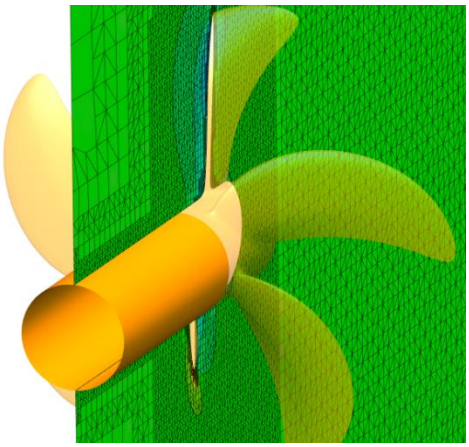
Coupling CAESES and OpenFOAM: Propeller Example

The purpose of this tutorial is to learn the integration of a 3rd party CFD software, in this case OpenFOAM. You will connect an existing propeller model and couple this with the propeller tutorial which is provided with OpenFOAM.

The propeller tutorial of OpenFOAM is basically an "Open Water" simulation with a sliding mesh approach. Due to the mesh motion, the actual simulation time will take some hours.

The propeller model of CAESES, which is provided with this tutorial, is prepared in a way, that it has the same dimensions and exported parts like the OpenFOAM propeller of the tutorial.

All you need is a basic understanding of how geometry gets created in CAESES, but it is not necessary. Some feature definitions are used as well in this model; see the feature definitions tutorial for more information.



The coupling is done with OpenFOAM 2.0, but could be used with all OpenFOAM versions as long as the dimensions of the propeller model do not change. The integration part can be done with CAESES versions > 3.1.1.

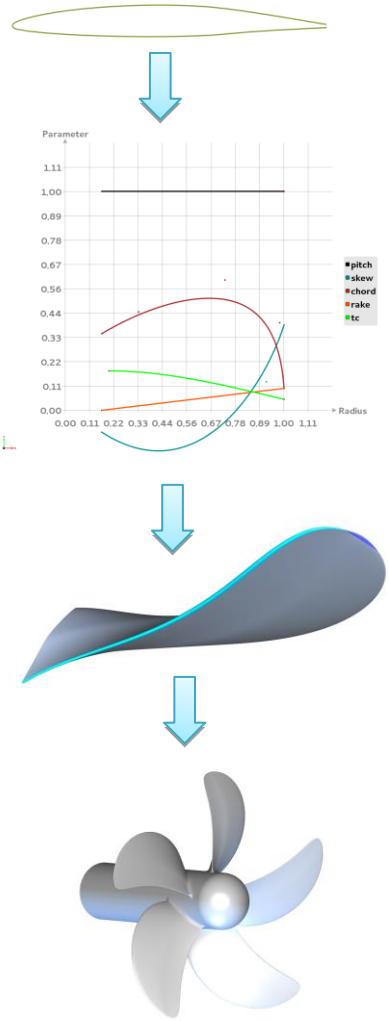
For any further questions please use the CAESES forum.

CAESES Project

This tutorial model can be found in the section *samples* > *tutorials* of the documentation browser.

1 Propeller Model: Brief Overview

The first step of modeling a propeller blade is to define the profile definition. This is done with a feature definition (using NACA 66) which is the input for a generic blade. Different parameters such as lift coefficient and chord length can be defined here. The generic blade is a surface type and needs a curve engine as input, where the functions for the radial distribution of parameters like rake, skew or pitch are provided. With these two-dimensional functions the three dimensional blade is created. The functions are usually smooth B-Spline curves where the x-value defines the radial coordinate and the y-value the value of the parameter. These y-values can be controlled by design variables. All the predefined design variables are located in *01_bladeModeling|1_parameters*. The yellow objects are design variables and the blue ones are parameters. You can play around with the design variables in certain ranges and see how the parametric model gets changed.



The generic blade is not a closed surface, which is why the tip and trailing edge have to be modeled separately (*01_bladeModeling|2_blade|closings_blade01*). All surfaces are combined into a Brep with the name “BladeBRep”.

Shaft and hub BReps are prepared for the OpenFOAM tutorial. This is done in the scope *03_prepareForOpenFoamTutorial*.

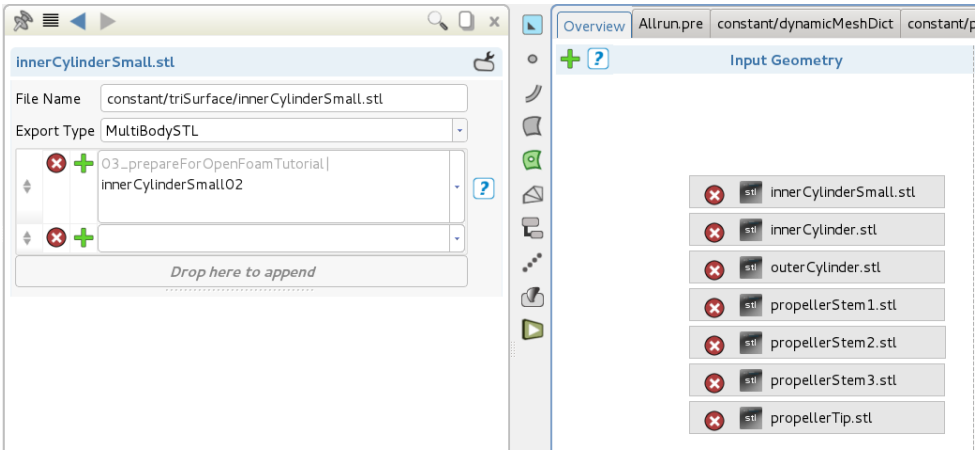
The first step is to rotate the created blade by using the feature “PeriodicBRep”. Afterwards the all the BRep parts are combined and scaled to the dimensions of the OpenFOAM tutorial. In order to give the exported STL patches names, we use custom colors with the desired names, in this case “ascii”. This is done with a feature called “ExportSTLPatchNamesByColor”. The BReps will finally be exported as *Multi Body STL*.

2 Software Connector – Start with Input Geometry

Let's get started: The *software connector* is the widget where external tools can be plugged-in. We will set up the connection step by step:

- Open the file "81_OpenFOAM_Propeller.fdb" from the tutorials section (samples > tutorials) in the documentation browser and safe it somewhere on your harddisc.
- Go to the *Connections* tab in the pull down menu and create a software connector.
- Drag and drop the 7 BReps from *03_prepareForOpenFoamTutorial* into the *Input Geometry Window* of the software connector.
- Click on each of the 7 parts in the connector, and specify the export name and the type, respectively. Set the export type for each geometry part to *MultiBodySTL*. Since the STL location is in a subfolder of the OpenFOAM project directory, we have to specify the file names as follows:

    - constant/triSurface/innerCylinderSmall.stl
    - constant/triSurface/innerCylinder.stl
    - constant/triSurface/outerCylinder.stl
    - constant/triSurface/propellerStem1.stl
    - constant/triSurface/propellerStem2.stl
    - constant/triSurface/propellerStem3.stl
    - constant/triSurface/propellerTip.stl

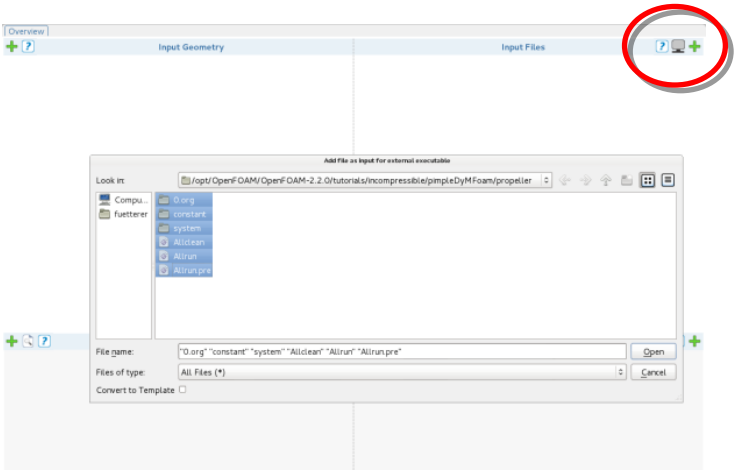


✓ By specifying the path e.g constant/triSurface/myGeometry.stl, CAESES will automatically create this folder in the current design directory.

3 Input Files

In the next step, we will insert the OpenFOAM text files:

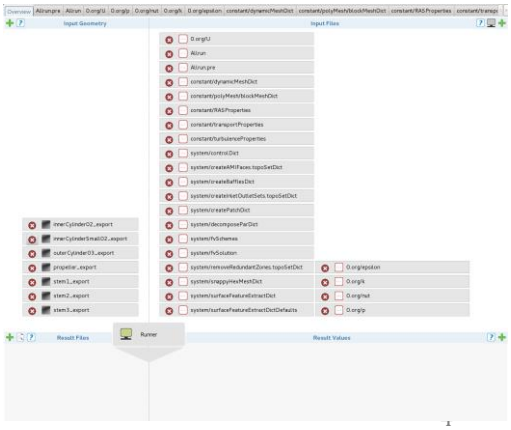
- Click on the plus button in the *Input Files* window. Select the files from the OpenFOAM propeller tutorial. You can find it in …/OpenFOAM-2.x.x/tutorials/incompressible/pimpleDyMFoam/propeller.



- You can delete all geometry files which end by .obj and the Allclean file.

Now, for each file the absolute path is shown. This means that the software connector references to the absolute patch on the disc. But as we want to modify the files, we want the copy and load the files into the CAESES project.

- This is simply done by double-clicking on each file: With a double click, you will be directed into the file window.
- Click on *Overview* to see the software connector main window again.

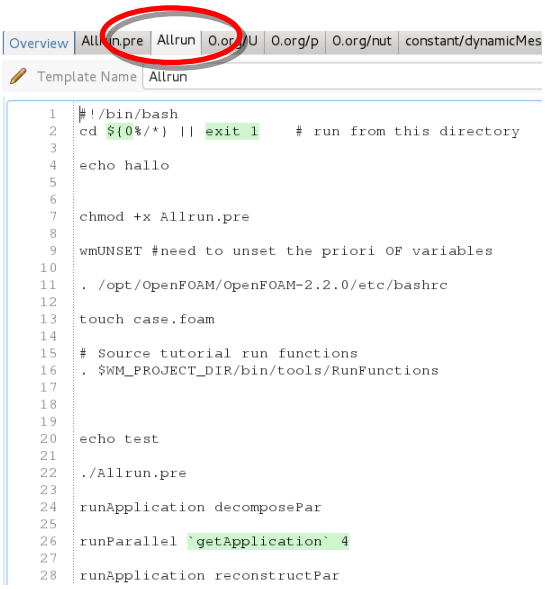


4 Allrun File

We modify the Allrun file in this step, note again that we work on a copy of the original file.

- Double-click on “Allrun”.
- Since the Allrun script calls the Allrun.pre script, which has no permission to be executed, we have to insert a line into the Allrun file:

```
chmod +x Allrun.pre
```



- Insert another line: If you don’t want to start CAESES from the terminal and you want to run OpenFOAM, we also need to source the OpenFOAM environment variable. This could be done for example in the following way:

```
wmUNSET

source  /opt/OpenFOAM/OpenFOAM-2.2.0/etc/bashrc
```

- For postprocessing reasons, we need to create the “case.foam“ file by adding the following command into the Allrun script:

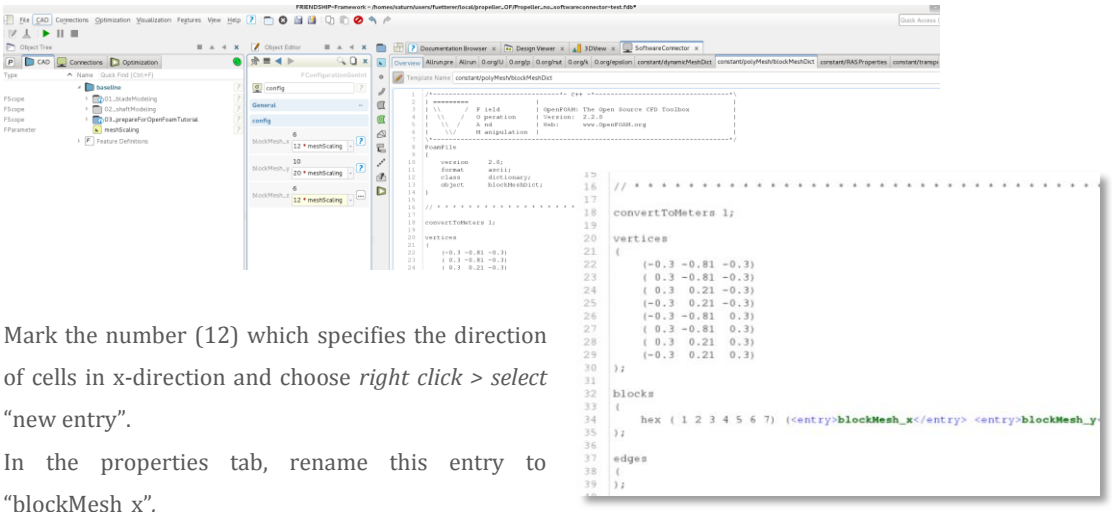
```
touch case.foam
```

5 Introduce Parameters in Dict Files

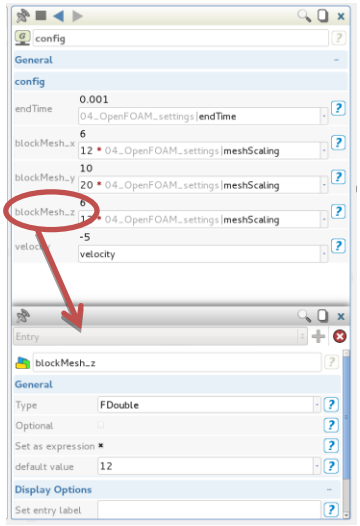
In this step, we introduce some parameters, which control OpenFOAM settings.

First, it is useful to add a parameter which scales the mesh uniformly in all directions.

- Double-click on the "blockMeshDict" file.
- Switch to plain mode by clicking on the little pen on the top left of the file.



- Mark the number (12) which specifies the direction of cells in x-direction and choose *right click > select* "new entry".
- In the properties tab, rename this entry to "blockMesh_x".
- Change the value to 12*0.5.
- Then select the value 0.5 in the editor, choose *right click > create design variable*, which could be renamed as "meshScaling".



You can find the design variable in the *CAD* tab below the three scopes.

- Do the same for the other directions, while using the "meshScaling" design variable for all three directions.

We can also create a parameter which controls the end time of the solution.

- In the software connector, select the "controlDict" file and create a new entry for the item "endTime".
- In the *Configuration* tab, create a parameter for the "endTime" similar to what we did for the "meshScaling", and reduce the time to 0.001.
- Put the parameter "endTime" and the design variable "meshScaling" into a scope called "04_OpenFOAM_settings".

6 Consider STL in Dicts

We have to adjust “snappyHexMeshDict” as well as “surfaceFeatureExtractDict” since we will use an .stl instead of the .obj format.

- Select “snappyHexMeshDict” and replace the endings of the imported geometry from “.obj” and ”.obj.gz” to “.stl”.
- Select “surfaceFeatureExtractDict” and replace the endings of the imported geometry from “.obj” to “.stl”.

In order to get the moments and forces of the calculation, we have to include following code into the end of “controlDict”:

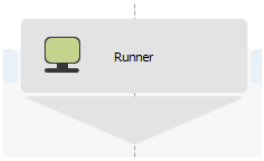
```
Functions
(
        forces
        {
                type                    forces;
                functionObjectLibs      (“libforces.so”);
                patches                 (propellerTip);
                CofR                    (0 0 0);
                rhoName                 rhoInf;
                rhoInf                  998;
                outputControl           timeStep;
                outputInterval 1;
        }

);
```

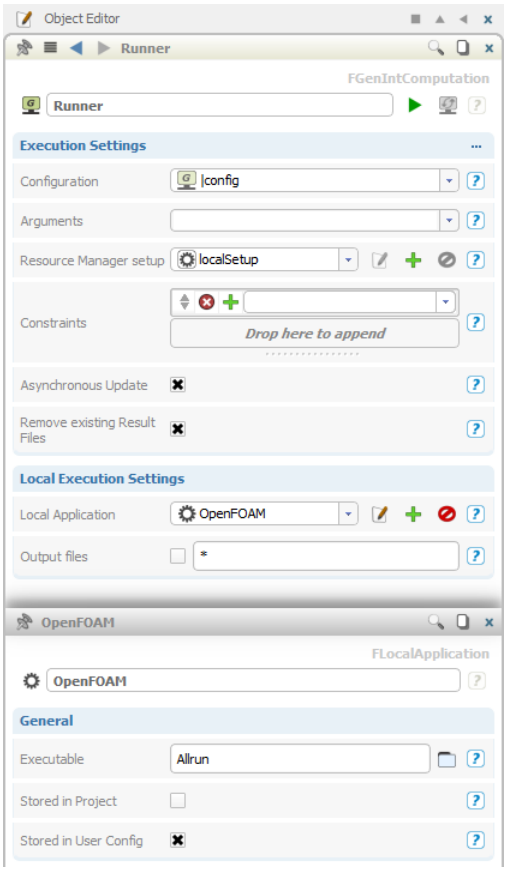
7

Software Connector: Computation

For postprocessing we need to have results files and values, which will be inserted in the *Results Files* and *Results Values* window of the software connector. In order to get these, we have to trigger a first run – either with CAESES, or without CAESES. Since we already changed parts of the input files in the previous steps, it makes sense to trigger the computation from within CAESES. For this, we have to set up the *computation*:



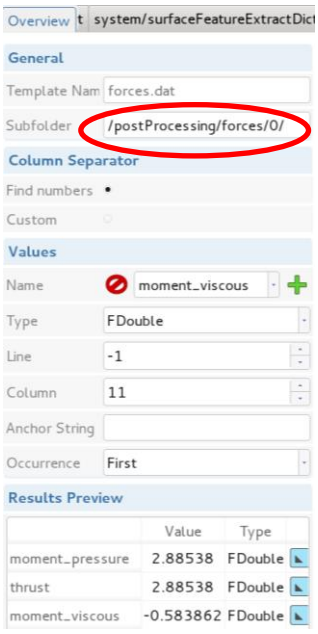
- Click on "Runner" in the middle of the software connector.
- In the object editor of the selected object "Runner", activate "Remove existing Result Files". This makes sure that with each run the previous simulation files get deleted automatically, and is similar to the "Allclean" command.
- Create a new "Local Application" by clicking on the plus-button next to the attribute, and call it "OpenFOAM".
- As executable write "Allrun". This will trigger the "Allrun" script.
- Run the computation by clicking the green run button (▶).
- While the computation is running, you can check the output in the *Task Monitor*.



When the computation is done, check the results on your hard disc, to see how the results are handled by CAESES: A new folder was created, with the name of the current project file (*.fdb). In this folder all results can be found.

8 Result Values

In order to assess the simulation, we need to have results files which provide for example pressure drop or, as given in this case, the torque and forces. These values can be extracted from any text files, but usually from *.csv or *.dat file formats. Since CAESES needs to know where the desired values are located in each design directory, we have to provide an example file.



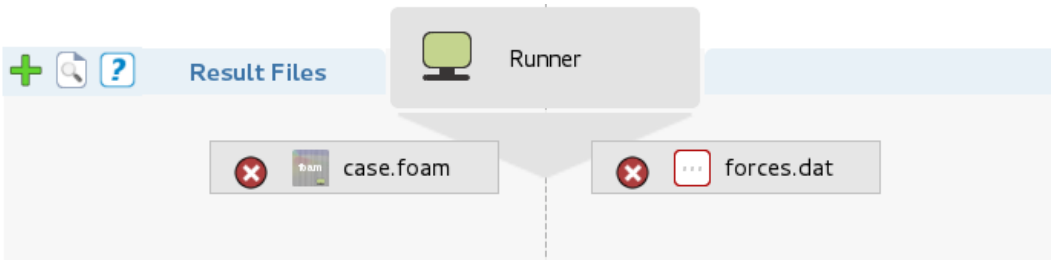
- In your file explorer, go to the baseline directory which was created during the last run and check for the "forces.dat" file in *postProcessing/forces/0/forces.dat.*
- Add the "forces.dat" file to the *Results Values* window of the software connector by using either drag & drop, or by using the plus button at the window.
- Double click on "forces.dat" in the software connector to see the content.
- In the upper left corner of the window, add a subfolder *postProcessing/forces/0/*
- Add a value by clicking on the plus button and name it "thrust".
- Set *line* to -1 (this means always the last row of the file).
- Set column to 2.
- Add a value by clicking on the plus button again, and name it "moment_pressure".
- Set line to -1.
- Set column to 8 if you are using OpenFOAM 2.2.0 or less otherwise set the column to 11.
- Add a value by clicking on the plus button, and name it "moment_viscous".
- Set line to -1.
- Set column to 11 if you are using OpenFOAM 2.2.0 or less otherwise set the column to 14.
- Now create a parameter of each value by clicking on the blue parameter symbol in the table preview. These parameters will be the objectives of the simulation at a later stage.
- Select the three evaluation parameters, and create a scope. Set the name of this scope "05_OpenFOAM_evaluation" (note, names can be user-defined i.e. arbitrary).

Remember that these parameters are the result values of the last iteration or time step. If you have a strong oscillating result, you might average the values over a specific time. This will be explained in more detail later on.

9 Result Files

In the *Result Files* window of the connector, output files from the CFD calculation are referenced. Typical output files are pictures, tables, text files and CFD solutions.

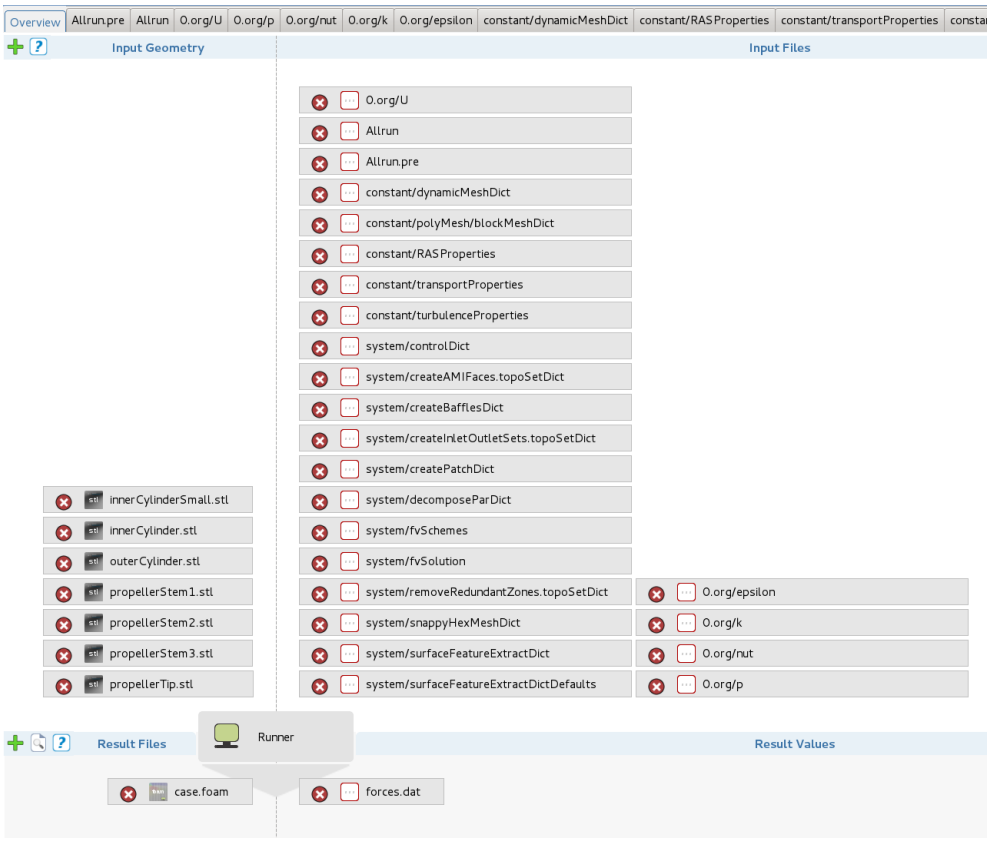
- Add the "case.foam" file from *manual_results/baseline/Runner* to this window, either by using drag & drop, or by using the plus button in the corner.



10 Running the Case

Now the software connection is completed, and a first test case can be started.

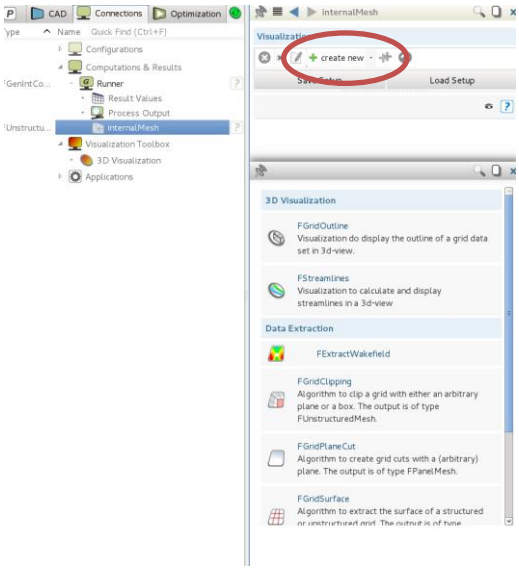
- Increase the parameter “endTime “ to 0.1, which is the original value from the OpenFOAM tutorial.
- You can also increase the “meshScaling” factor, but this will increase the computation time.
- Go back to the software connector and run the simulation (again by using the ▶ button).



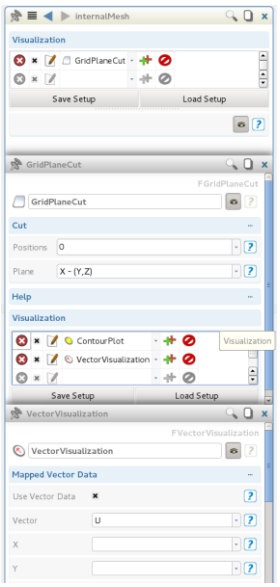
11 Postprocessing

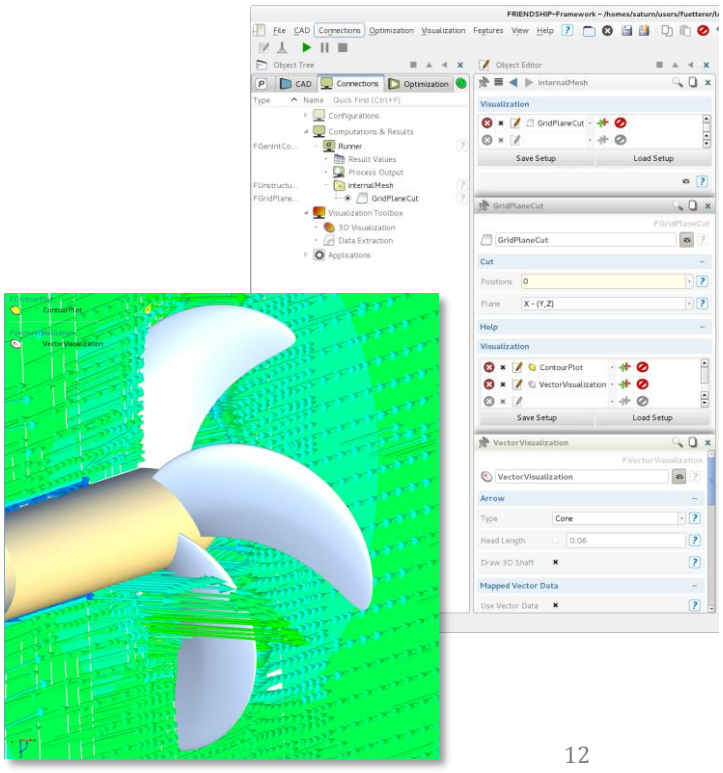
When the computation has finished, we can do the postprocessing and visualize the results.

- Go to the *Connections* tab and open *Computations & Results > Runner*.
- Select the grey “internalMesh” symbol.
- Create a new *Visualization*.
- A new window appears. Select “FGridPlaneCut”.
- Set the position to “0”.
- Create a new visualization at the bottom of this window.
- Select “FContourPlot”.
- Choose the mapped data you want to visualize, for example “U”. In the 3D window the visualization will appear.



You could also create a vector visualization, streamlines, iso-surfaces or mesh visualizations.



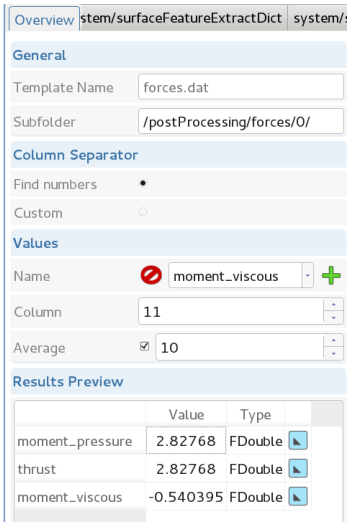


12

Averaging Result Values

Some improvements can be made which make it easier to evaluate the simulation results. For instance, averaging result values:

- Double click on the forces.dat file in the Results Values part of the software connector
- Select the thrust entry.
- Select Average and set the value to 10. This means that results of the last 10 rows is averaged.
- You can do the same for the moments.



Another step that would follow up in an optimization process is to calculate the efficiency. Therefor you will need the averaged values of the thrust and the moments over a specific number of revolutions (for example the average value of the last revolution). To calculate this you need the transient time step, the total number of revolutions and the time steps per revolution.