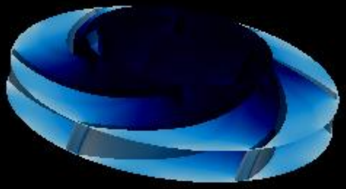
Lilian Chabannes
Jan Zbavitel

February 2018

Radial Pump with OpenFOAM : A case of a stady state, periodic impeller, bad mesh pump simulation with OpenFOAM5 and foam-extend 4.0



Download the case :
https://github.com/Lookid-OF/RadialPump
If you see mistakes, improvements or anything, please send a mail :
Chabannes-lilian@hotmail.fr
Be sure to join the OpenFOAM Discord !
https://discord.gg/P9p9eHn

Sources

- József Nagy’s YT channel
https://www.youtube.com/channel/UCjdgpuxuAxH9BqheyE82Vvw
- Tobias Holzmann YT channel and website
https://www.youtube.com/user/HolzmannCFD

https://holzmann-cfd.de/
- Damogran Pump

https://damogranlabs.com/2017/10/openfoam-a-case-of-a-bad-pump/
- OF wiki for turbomachinery

http://openfoamwiki.net/index.php/Sig_Turbomachinery_/_ERCOFTAC_centrifugal_pump_with_a_vaned_diffuser

Sources

- OpenFOAM 5

https://openfoam.org/version/5-0/

- foam-extend 4.0

http://openfoamwiki.net/index.php/Installation/Linux/foam-extend-4.0

- Install cfMesh with OpenFOAM 5

```
$ git clone https://git.code.sf.net/p/cfmesh/code cfmesh-code -b port-v1606+
$ cd cfmesh-code
```

Open cfmesh-code/meshLibrary/Make/options and change

```
CFMESH_MACROS = -DNoSizeType
```

into

```
CFMESH_MACROS = -DNoSizeType -DOpenCFDSpecific
```

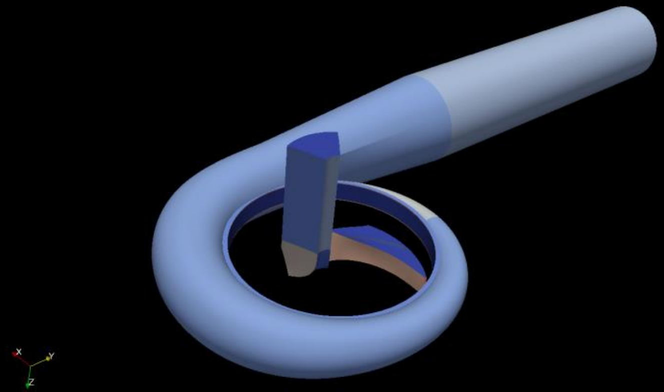
```
$ git checkout development
$ export WM_NCOMPPROCS=4
$ ./Allwmake
```

Pump specification

- $Q = 30\ l/s$
- $H = 32{,}5\ m$
- $N = 2900\ rpm$
- $D_2 = 174\ mm$

.stp to .stl patches

SALOME is used to create the different .stl files necessary for creating the mesh with cfMesh. The input file is a .stp file.



See Tobias Holzmann videos to obtain proper .stl files in SALOME
https://www.youtube.com/watch?v=NBmB6xsnznw&list=PLZDUQMOoipL7RenhfNiGLaiE2oL3CN4WA&index=7

CONTENTS

cfMesh is used to mesh 3 parts

- Inlet tube
- Impeller
- Volute + outlet tube

The outlet tube is necessary to avoid recirculation at the outlet, but won't be used for the head calculation.

patches

Many patches have been created in order to specify different mesh parameters. Moreover, sharp angles may not be well defined if they are not at the intersection of two patches.

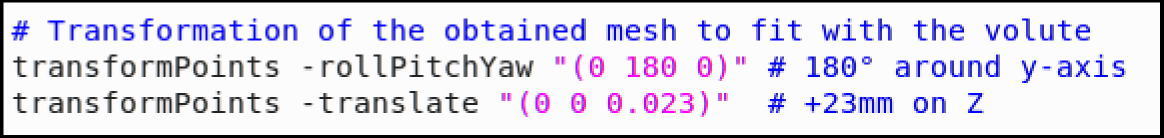
The impeller mesh has a high non-orthogonality (~78) that I couldn't get rid of, thus I decided to create a whole coarse mesh, the point being to make the simulation run.

In order to rotate, OpenFOAM asks for a cellZone, thus topoSet is applied to all the part in order to have those cellZones defined, eventhough the only necessary one is the impeller one.

CONTENTS

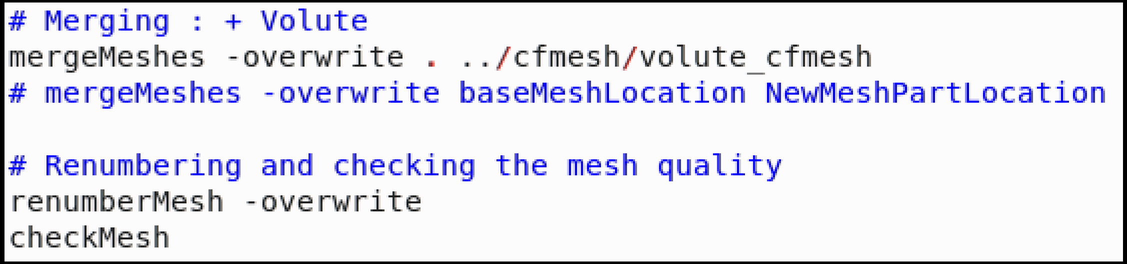
Merging the meshes

First, the inlet mesh is copied into the simulation directory, then the impeller mesh is merged to it. As your saw before, the impeller and the volute are not aligned, we need to modify the location of the mesh.



```
# Transformation of the obtained mesh to fit with the volute
transformPoints -rollPitchYaw "(0 180 0)" # 180° around y-axis
transformPoints -translate "(0 0 0.023)"  # +23mm on Z
```

The volute is then merged, and the mesh renumbered in order to speed up the simulation.

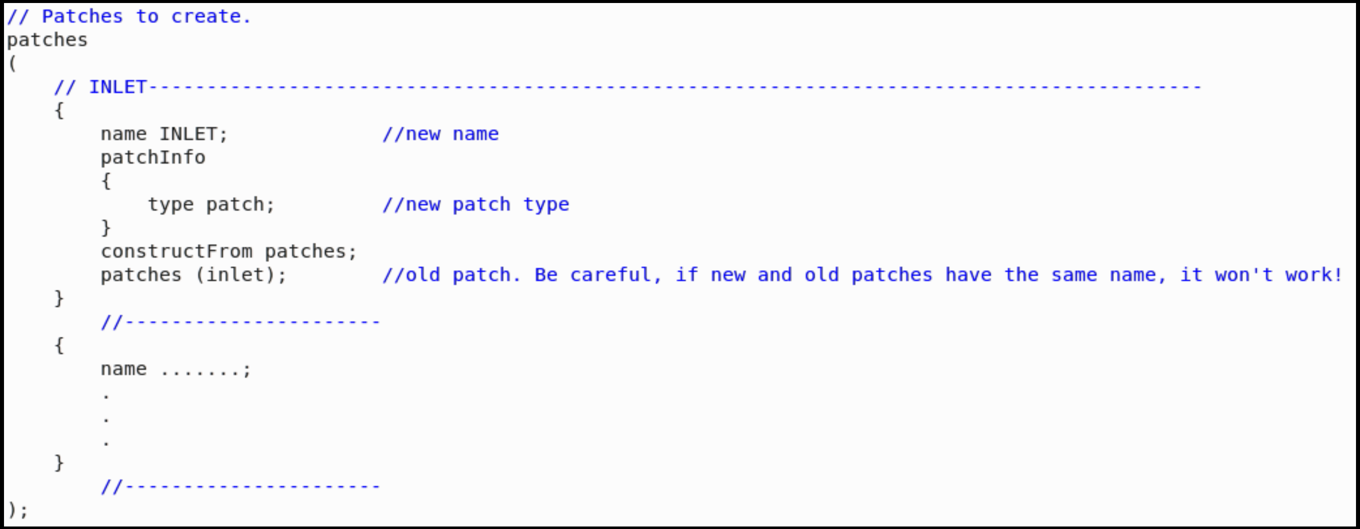


```
# Merging : + Volute
mergeMeshes -overwrite . ../cfmesh/volute_cfmesh
# mergeMeshes -overwrite baseMeshLocation NewMeshPartLocation

# Renumbering and checking the mesh quality
renumberMesh -overwrite
checkMesh
```

Create the patches

The constant/polyMesh/boundary file needs to be defined properly as now all the patches are walls. You can do this by hand or use the application createPatch controled by a … createPatchDict (surpriiiiiiise). The patches types will be patch for the inlet and the outlet, ggi for the connection between inlet/impeller and impeller/volute and cyclicGgi for all periodic patches. Here is how createPatchDict should look like :

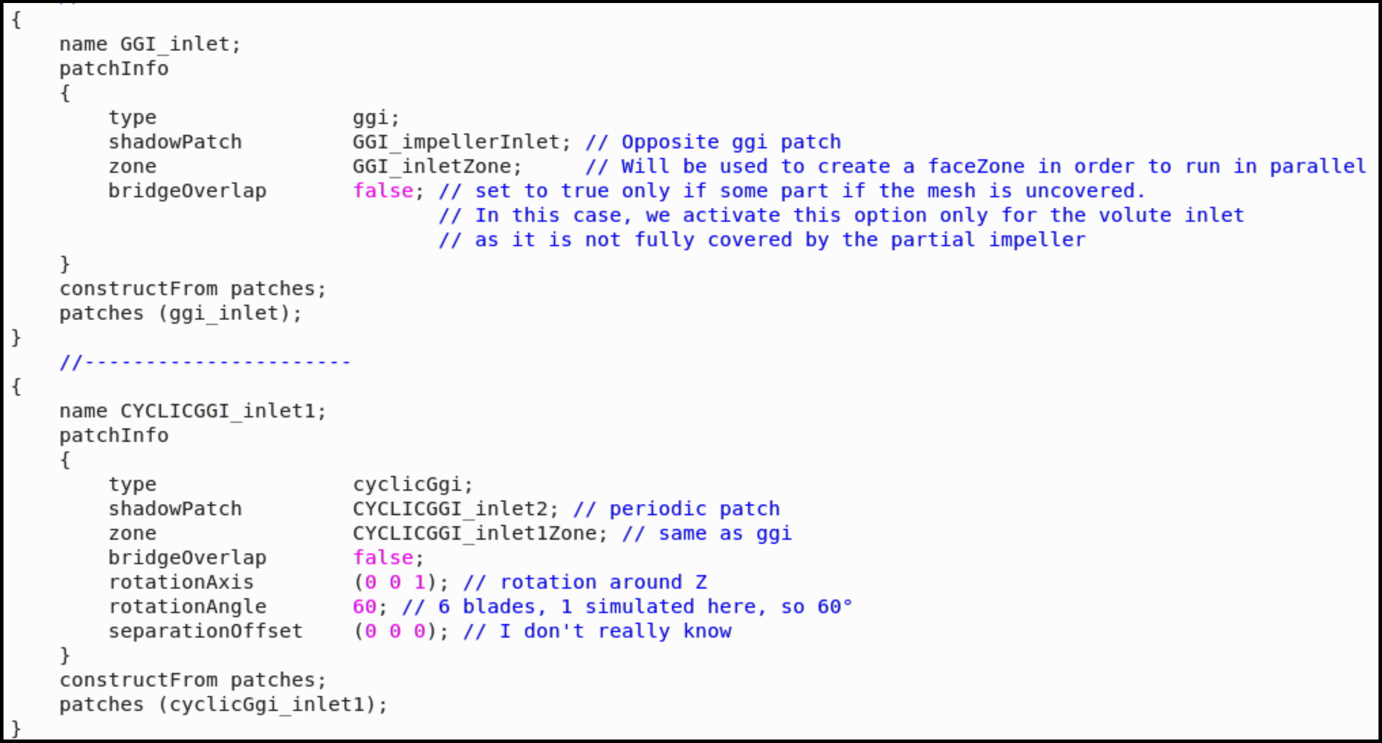


```
// Patches to create.
patches
(
    // INLET-----------------------------------------------------------------------------------------
    {
        name INLET;             //new name
        patchInfo
        {
            type patch;         //new patch type
        }
        constructFrom patches;
        patches (inlet);        //old patch. Be careful, if new and old patches have the same name, it won't work!
    }
        //--------------------
    {
        name .......;
        .
        .
        .
    }
        //--------------------
);
```

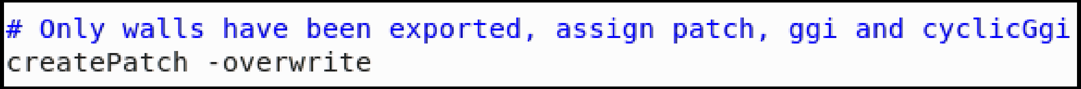
CONTENTS

Create the patches

For ggi and cyclicGgi. Be sure to use consistent names not to lose yourself (in the music, the moment you want it). Plus it will be easier to define the boundary conditions this way.



```
{
    name GGI_inlet;
    patchInfo
    {
        type                ggi;
        shadowPatch         GGI_impellerInlet; // Opposite ggi patch
        zone                GGI_inletZone;     // Will be used to create a faceZone in order to run in parallel
        bridgeOverlap       false; // set to true only if some part if the mesh is uncovered.
                                   // In this case, we activate this option only for the volute inlet
                                   // as it is not fully covered by the partial impeller
    }
    constructFrom patches;
    patches (ggi_inlet);
}
    //----------------------
{
    name CYCLICGGI_inlet1;
    patchInfo
    {
        type                cyclicGgi;
        shadowPatch         CYCLICGGI_inlet2; // periodic patch
        zone                CYCLICGGI_inlet1Zone; // same as ggi
        bridgeOverlap       false;
        rotationAxis        (0 0 1); // rotation around Z
        rotationAngle       60; // 6 blades, 1 simulated here, so 60°
        separationOffset    (0 0 0); // I don't really know
    }
    constructFrom patches;
    patches (cyclicGgi_inlet1);
}
```



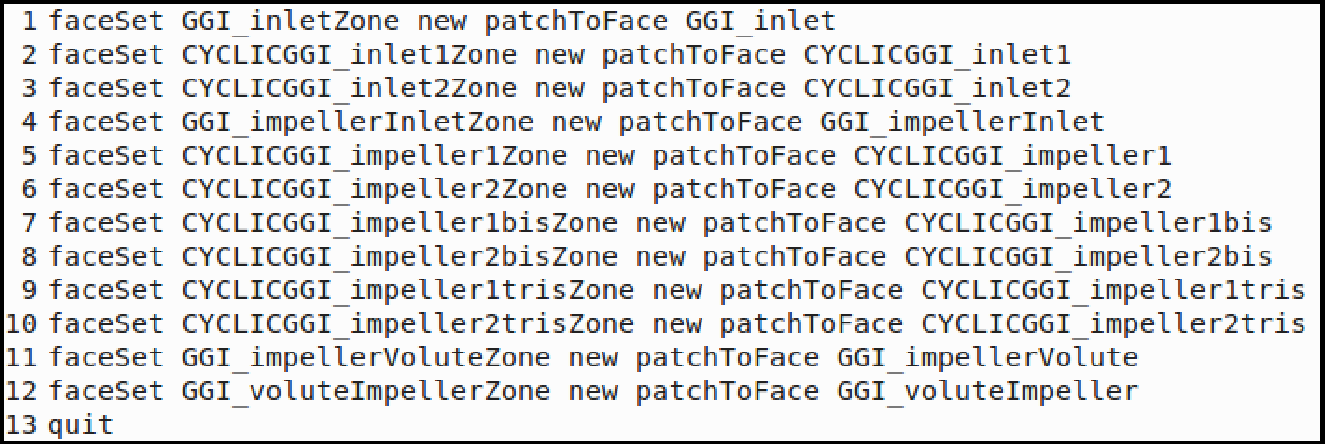
```
# Only walls have been exported, assign patch, ggi and cyclicGgi
createPatch -overwrite
```

More informations :
http://openfoamwiki.net/index.php/Sig_Turbomachinery_/_ERCOFTAC_centrifugal_pump_with_a_vaned_diffuser

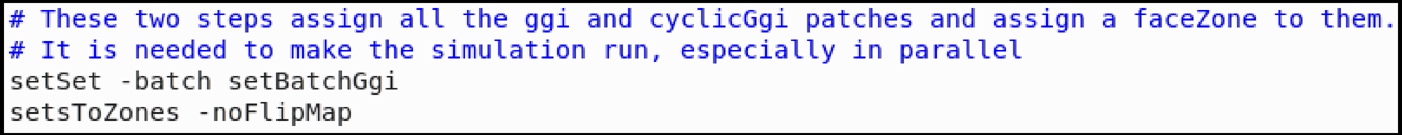
CONTENTS

Assign the faceZones

A setBatchGgi needs to be defined to create quickly those faceZones



```
faceSet GGI_inletZone new patchToFace GGI_inlet
faceSet CYCLICGGI_inlet1Zone new patchToFace CYCLICGGI_inlet1
faceSet CYCLICGGI_inlet2Zone new patchToFace CYCLICGGI_inlet2
faceSet GGI_impellerInletZone new patchToFace GGI_impellerInlet
faceSet CYCLICGGI_impeller1Zone new patchToFace CYCLICGGI_impeller1
faceSet CYCLICGGI_impeller2Zone new patchToFace CYCLICGGI_impeller2
faceSet CYCLICGGI_impeller1bisZone new patchToFace CYCLICGGI_impeller1bis
faceSet CYCLICGGI_impeller2bisZone new patchToFace CYCLICGGI_impeller2bis
faceSet CYCLICGGI_impeller1trisZone new patchToFace CYCLICGGI_impeller1tris
faceSet CYCLICGGI_impeller2trisZone new patchToFace CYCLICGGI_impeller2tris
faceSet GGI_impellerVoluteZone new patchToFace GGI_impellerVolute
faceSet GGI_voluteImpellerZone new patchToFace GGI_voluteImpeller
quit
```



```
# These two steps assign all the ggi and cyclicGgi patches and assign a faceZone to them.
# It is needed to make the simulation run, especially in parallel
setSet -batch setBatchGgi
setsToZones -noFlipMap
```

I don't know much what to say, see the link below.

More informations :
http://openfoamwiki.net/index.php/Sig_Turbomachinery_/_ERCOFTAC_centrifugal_pump_with_a_vaned_diffuser

CONTENTS

Boundary conditions, MRFZones, RASProperties, transportProperties, system files

Because you paid attention to the consistency of the names of your patches, it is now easy to define the boundary conditions, yey. Nothing special here, a flow rate inlet velocity is defined (30 l/s divided by 6 since we model only one passage). The pressure is fixed to 0 at the outlet. Be careful to the difference that may appears between the names used in OpenFOAM5 and the extend versions. For the turbulence parameters, the value are calculated according to the theory presented in József Nagy's videos.

For the other files, there is also differences with the official version in how to write the informations, but it is mainly the same.

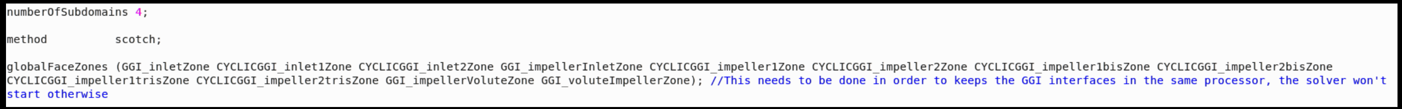
The system files (controlDict, fvSchemes, fvSolutions) won't be discussed, since the mesh is bad, it's useless trying to have great convergence with such a bad mesh.

József Nagy's videos :
https://www.youtube.com/watch?v=IPExwi2Ar-g

CONTENTS

Decompose the case

Here the step where we set the faceZones becomes really useful. In order to run in parallel, ggi and cyclicGgi neighbours patches need to be in the same processor, you can force this simply in decomposeParDict



```
numberOfSubdomains 4;

method          scotch;

globalFaceZones (GGI_inletZone CYCLICGGI_inlet1Zone CYCLICGGI_inlet2Zone GGI_impellerInletZone CYCLICGGI_impeller1Zone CYCLICGGI_impeller2Zone CYCLICGGI_impeller1bisZone CYCLICGGI_impeller2bisZone
CYCLICGGI_impeller1trisZone CYCLICGGI_impeller2trisZone GGI_impellerVoluteZone GGI_voluteImpellerZone); //This needs to be done in order to keeps the GGI interfaces in the same processor, the solver won't
start otherwise
```

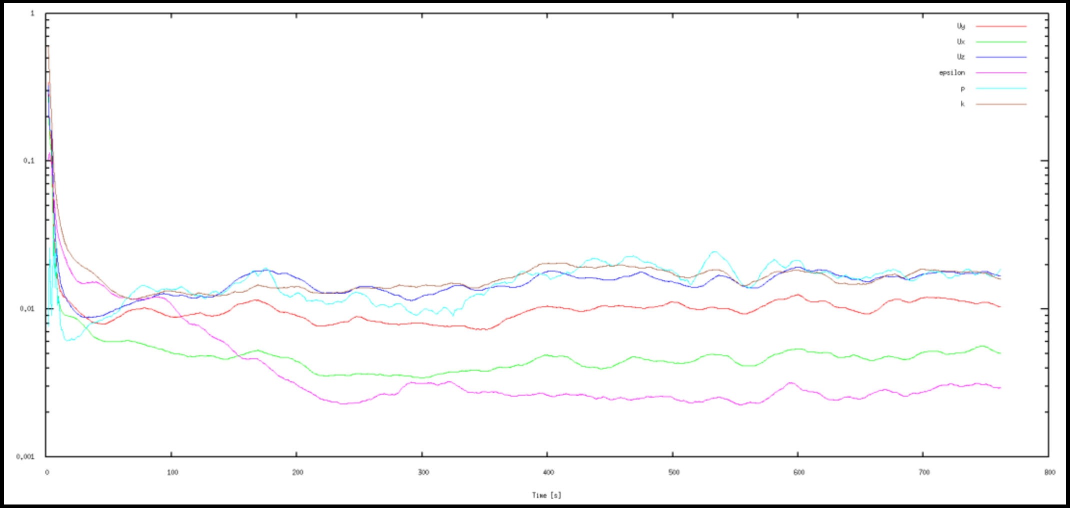
CONTENTS

Run the case

As said before, the mesh has a bad quality, thus the convergence is not the best.

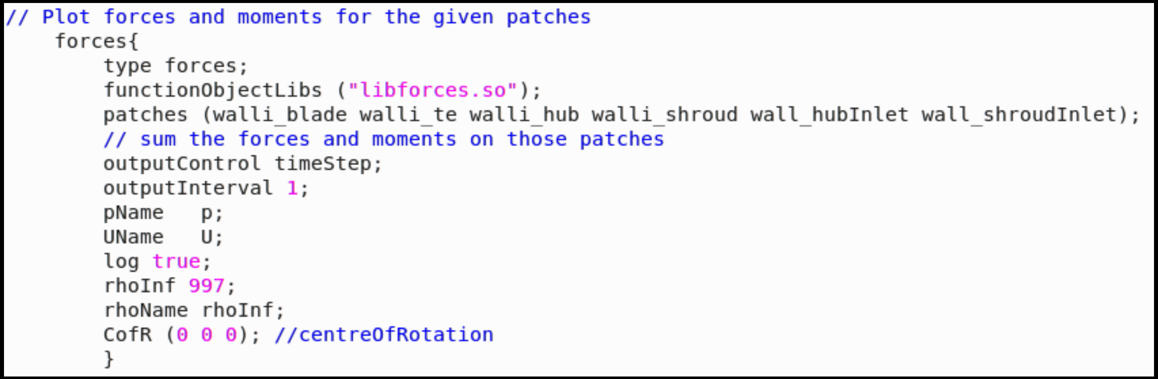
```
# Run MRFSimpleFoam in parallel on 4 cores
mpirun -np 4 MRFSimpleFoam -parallel > log.MRF &
#MRFSimpleFoam > log.MRF &

# Plot the residuals
pyFoamPlotWatcher.py log.MRF
# pyFoam is pre-installed with foam-extend
```



Post-processing

The torque is calculated by using the Z-moment for 1 impeller passage.



```
// Plot forces and moments for the given patches
    forces{
        type forces;
        functionObjectLibs ("libforces.so");
        patches (walli_blade walli_te walli_hub walli_shroud wall_hubInlet wall_shroudInlet);
        // sum the forces and moments on those patches
        outputControl timeStep;
        outputInterval 1;
        pName   p;
        UName   U;
        log true;
        rhoInf 997;
        rhoName rhoInf;
        CofR (0 0 0); //centreOfRotation
        }
```

Torque : $T_Z = 6.M_Z = 34.64\ N.m$

Shaft Power : $P_{sh} = T_Z.\omega = 10521\ W$

The head is calculed by using the surface averaged pressure at the inlet and the « real » outlet of the volute (before the outlet tube) and is measured in Paraview. Be careful, the pressure is divided by the density already, so :

Head : $H = \frac{p_{out} - p_{in}}{g} = 30.36\ m$

Post-processing

Water Power : $\mathrm{P_w} = Hg\dot{m} = 8909\ W$

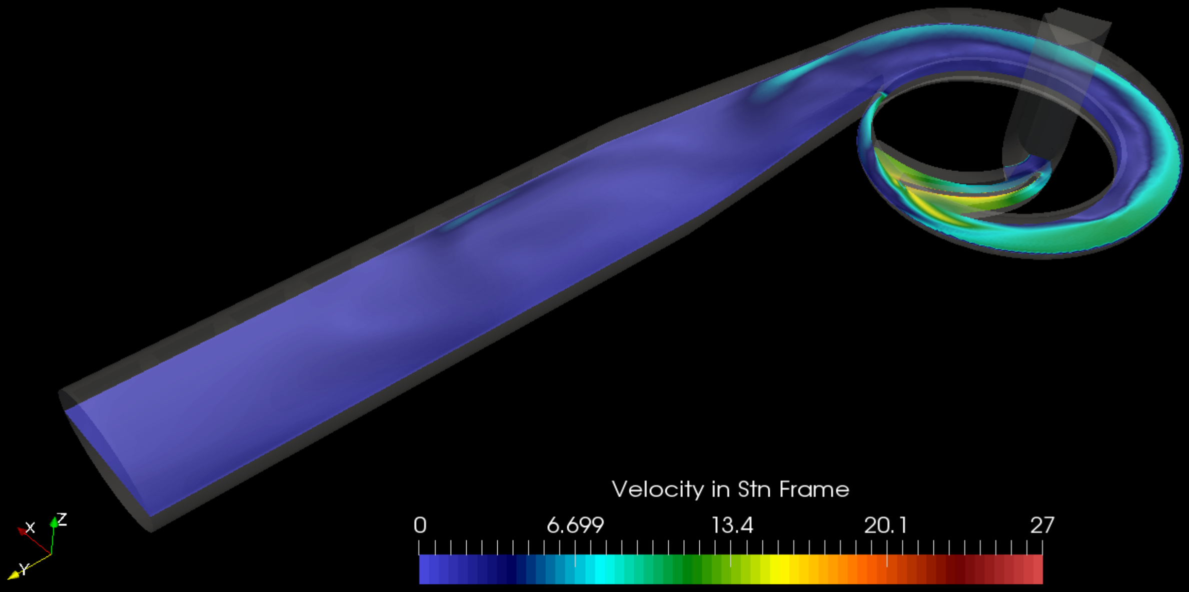
Efficiency : $\eta = \frac{P_w}{P_{sh}} = 0.847$

Comparison with CFX results :

	Head [m]	$\boldsymbol{P_{sh}}$**[W]**	$\mathrm{P_w}$ **[W]**	η
OpenFOAM	30.36	10521	8909	0.847
CFX	31.51	10562	9245	0.875
Diff	3.7%	0.39%	3.63%	3.2%

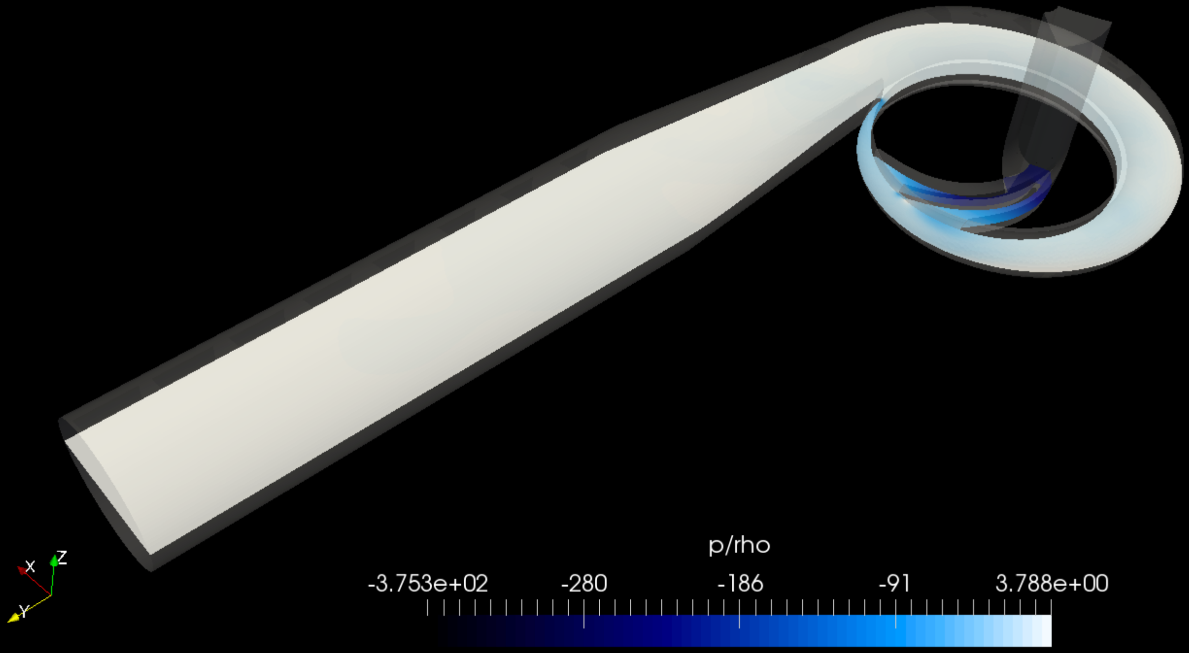
CONTENTS

Post-processing



CONTENTS

Post-processing



Post-processing

CONTENTS