
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

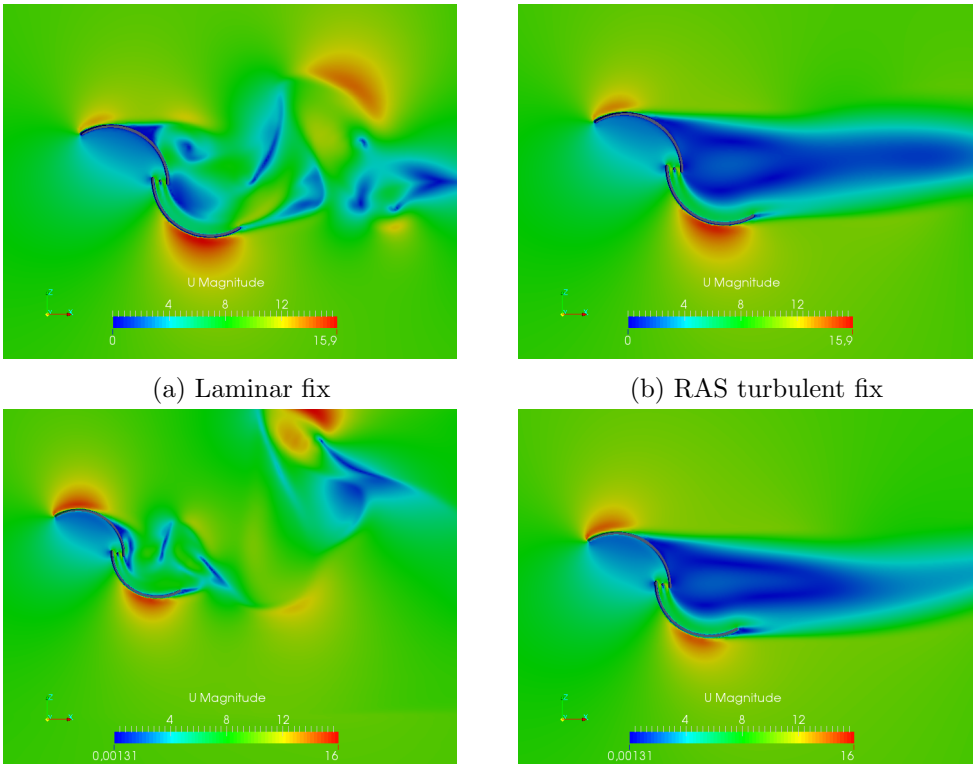
Figure C.2: Pressure fields at 0 degrees with the 2D mesh



(a) Fix

(b) Rotation

Figure C.3: Turbulent kinematic viscosity ν_t field at 0 degrees with the 2D mesh



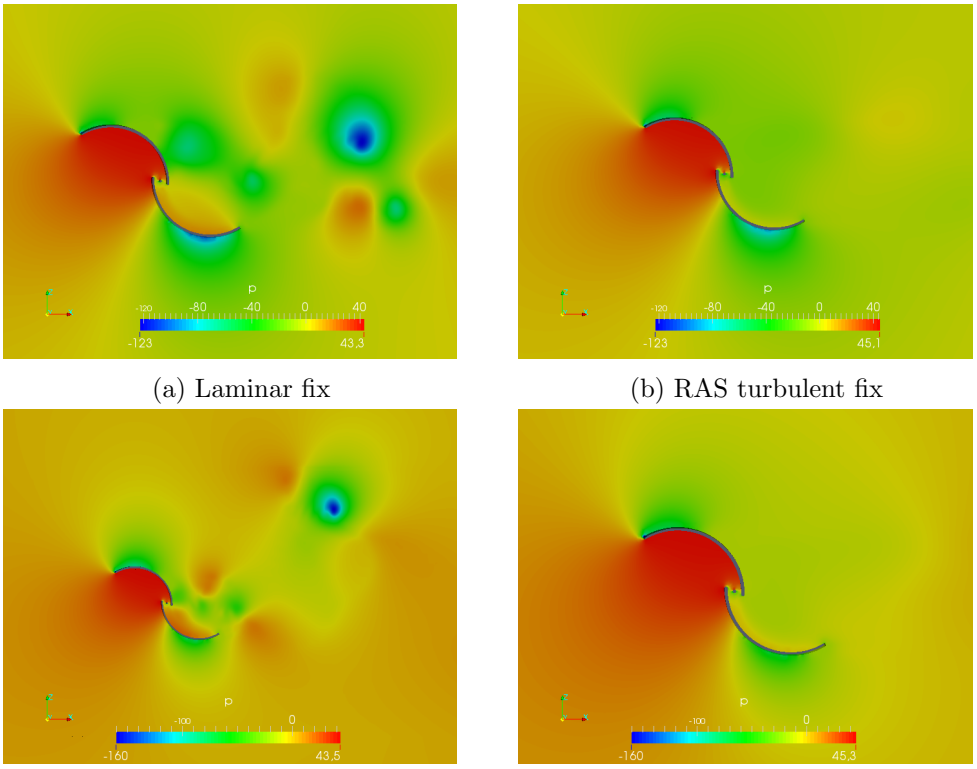
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

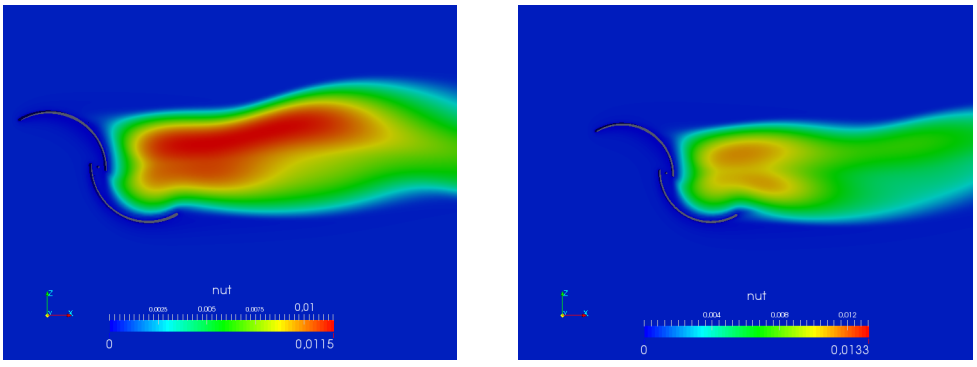
Figure C.4: Velocity fields at 30degrees with the 2D mesh



(a) Laminar fix (b) RAS turbulent fix

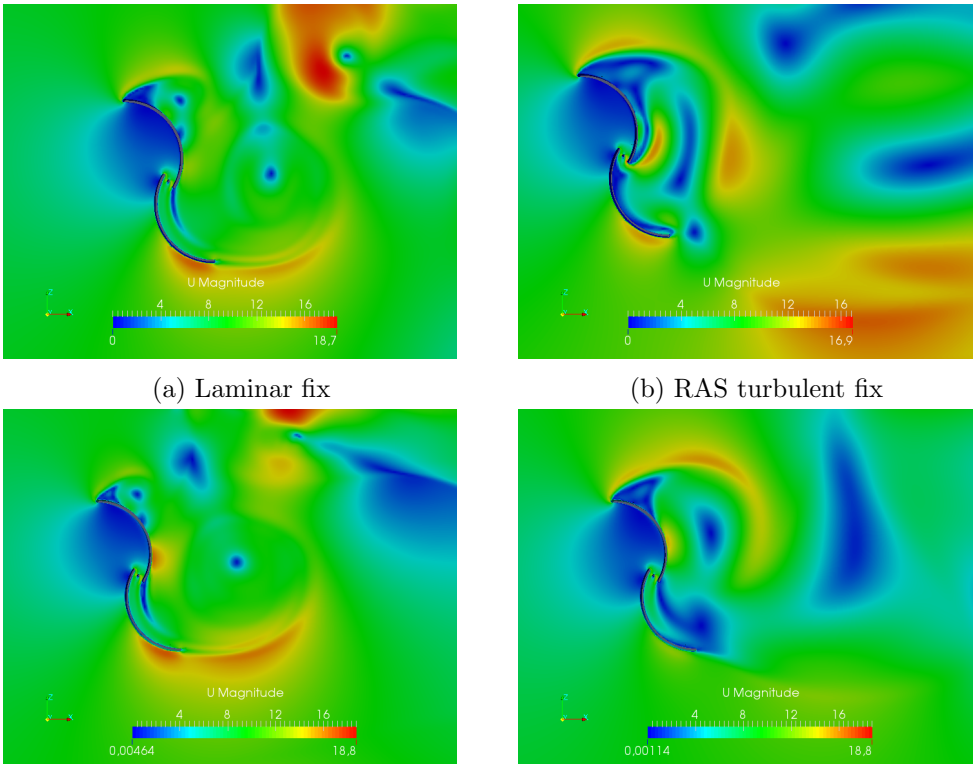
(c) Laminar rotation (d) RAS turbulent rotation

Figure C.5: Pressure fields at 30 degrees with the 2D mesh



(a) Fix (b) Rotation

Figure C.6: Turbulent kinematic viscosity ν_t field at 30 degrees with the 2D mesh



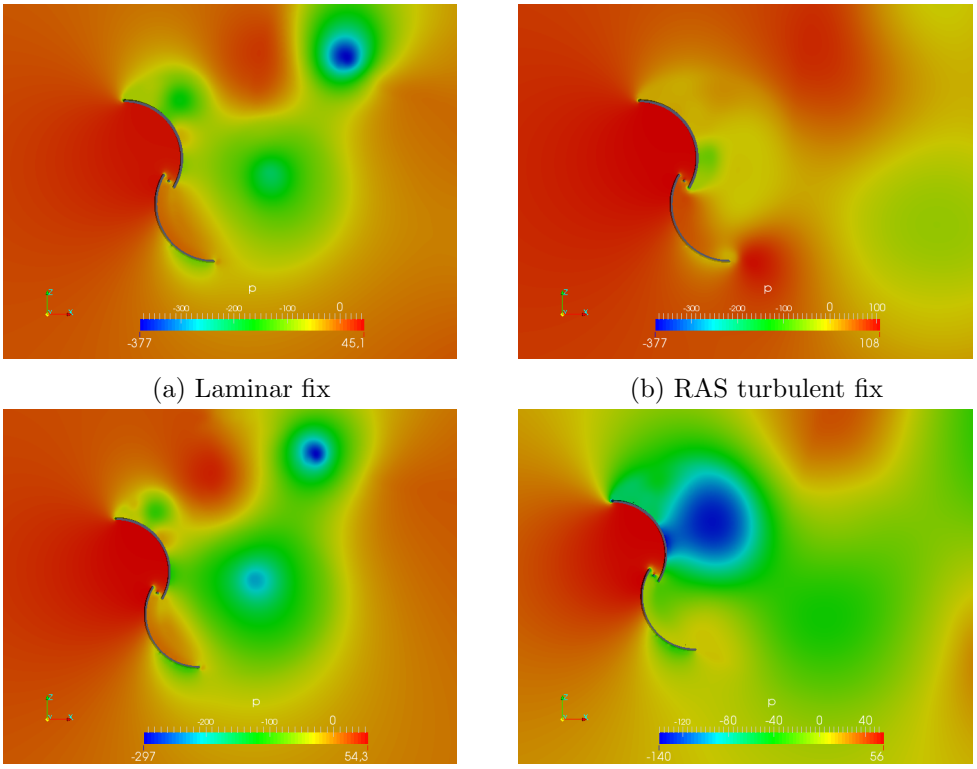
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

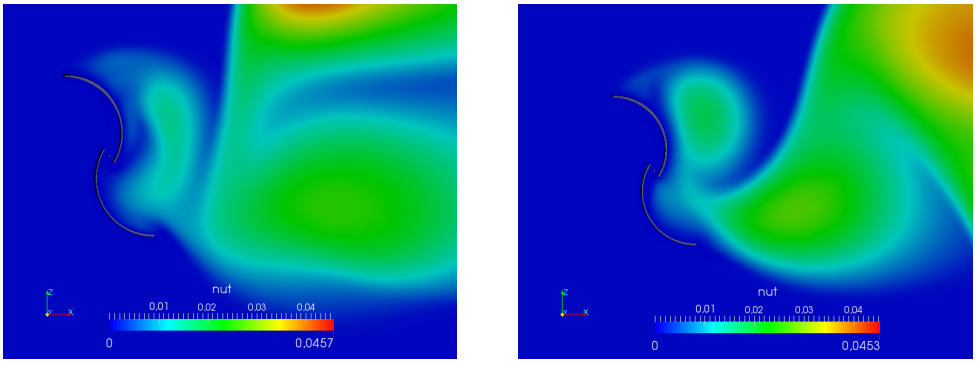
Figure C.7: Velocity fields at 60degrees with the 2D mesh



(a) Laminar fix (b) RAS turbulent fix

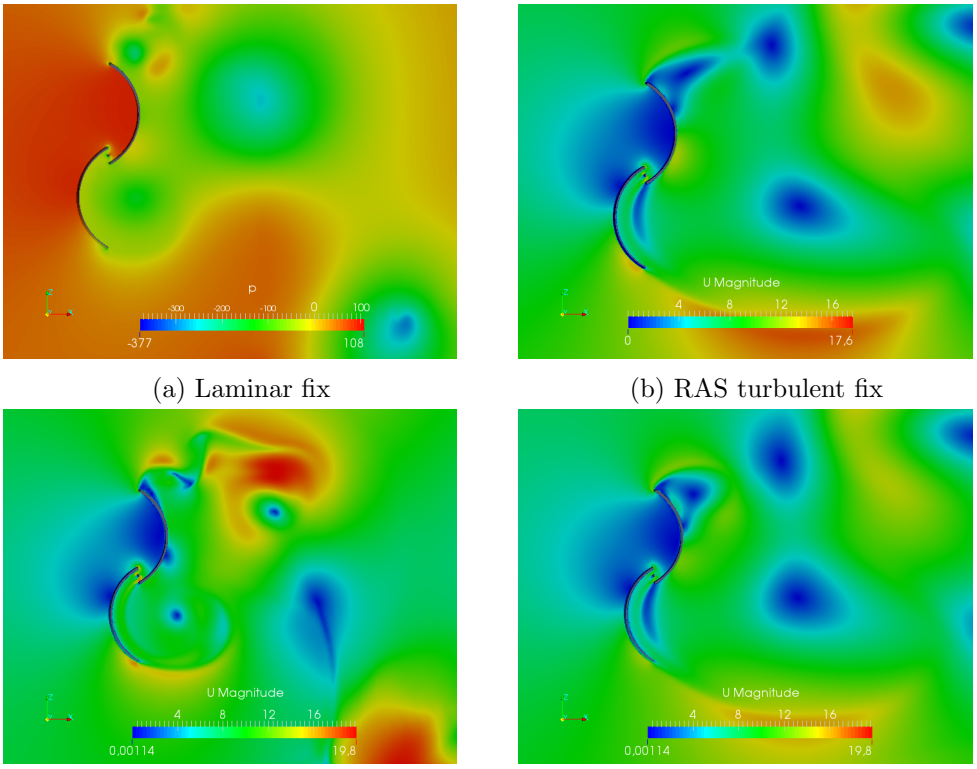
(c) Laminar rotation (d) RAS turbulent rotation

Figure C.8: Pressure fields at 60 degrees with the 2D mesh



(a) Fix (b) Rotation

Figure C.9: Turbulent kinematic viscosity ν_t field at 60 degrees with the 2D mesh



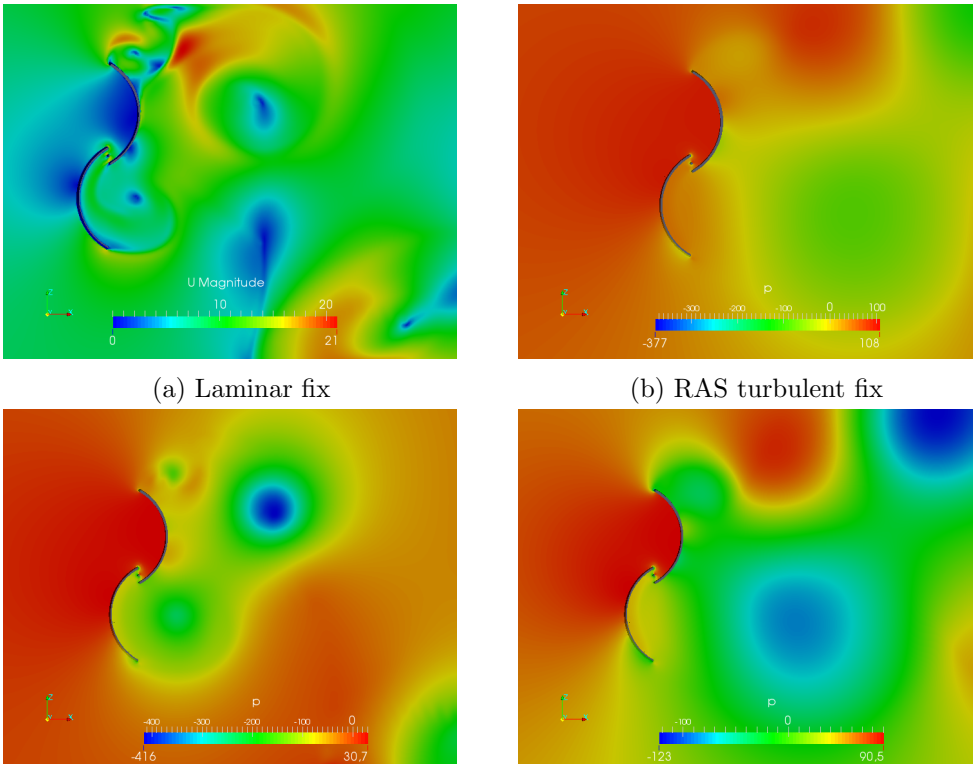
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

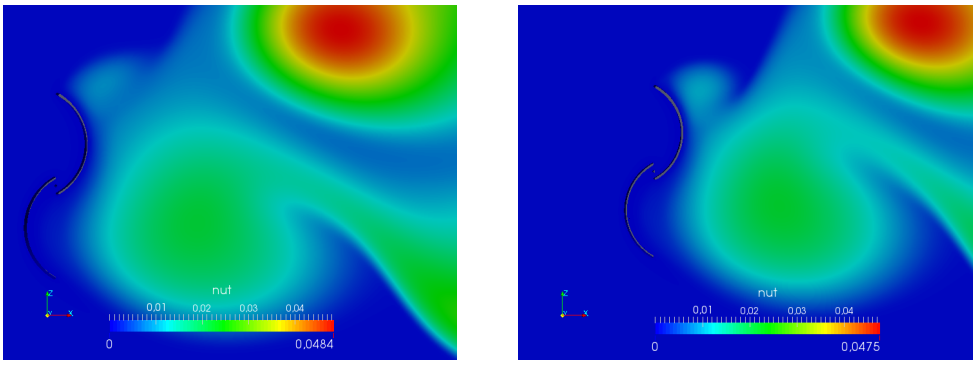
Figure C.10: Velocity fields at 90degrees with the 2D mesh



(a) Laminar fix (b) RAS turbulent fix

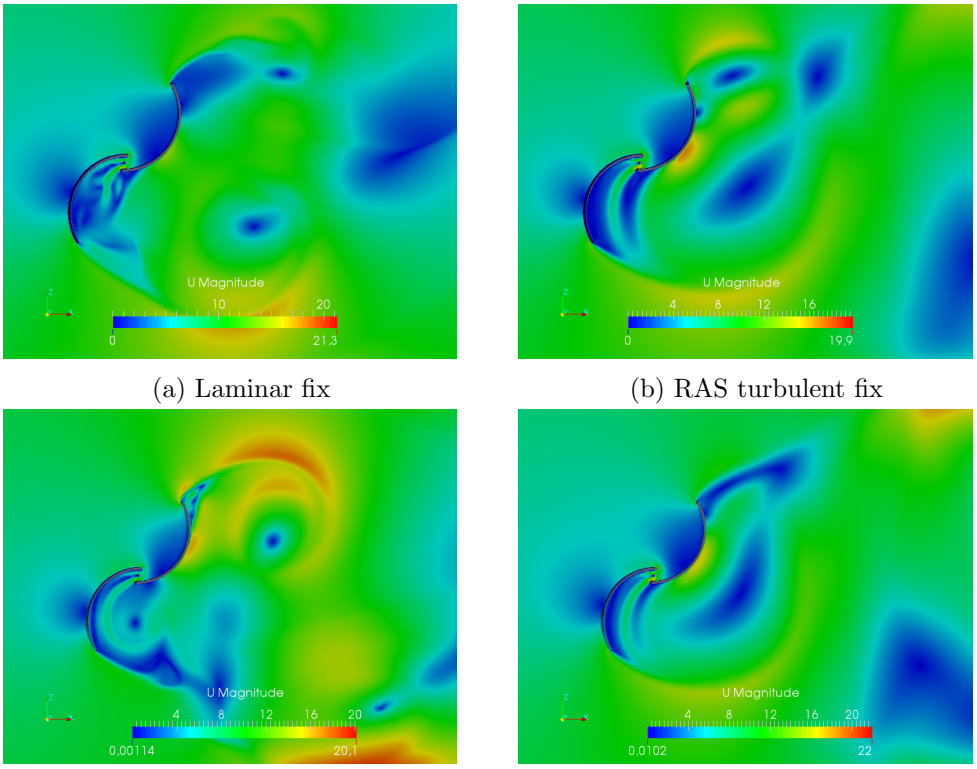
(c) Laminar rotation (d) RAS turbulent rotation

Figure C.11: Pressure fields at 90 degrees with the 2D mesh



(a) Fix (b) Rotation

Figure C.12: Turbulent kinematic viscosity ν_t field at 90 degrees with the 2D mesh



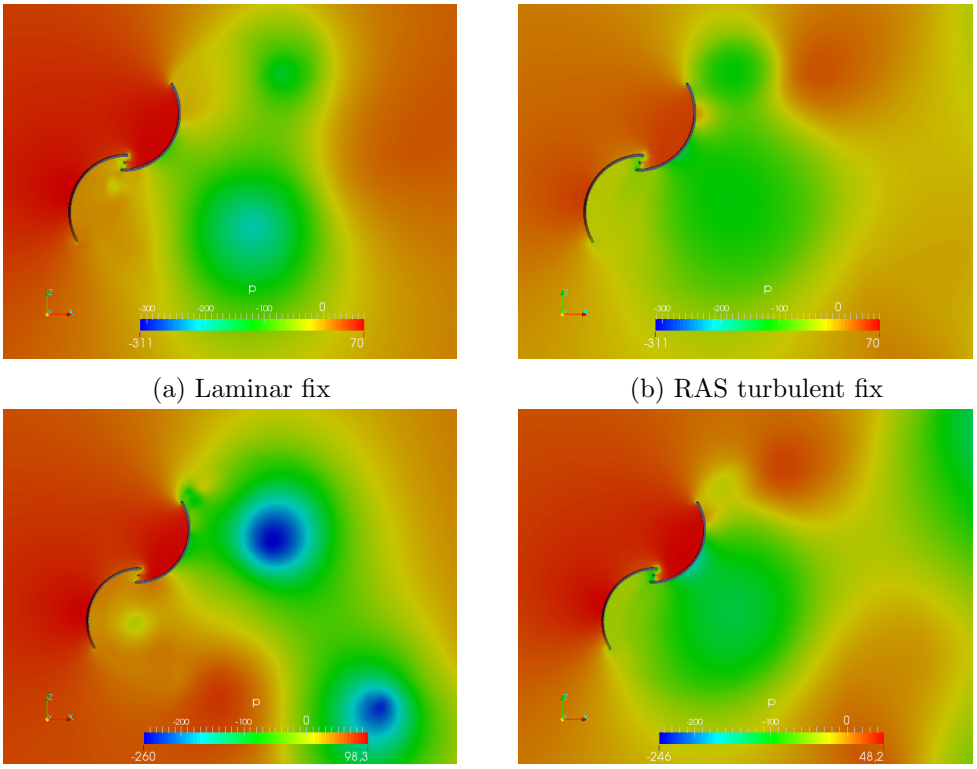
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

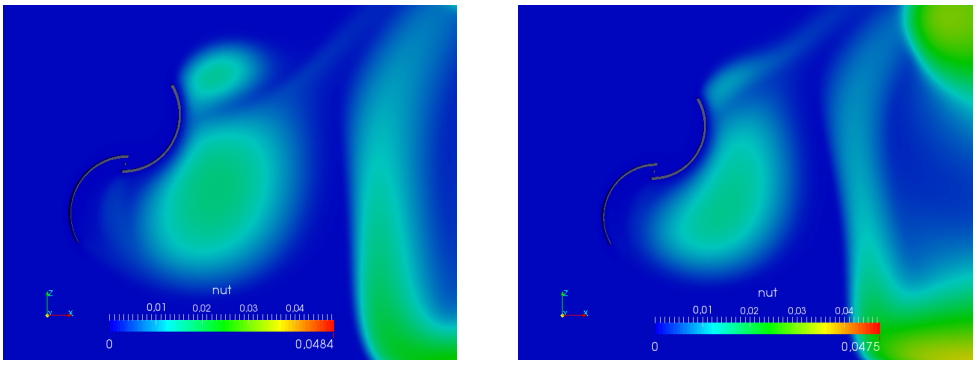
Figure C.13: Velocity fields at 120degrees with the 2D mesh



(a) Laminar fix (b) RAS turbulent fix

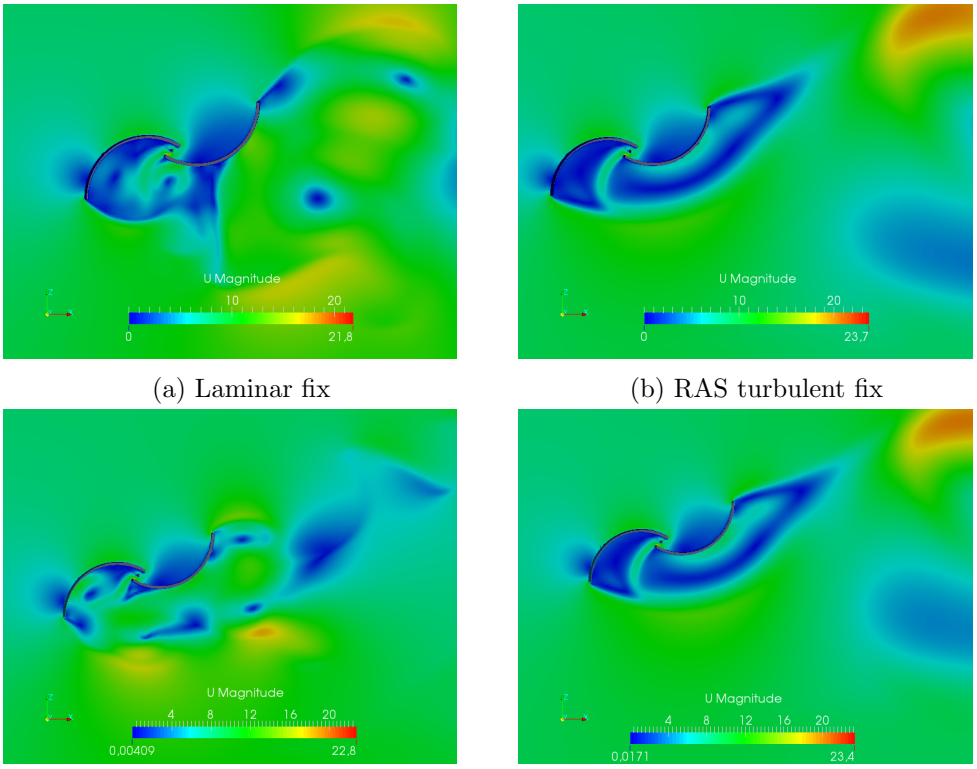
(c) Laminar rotation (d) RAS turbulent rotation

Figure C.14: Pressure fields at 120 degrees with the 2D mesh



(a) Fix (b) Rotation

Figure C.15: Turbulent kinematic viscosity ν_t field at 120 degrees with the 2D mesh



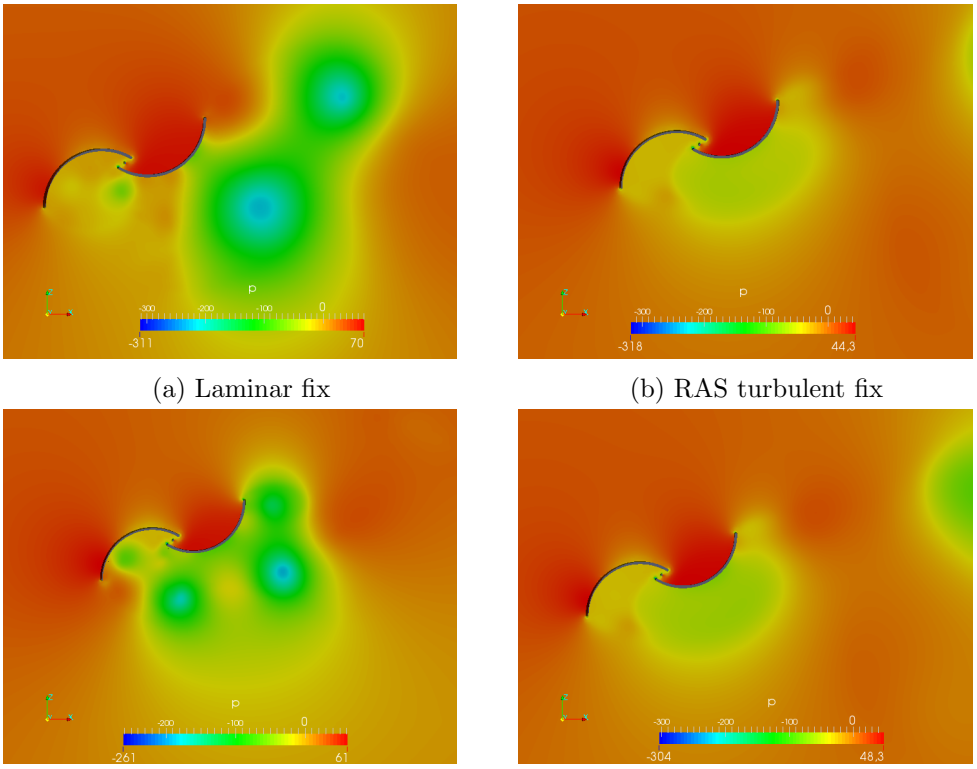
(a) Laminar fix

(b) RAS turbulent fix

(c) Laminar rotation

(d) RAS turbulent rotation

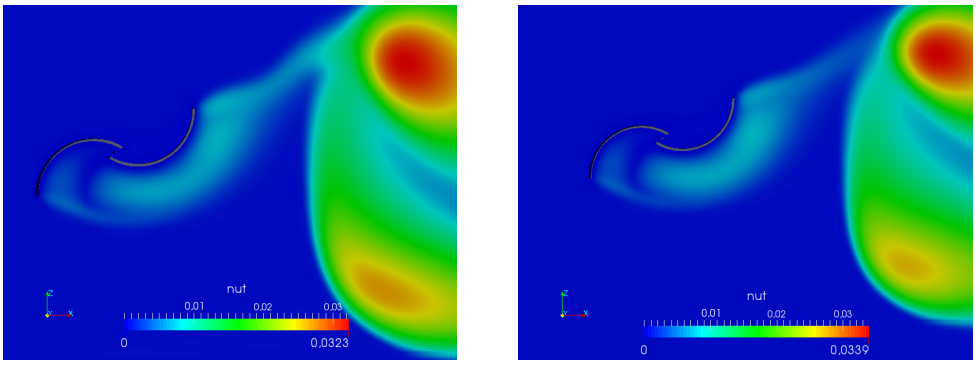
Figure C.16: Velocity fields at 150degrees with the 2D mesh



(a) Laminar fix (b) RAS turbulent fix

(c) Laminar rotation (d) RAS turbulent rotation

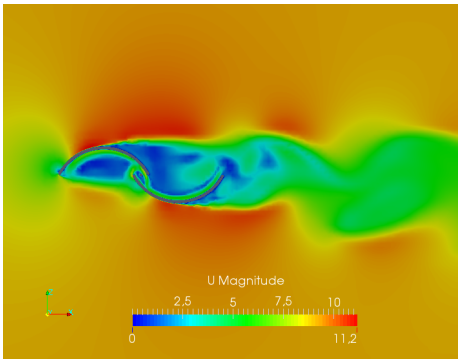
Figure C.17: Pressure fields at 150 degrees with the 2D mesh



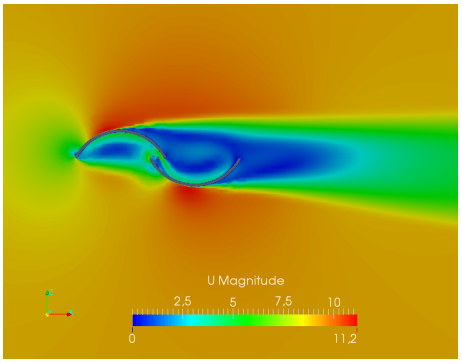
(a) Fix (b) Rotation

Figure C.18: Turbulent kinematic viscosity ν_t field at 150 degrees with the 2D mesh

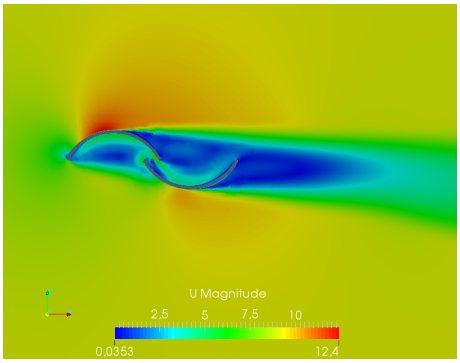
C.3 Savonius 3D



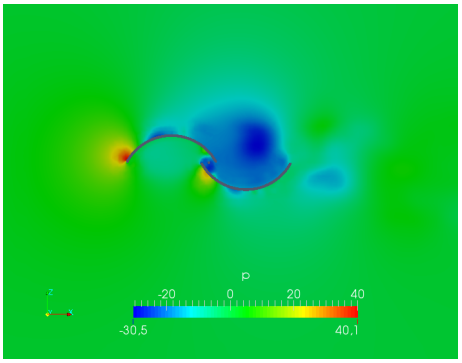
(a) Velocity field laminar fix



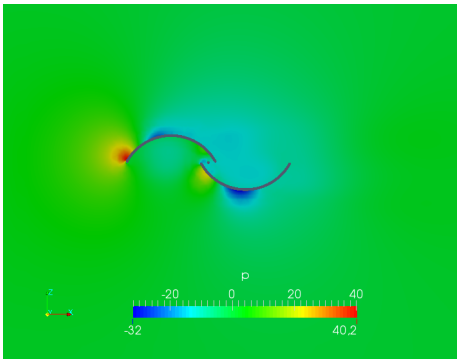
(b) Velocity field RAS turbulent fix



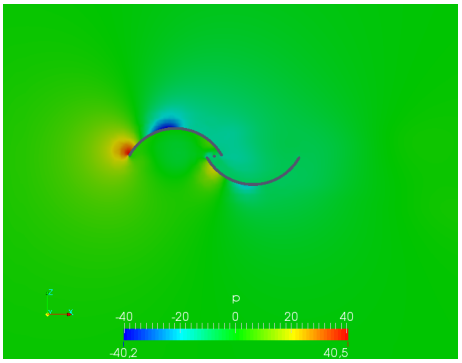
(c) Velocity field RAS turbulent rotation



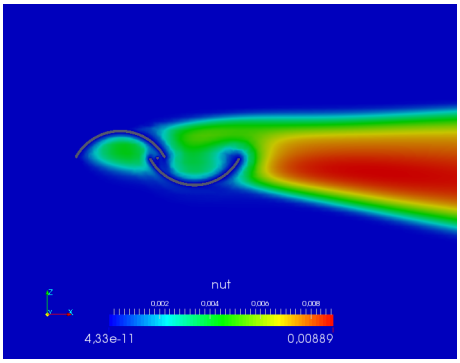
(d) Pressure field Laminar fix



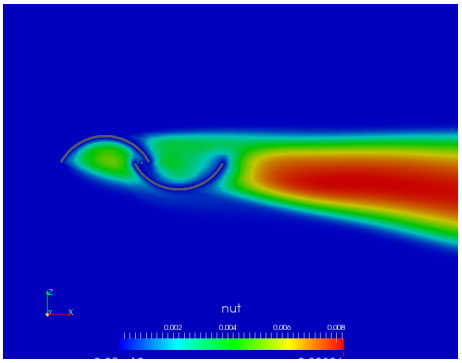
(e) Pressure field RAS turbulent fix



(f) Pressure field RAS turbulent rotation

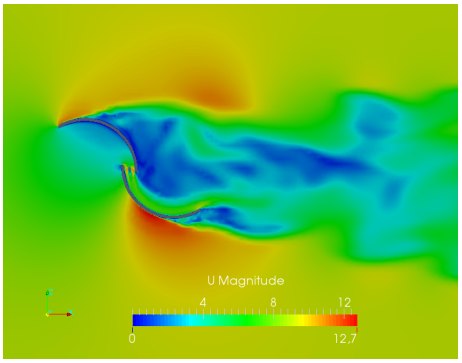


(g) Turbulent kinematic viscosity without rotation

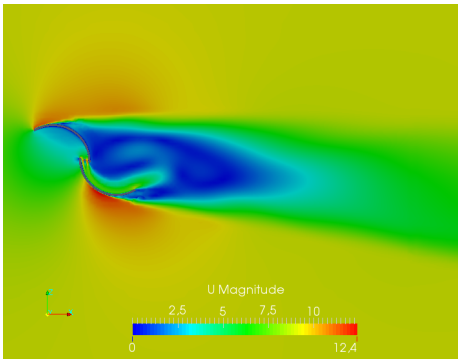


(h) Turbulent kinematic viscosity in rotation

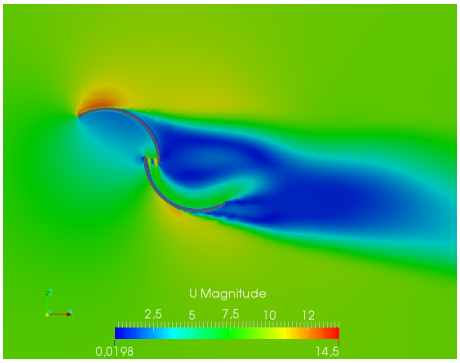
Figure C.19: Fields at 0 degrees with the 3D mesh



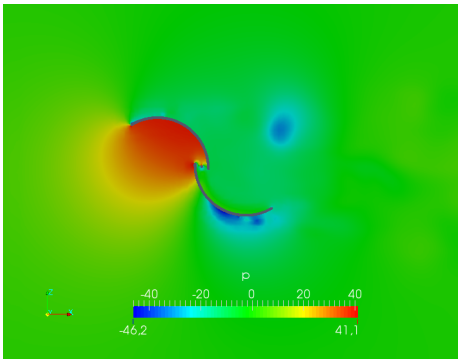
(a) Velocity field laminar fix



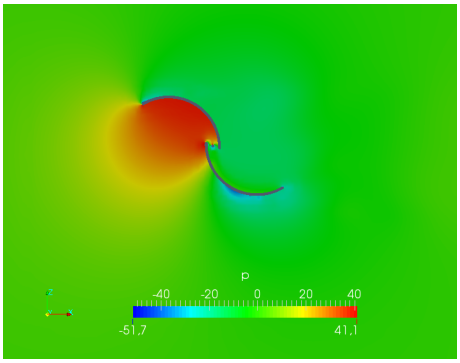
(b) Velocity field RAS turbulent fix



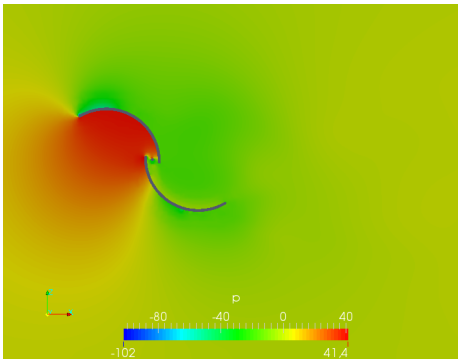
(c) Velocity field RAS turbulent rotation



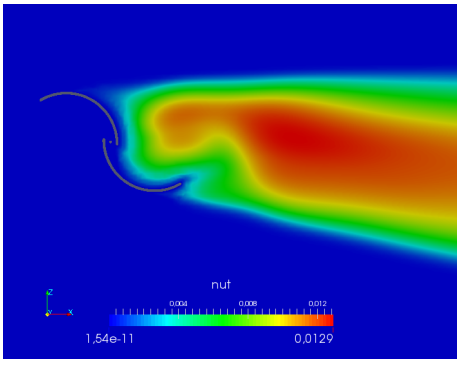
(d) Pressure field Laminar fix



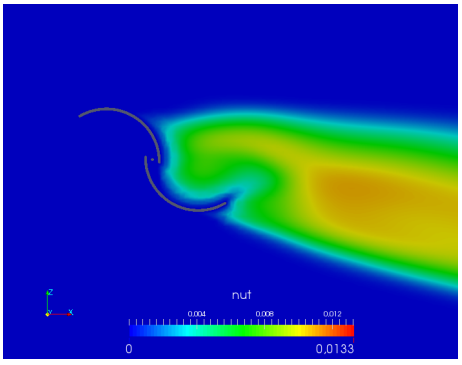
(e) Pressure field RAS turbulent fix



(f) Pressure field RAS turbulent rotation

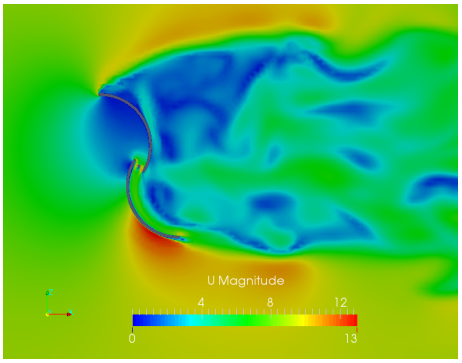


(g) Turbulent kinematic viscosity without rotation

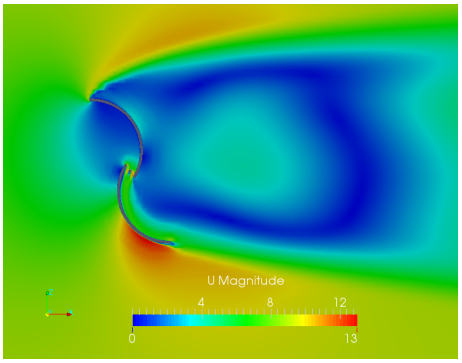


(h) Turbulent kinematic viscosity in rotation

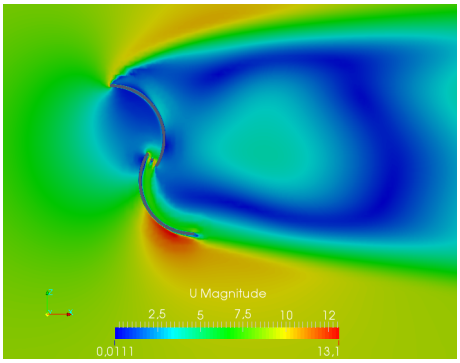
Figure C.20: Fields at 30 degrees with the 3D mesh



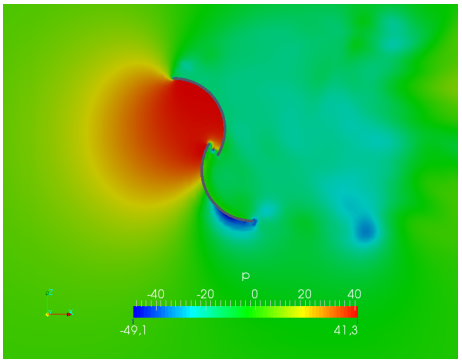
(a) Velocity field laminar fix



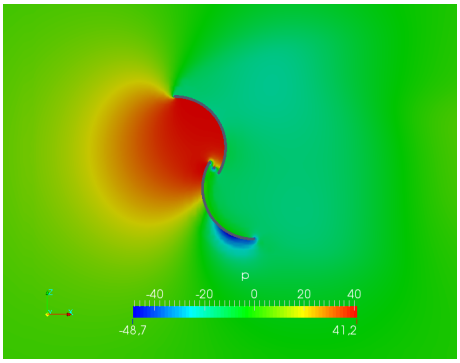
(b) Velocity field RAS turbulent fix



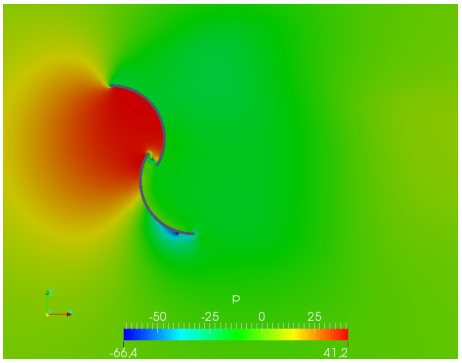
(c) Velocity field RAS turbulent rotation



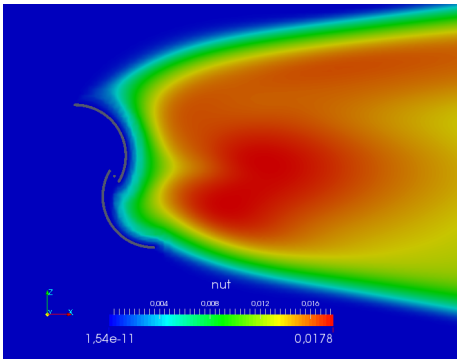
(d) Pressure field Laminar fix



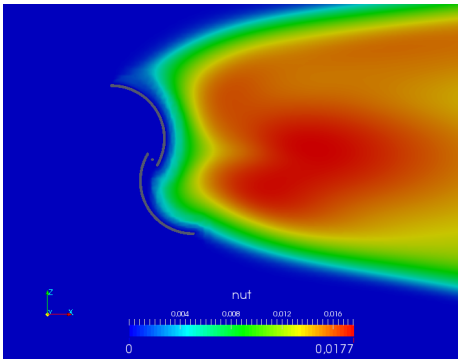
(e) Pressure field RAS turbulent fix



(f) Pressure field RAS turbulent rotation

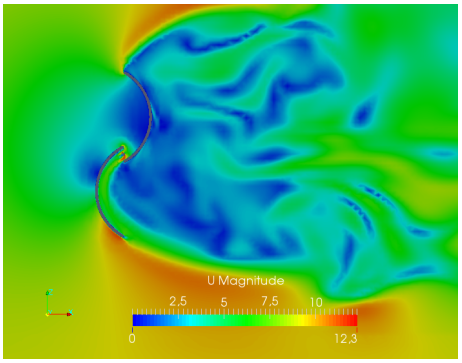


(g) Turbulent kinematic viscosity without rotation

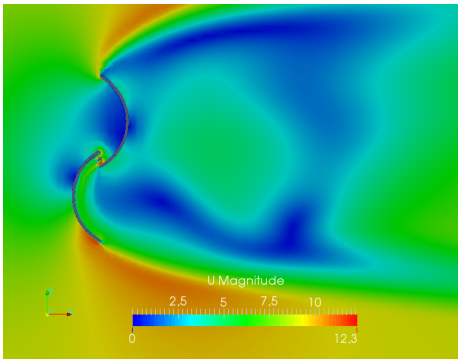


(h) Turbulent kinematic viscosity in rotation

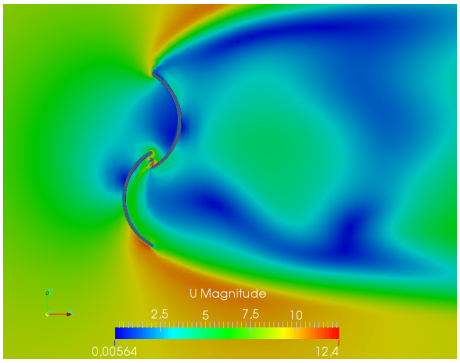
Figure C.21: Fields at 60 degrees with the 3D mesh



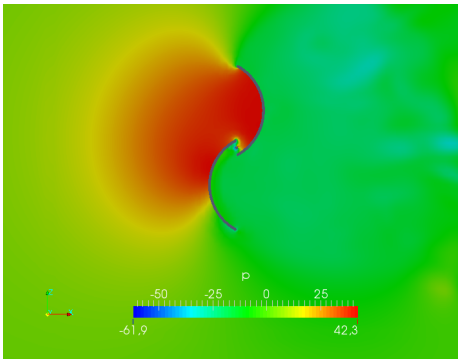
(a) Velocity field laminar fix



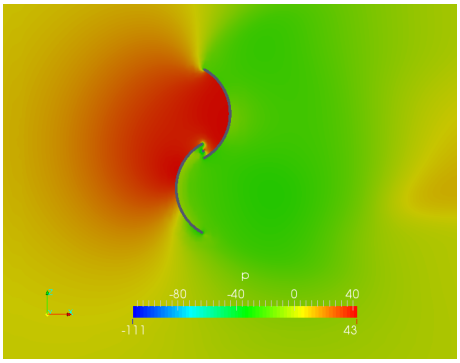
(b) Velocity field RAS turbulent fix



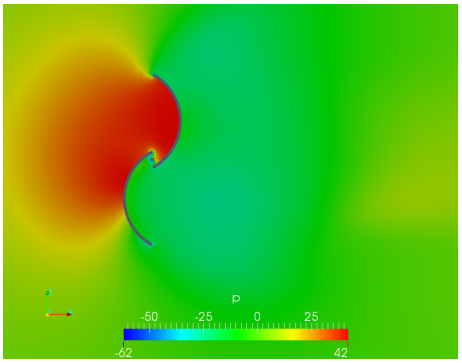
(c) Velocity field RAS turbulent rotation



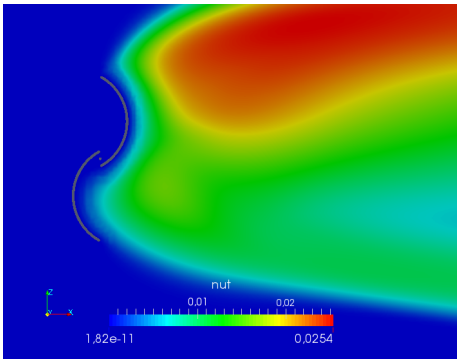
(d) Pressure field Laminar fix



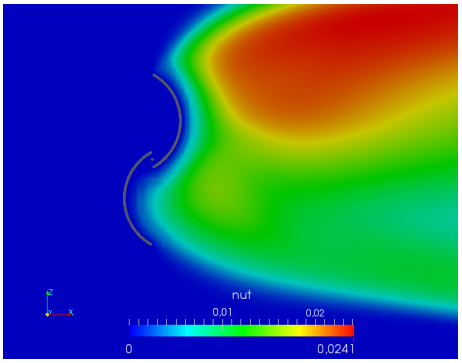
(e) Pressure field RAS turbulent fix



(f) Pressure field RAS turbulent rotation

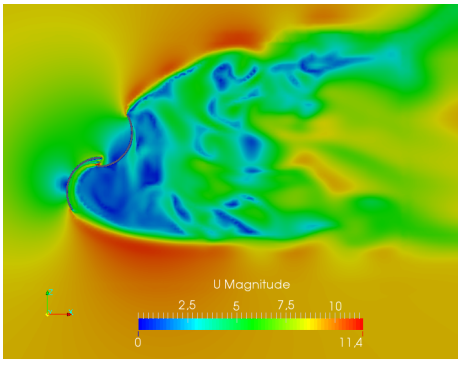


(g) Turbulent kinematic viscosity without rotation

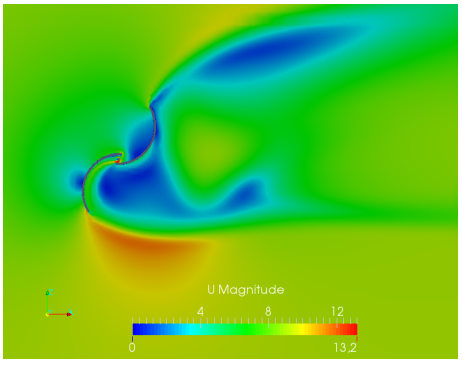


(h) Turbulent kinematic viscosity in rotation

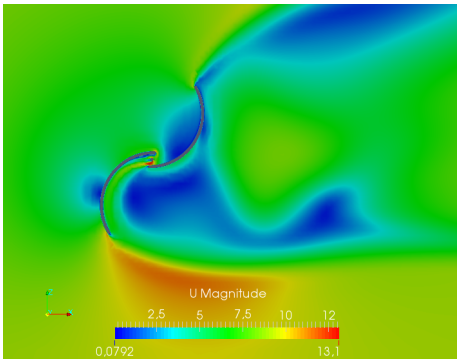
Figure C.22: Fields at 90 degrees with the 3D mesh



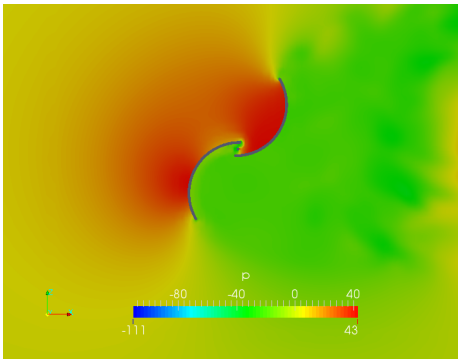
(a) Velocity field laminar fix



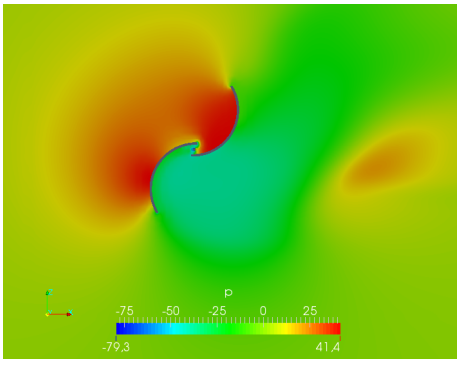
(b) Velocity field RAS turbulent fix



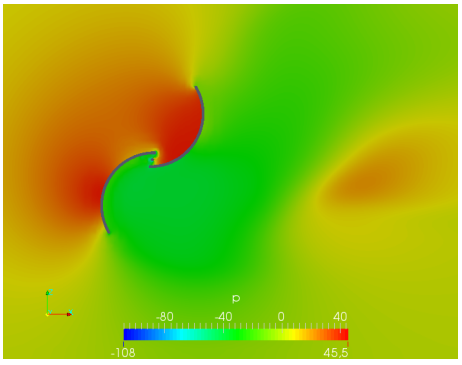
(c) Velocity field RAS turbulent rotation



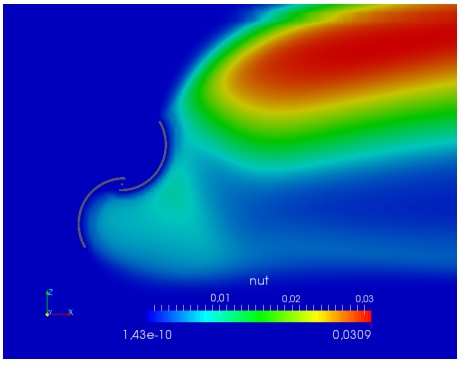
(d) Pressure field Laminar fix



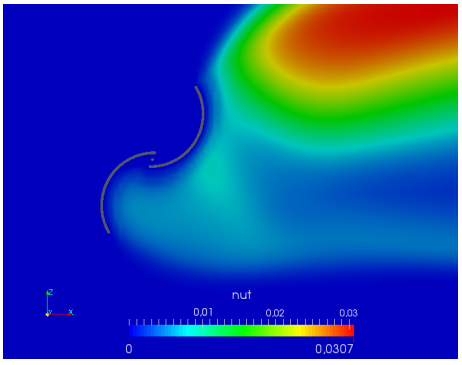
(e) Pressure field RAS turbulent fix



(f) Pressure field RAS turbulent rotation

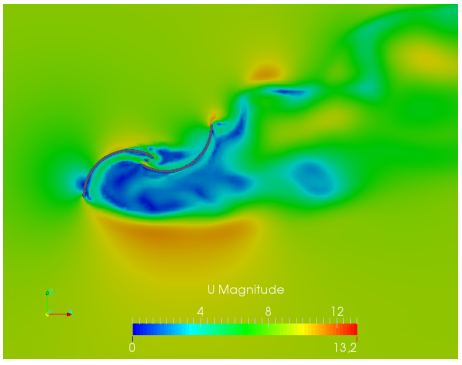


(g) Turbulent kinematic viscosity without rotation

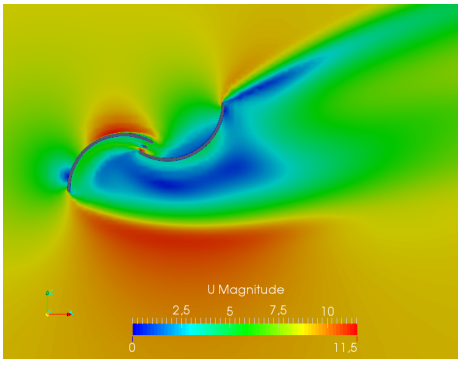


(h) Turbulent kinematic viscosity in rotation

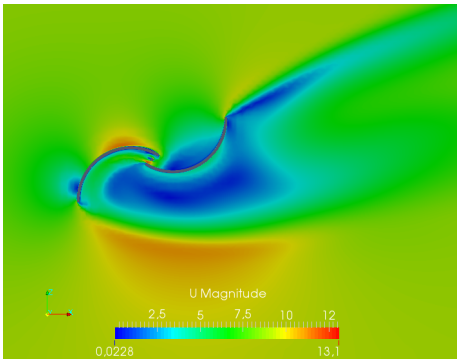
Figure C.23: Fields at 120 degrees with the 3D mesh



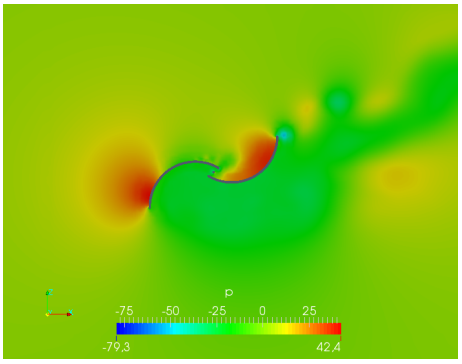
(a) Velocity field laminar fix



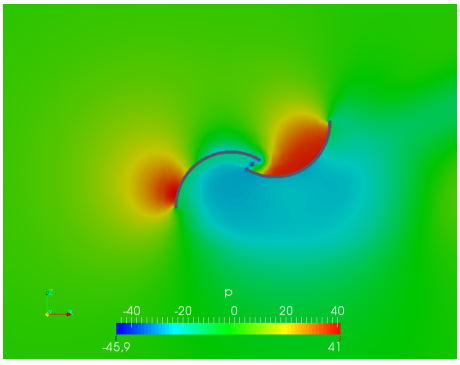
(b) Velocity field RAS turbulent fix



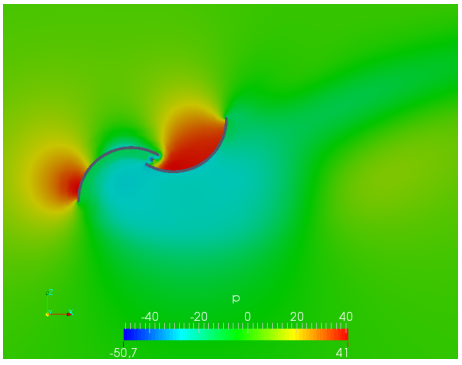
(c) Velocity field RAS turbulent rotation



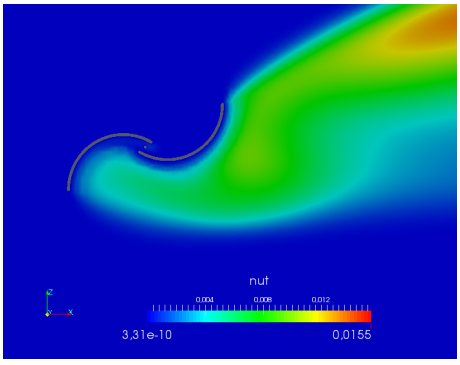
(d) Pressure field Laminar fix



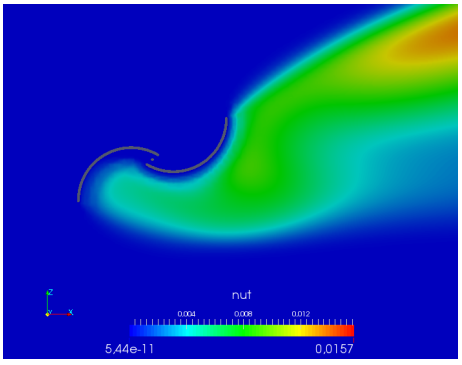
(e) Pressure field RAS turbulent fix



(f) Pressure field RAS turbulent rotation



(g) Turbulent kinematic viscosity without rotation



(h) Turbulent kinematic viscosity in rotation

Figure C.24: Fields at 150 degrees with the 3D mesh