OpenFOAM Training

WIND. ASSURING CONFIDENCE THROUGH COMPETENCE

bernhard.stoevesandt@iwes.fraunhofer.de

Fraunhofer IWES

Introduction

Documentation

- OpenFOAM is open-source and lacks a little in documentation
- Documentation and help can be found in

  the official user guide

  http://www.openfoam.org/docs/user/

  the OpenFOAM installation directory

  (e.g. with commands grep, locate, find)

  the C++ source guide

  http://www.openfoam.org/docs/cpp/

  the CFD online forum

  http://www.cfd-online.com/

Installation directory

- OpenFOAM has a lot of various solvers, applications, libraries, tutorials etc.
- But where to search for all the options available?
- Important folders in the installation directory are:

applications/solvers

source code of the solvers

applications/test

examples of functions for coding in OpenFOAM

applications/utilities

more applications for post-processing, meshing etc.

tutorials

tutorials for all solvers and several applications, sorted by field of interest (incompressible, mesh etc.)

src

contains the source code for the libraries with fundamental functions (finite volume mesh, discretization schemes etc.)

Some useful tips

- Don’t panic!
- A lot of problems other users already had => search for help on **http://www.cfd-online.com/**
- Use “bananas” as a place holder to get a list of allowed entries from OpenFOAM
- Check the code for detailed information and examples
- Compile applications with **wmake** and libararies with **wmake libso**

Compiling OpenFOAM

- If you want to use a version of OpenFOAM without deb packages for your system, you need to compile it yourself

- Standard settings in $FOAM_INST_DIR/etc/bashrc have to be changed
  - e.g. to foamInstall=/opt (if you are root) or to foamInstall=$HOME/$WM_PROJECT
- Some versions need small changes for the use of ThirdParty software such as paraview or scotch

- Compile on many processors using export WM_NCOMPPROCS =nProcs

- Run ./Allwmake >& log.Allwmake1

Fraunhofer IWES

Airfoil Simulations

2D-Airfoil simulations

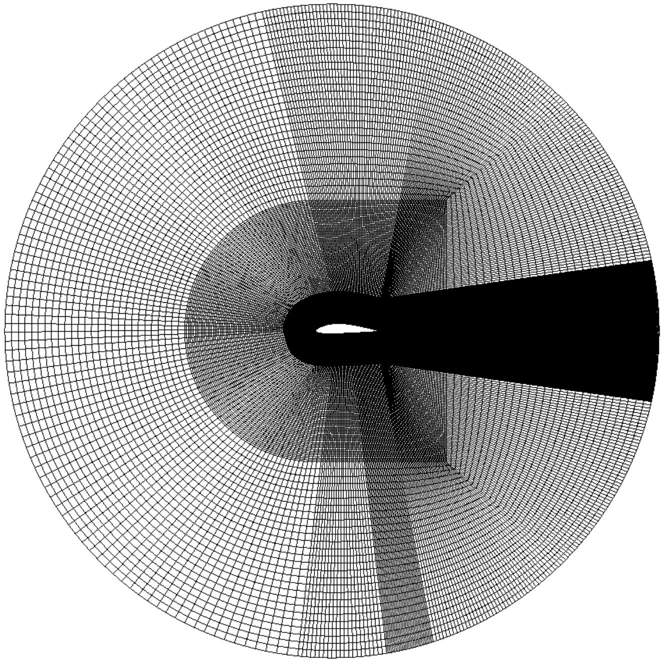
- Mesh generation possible with blockMesh or snappyHexMesh
- Better meshes usually with external tools, e.g. construct2D and conversion to blockMeshDict
- Choice of turbulence models for fully-turbulent flows
  - kOmegaSST
  - SpalartAllmaras
- Models for transition
  - kkLOmega
  - kOmegaSSTgamma (not yet in standard OpenFOAM)
- Initialization with potentialFoam
- Case run with SIMPLE-consistent (due to less under-relaxation much faster than SIMPLE)

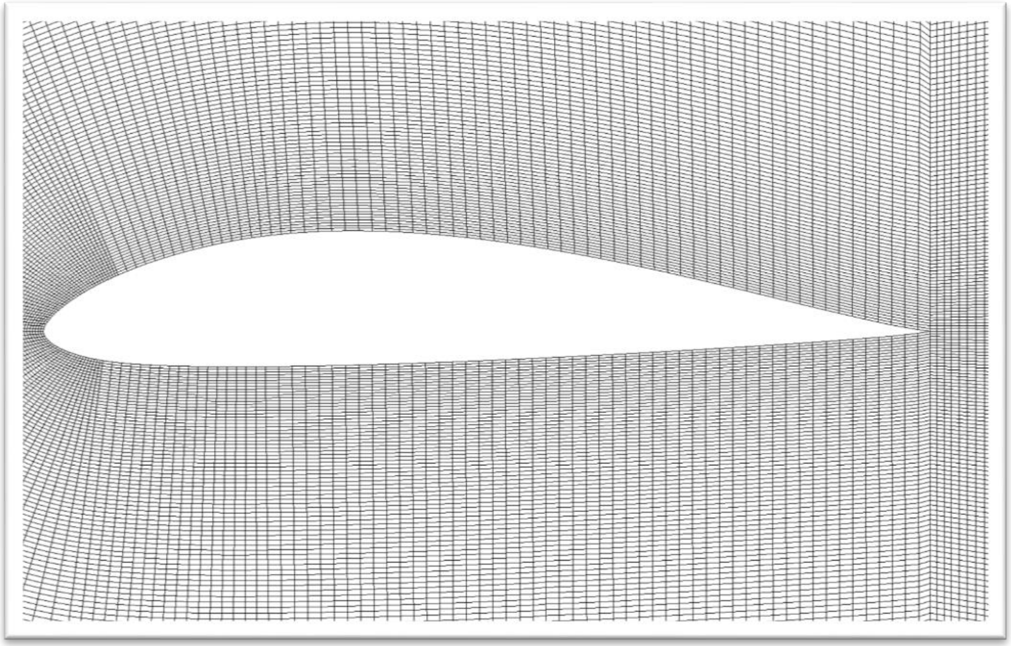
- Full automatization of set-up possible

Fraunhofer IWES

Meshing airfoils with blockMesh

- Meshing with blockMesh is possible in general
- Script-based approach useful
- Block-structure is still clearly visible and smoothing is needed



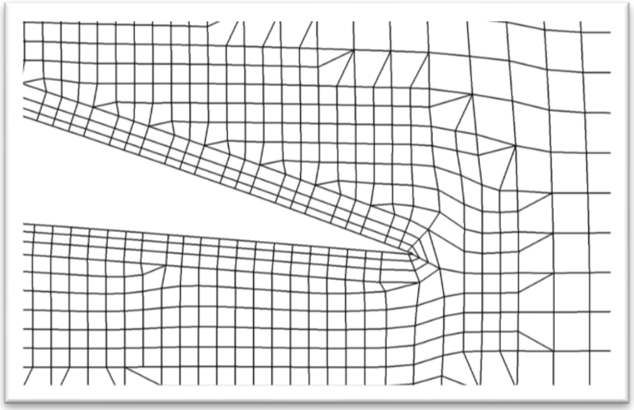


Fraunhofer IWES

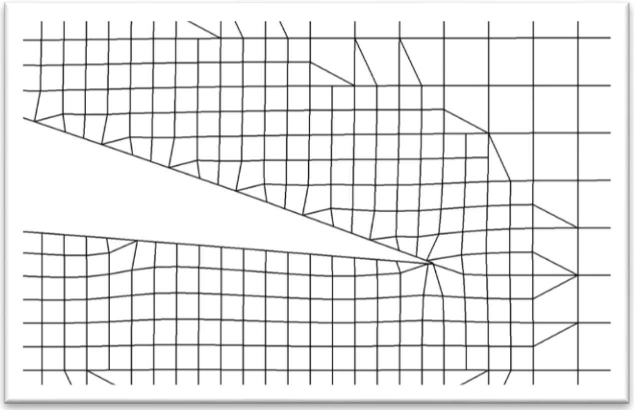
Meshing Airfoils with snappyHexMesh

- Meshing with snappyHexMesh is possible in general
- However, high level of experience required for good meshes
- Examples show results by changing parameters in tutorial incompressible/airfoil2D

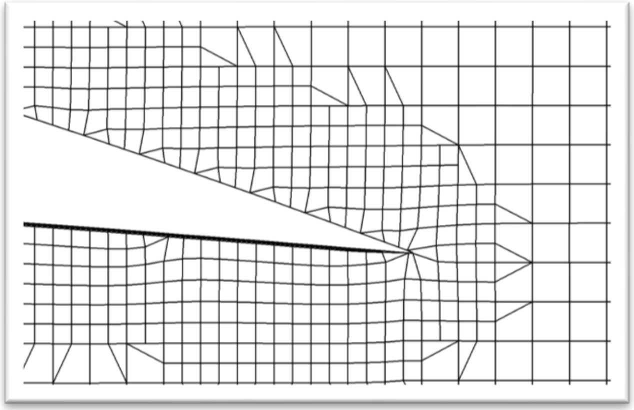
original tutorial



more cells in boundary layer fails



decrease of off-wall distance fails



Fraunhofer IWES

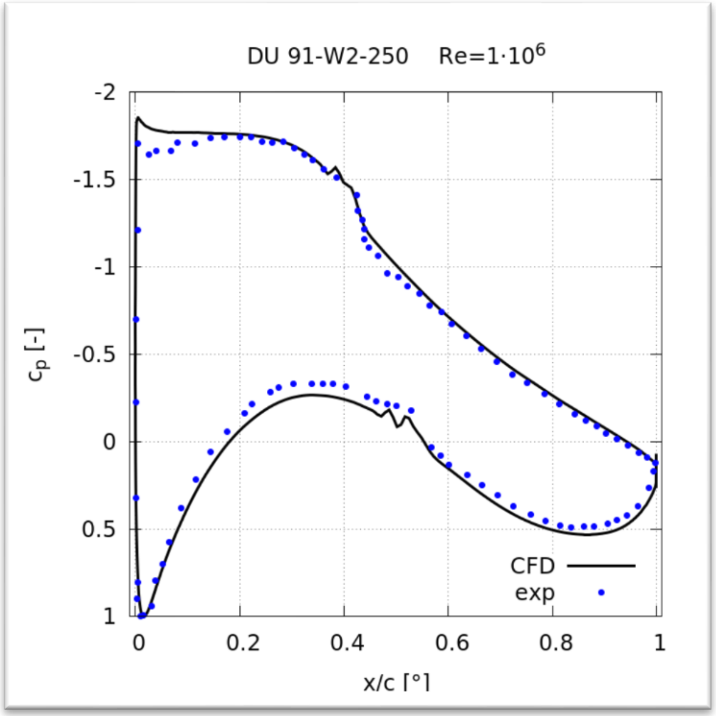
Tutorial for airfoil simulation

- Mesh of DU 91-W2-250 is generated via construct2D and **blockMesh**
- 2D mesh consists of approx. 130,000 cells
- Fully-turbulent (**SpalartAllmaras**) and transitional turbulence modelling (**kkLOmega**)
- Resolved boundary layer ($y^+ \approx 1$) at $Re = 1 \cdot 10^6$

- Compile application **calculateCp2D** with **wmake (OF 4.1)**
- Run the airfoil tutorial by IWES
- Compare pressure distribution with experimental data

  K. Boorsma, *Comparison of experimental and computational aerodynamic section characteristics of the DU 91-W2-250 profile*. Delft University of Technology, 2003.

- Location of transition visible in c_p-curve



Fraunhofer IWES

2.5D-Airfoil simulations

- Extrusion of 2D mesh via extrudeMesh (requires extrudeMeshDict in system folder)
- Use of IDDES (SpalartAllmarasIDDES) in order to avoid Grid-Induced Separation, Log-Layer Mismatch etc.

- For relatively small angles of attack:
  - 10-30 cells in spanwise direction can be enough
  - Use of wall functions possible ($y^+>30$)
  - Wall functions can lead to faster computation due to easier fulfillment of Courant number

- For higher angles of attack:
  - 100 cells in spanwise direction and resolved boundary layer ($y^+=1$)

- In general, hybrid methods like IDDES require high computational effort on several cores

Turbine Simulation Concepts used at IWES

Fraunhofer IWES

Turbine simulation concepts

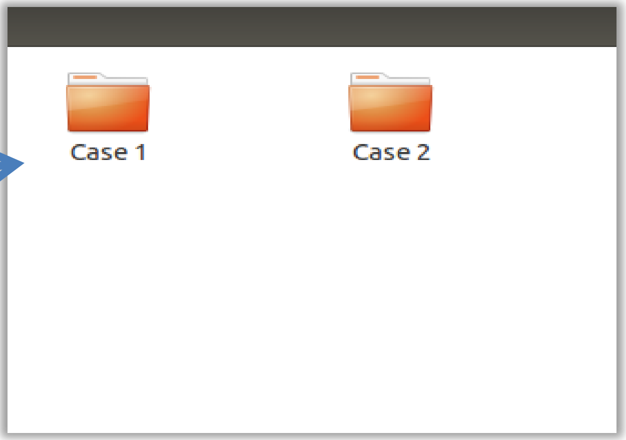
- Unfortunately no overset grid / Chimera capabilities available for OpenFOAM ☹
  - However, we are working on it…
- Meshes for full rotor simulations have often to be "assembled" from different sub-meshes
- Non-conformal sub-meshes can be connected by sliding interfaces (AMI)
  - However, AMI interfaces **slow down** transient simulations considerably
- Therefore, avoid AMI interfaces, if possible!
- In the current implementation of OF, AMI interfaces are re-computed every time step, also if the relative position of patches doesn't change! (DyM)

Fraunhofer IWES

Application mergeMeshes

- All kinds of sub-meshes can be merged into the final, complete mesh
- Cell types can be different in each sub-mesh

- The tool of choice for merging meshes is the application **mergeMeshes**
- Use **mergeMeshes Case1 Case2** to merge Case 2 into Case 1
- New merged mesh can be found in folder of Case 1 (in a new time folder)
- **mergeMeshes** needs to be executed from one level above the case folders



Execute **mergeMeshes** from this folder level

Fraunhofer IWES

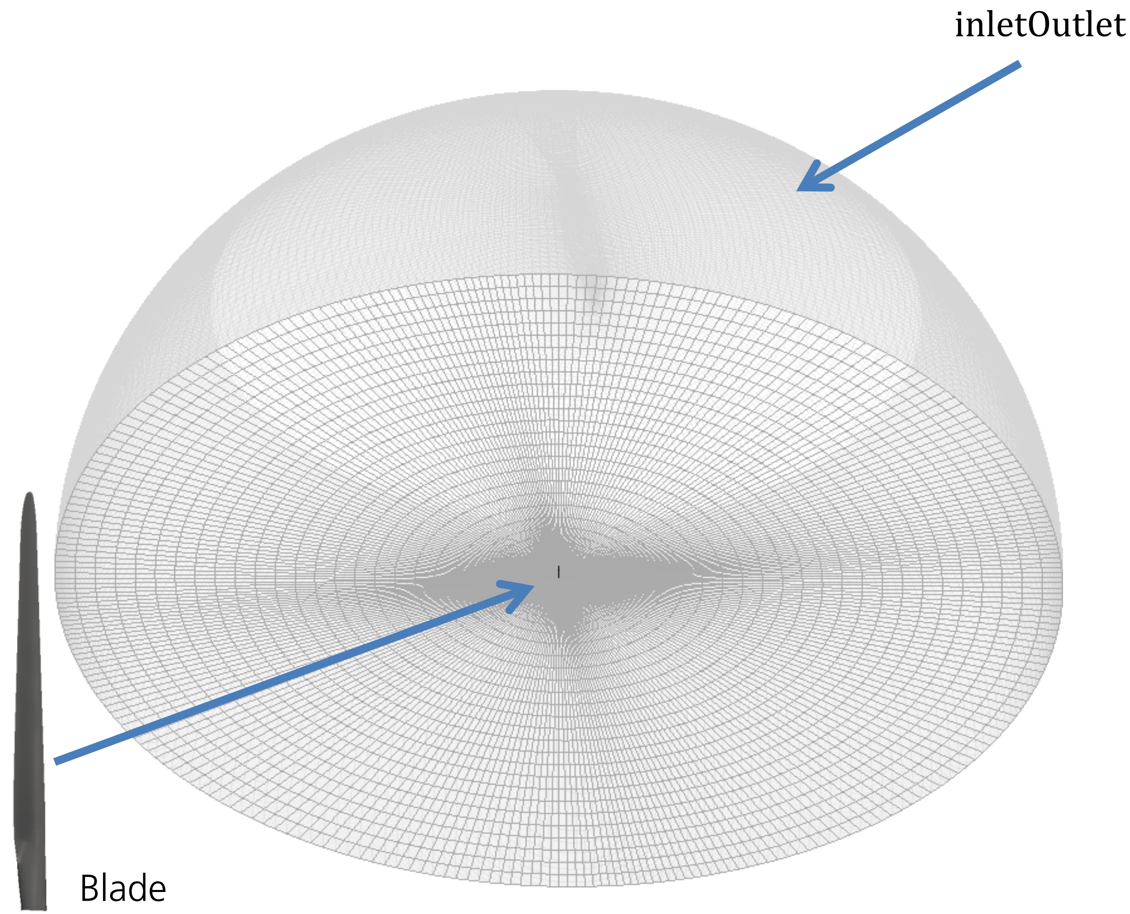
Application stitchMeshes

- After two sub-meshes are merged and a pair of boundary patches fits exactly to each other, they can be stitched
- If the two patches are stitched, they become an internal boundary
- In that case, an AMI can be avoided!

- The tool of choice for stitching meshes is the application stitchMesh
- Use stitchMesh BoundaryPatch1 BoundaryPatch2 -perfect to stitch BoundaryPatch1 and BoundaryPatch2
- The new mesh is written into a new time folder
- Delete the old empty entries in the file polyMesh/boundary
- Also change the overall number of patches in polyMesh/boundary

Concepts for full rotor simulations I

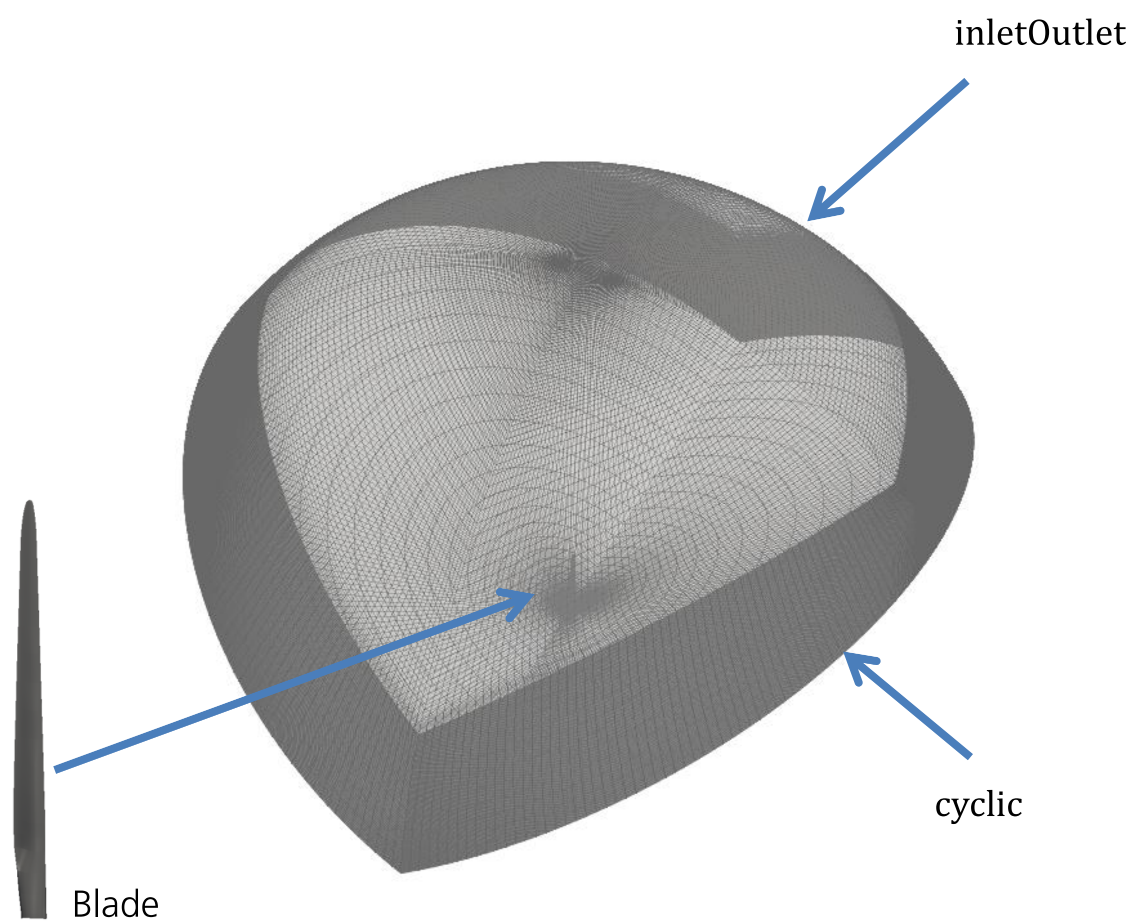
- Concept I:
  - Fully structured grid
  - 1 or 2 blade(s)
  - Half-spherical domain
  - No AMI needed
  - DyM for all cells



Fraunhofer IWES

Concepts for full rotor simulations II

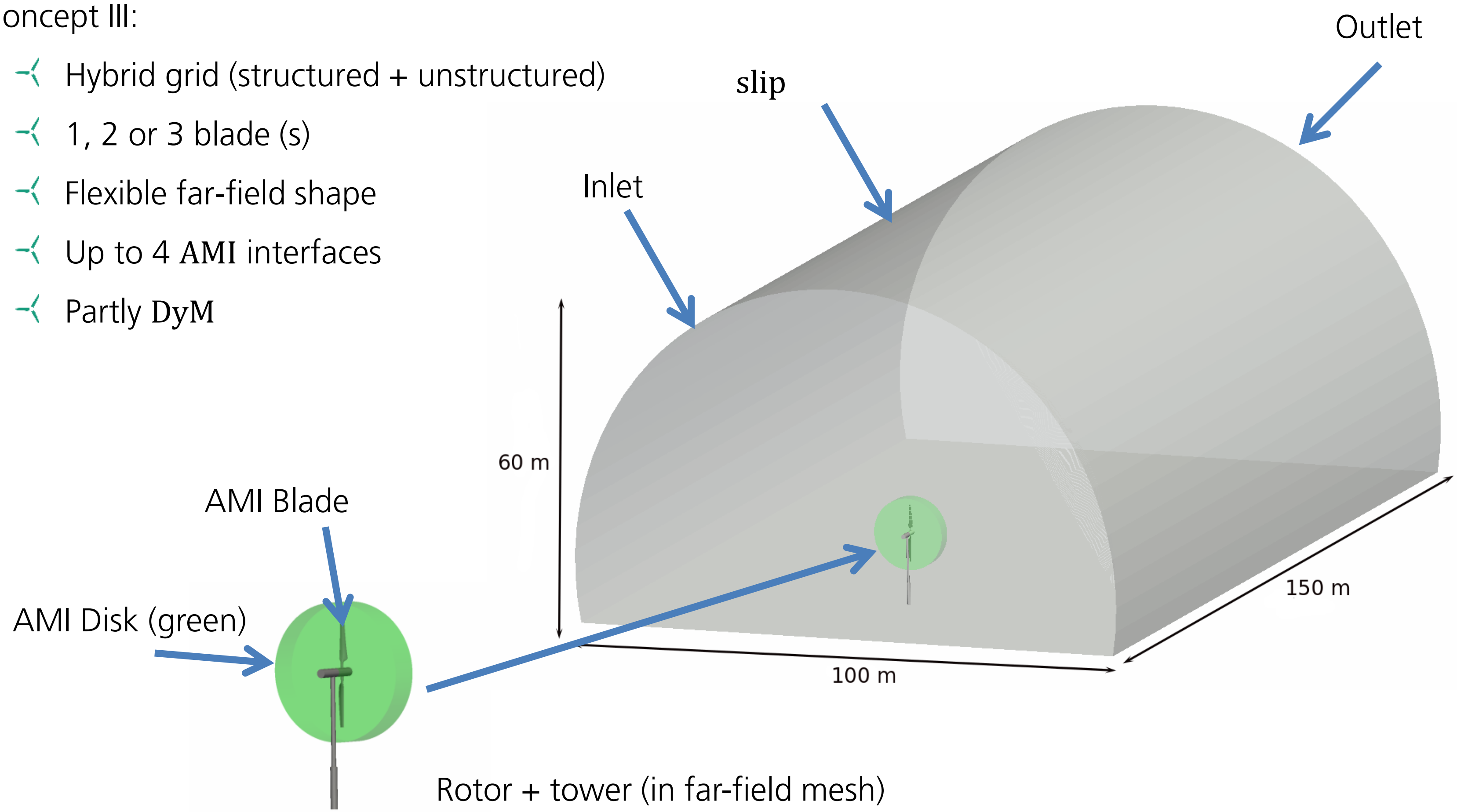
- Concept II:
  - Fully structured grid
  - 1 or 3 blade(s)
  - 1/3-spherical domain
  - No AMI needed
  - DyM for all cells



Fraunhofer IWES

Concepts for full rotor simulations III

- Concept III:
  - Hybrid grid (structured + unstructured)
  - 1, 2 or 3 blade (s)
  - Flexible far-field shape
  - Up to 4 AMI interfaces
  - Partly DyM

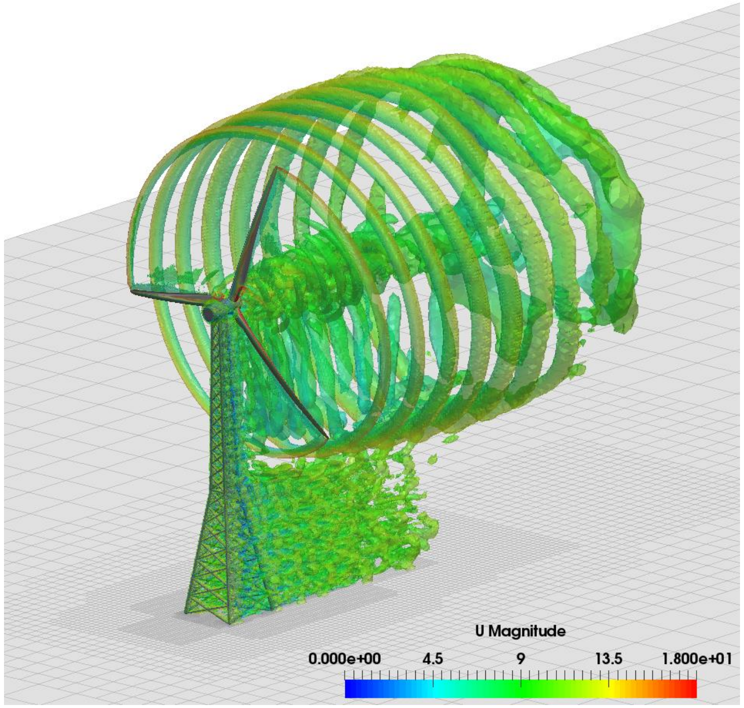


Rotor + tower (in far-field mesh)

Fraunhofer IWES

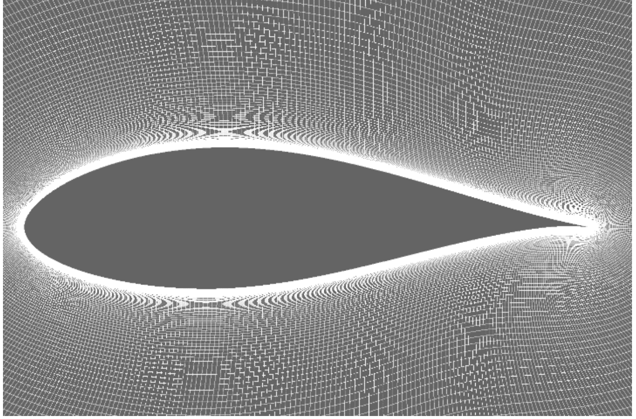
Example of full rotor simulation with tower

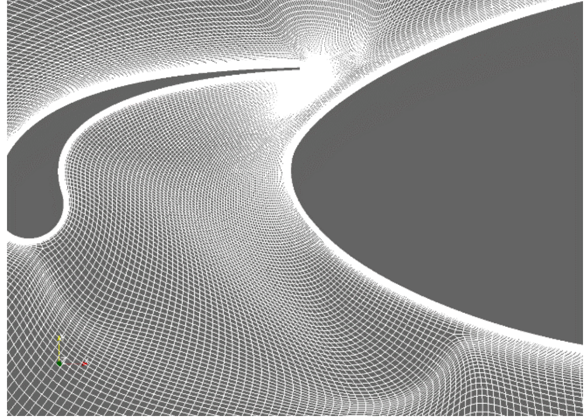
- Combinations of concept I-III
- Almost everything is possible
- Sub-meshes just need to fit to each other!
- Be careful with AMI in regions of high flow gradients!

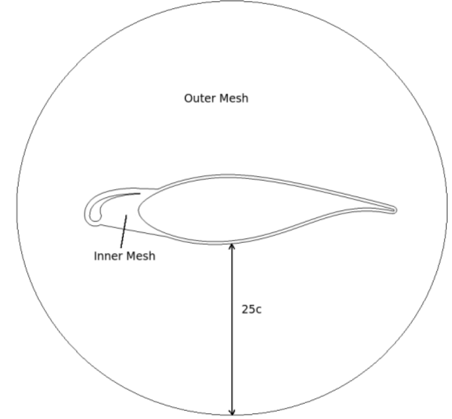


Fraunhofer IWES

Meshing of Wind Turbines







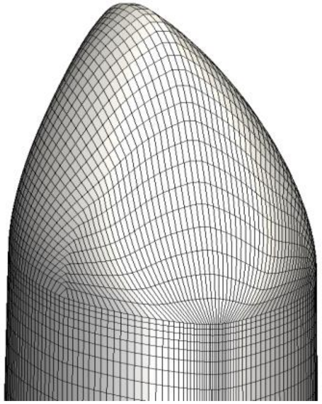
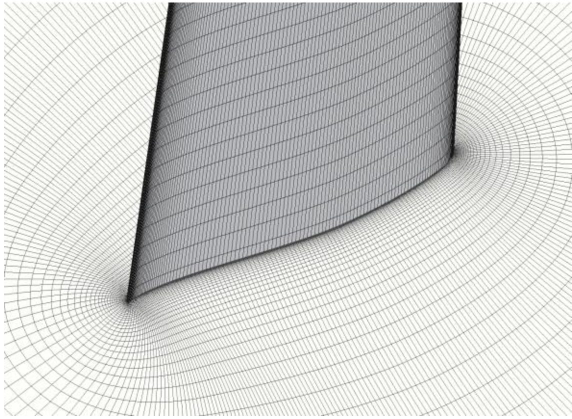
Construct2d+Converter

Fastest solution for 2D – Airfoil grids

- Input: airfoil geometry, Reynolds number, domain size
- Output: Hyperbolic or Elliptic Grid, Plot3D NASA Format + OpenFOAM Format
- Time needed: < 1 min
- Extension to: slat / flap / add-ons

Schramm, M.: Entwicklung und Optimierung mehrteiliger Profile. IWES Report. SmartBlades Projekt. BMWi 0325601-B (2016)
Rezazadeh, H.: 2D-Simulations of Flows over Airfoils with Fixed and Movable Leading Edge Slats. Master’s Thesis. University of Oldenburg (2016)

Fraunhofer IWES

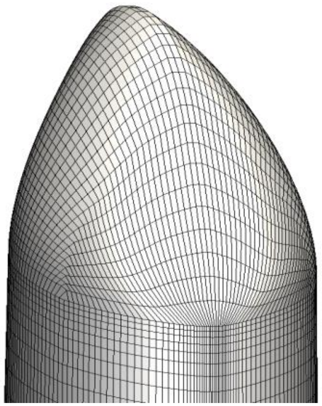
BladeBlockMesher

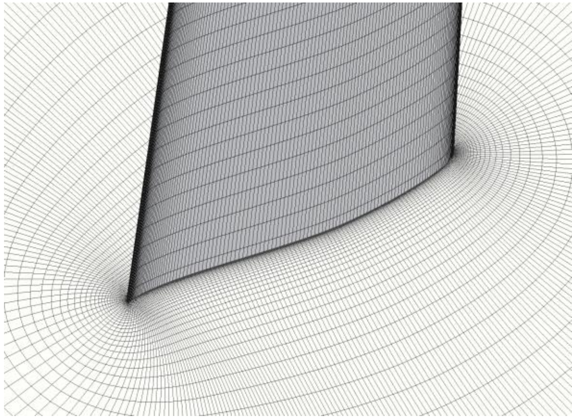
Simplest way from design to CFD

- Input: a BEM-like table / Output: OpenFOAM Format
- Time needed: *output file* < 2 min (in parallel) / *grid* < 10 min
- Extension to: tip shapes / add-ons

Rahimi, H., Daniele, E., Stoevesandt, B., Peinke, J.: Development and application of a grid generation tool for aerodynamic simulations of wind turbines. Wind Engineering, 40(2), 148-172 (2016) doi: 10.1177/0309524X16636318

Fraunhofer IWES





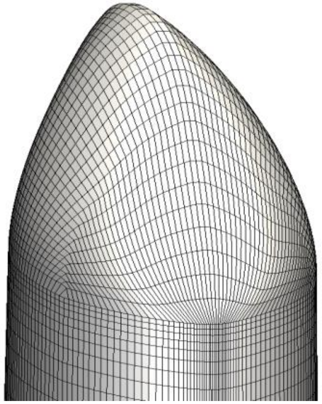
BladeBlockMesher

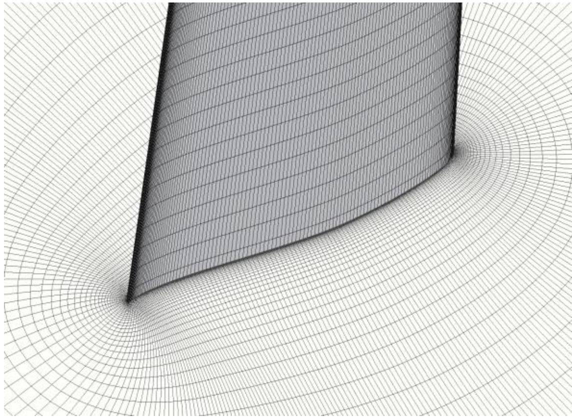
Input: a BEM-like table:

- section spanwise position, r (m)
- section chord length, c (m)
- section twist angle, θ (deg)
- section alignment point, xa/c
- section airfoil coordinate file
- section Pre Bending and Swept

- **The input file can be also be generated automatically from CAD files**

Fraunhofer IWES

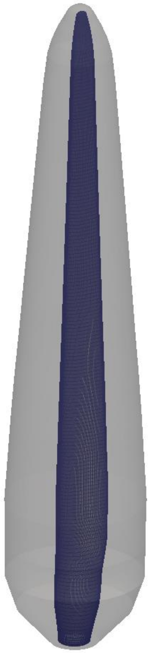




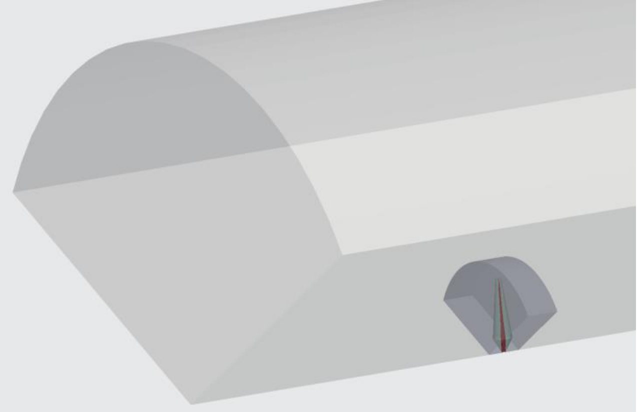
BladeBlockMesher

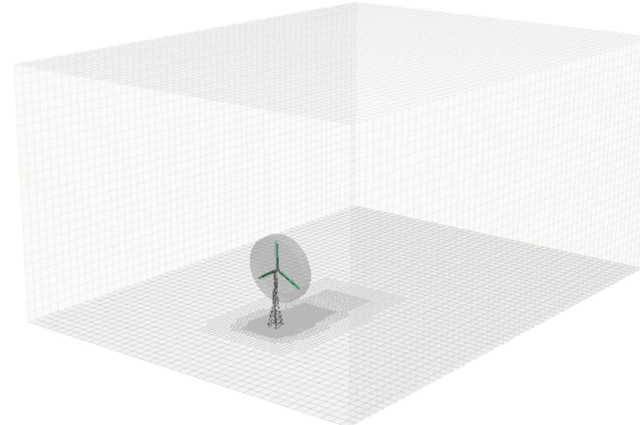
Output:

- Cylindrical grid around the blade
- Fully structured grid
- Smoothing with Hyperbolic or Elliptic methods
- Tip shapes: rounded or flat



Fraunhofer IWES



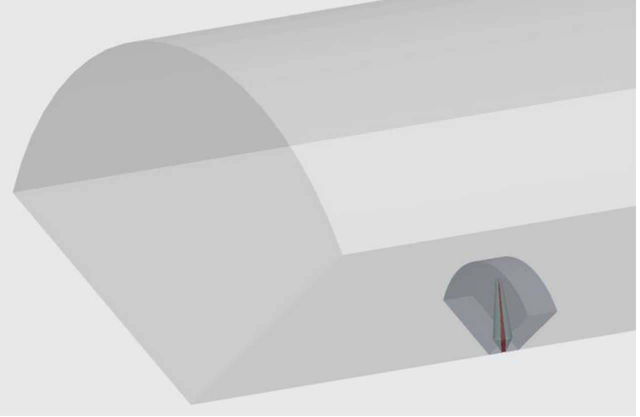


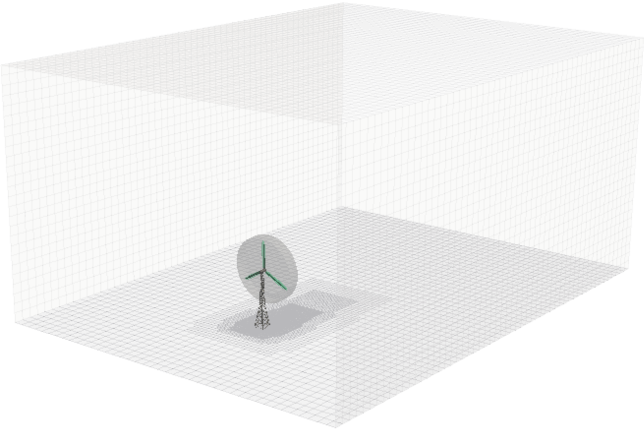
WindTurbineMesher

Automated meshing of complete wind turbines

- Input: Blade mesh from BladeBlockMesher or any other code/ Output: Ready-to-use-mesh
- Automated generation of secondary geometries (hub, tower ...)
- Automated wake refinements + setup of boundary conditions / solver settings
- Time needed: *output file* < 30 to 90 min (in parallel)

Rahimi, H., Dose, B., Stoevesandt, B., : Development and application of a grid generation tool for aerodynamic simulations of wind turbines. Inhouse report.

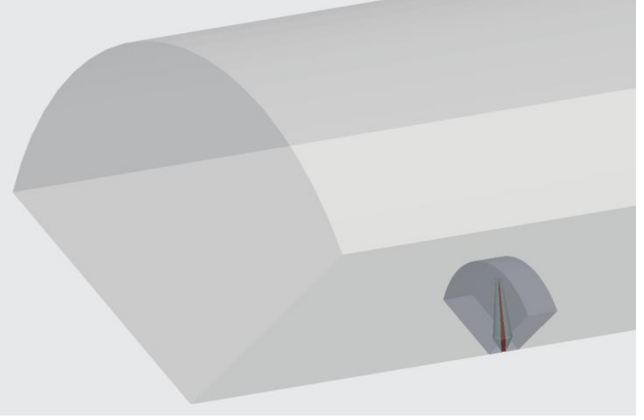


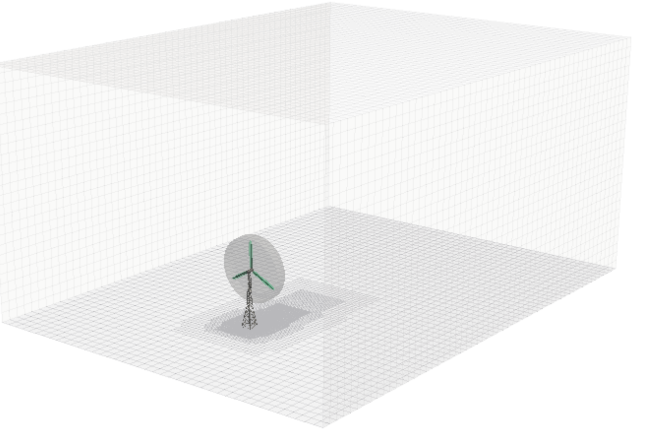


WindTurbineMesher

windTurbineMesher -> C++ based code

- Heavy use of OF tools like transformPoints, mergeMesh etc
- Paraview is used to generate geometries (hub, tower)
- snappyHexMesh is the main mesh application used
- Required dictionaries are included in windTurbineMesher





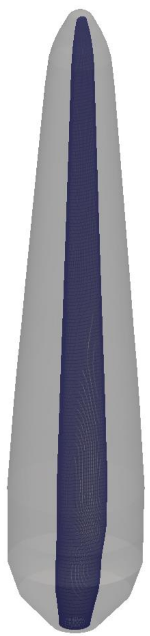
WindTurbineMesher

Concept of windTurbineMesher

- windTurbineMesher uses 4 different steps / modules
  - Step 1: Preparation of blade meshes
  - Step 2: Generation of rotor mesh
  - Step 3: Generation of farfield mesh
  - Step 4: Final mesh assembly (Merging, zero folder etc)
- Steps can be skipped

Fraunhofer IWES

Concept of windTurbineMesher

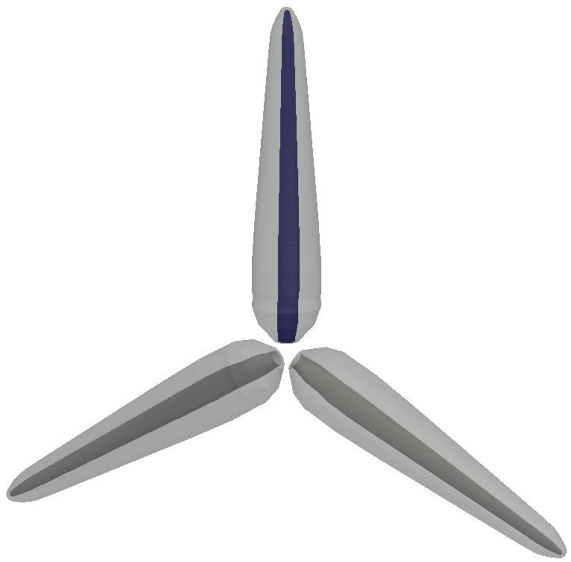


Step 1 - Preparation of blade meshes

- One blade mesh has to be provided
- Blade mesh generation not integrated into windTurbineMesher
- Flexbility -> Different meshing tools can be used to generate blade mesh

Step 1 - Output

- 1-3 blades
- Renamed patches
- Blade can be mirrored

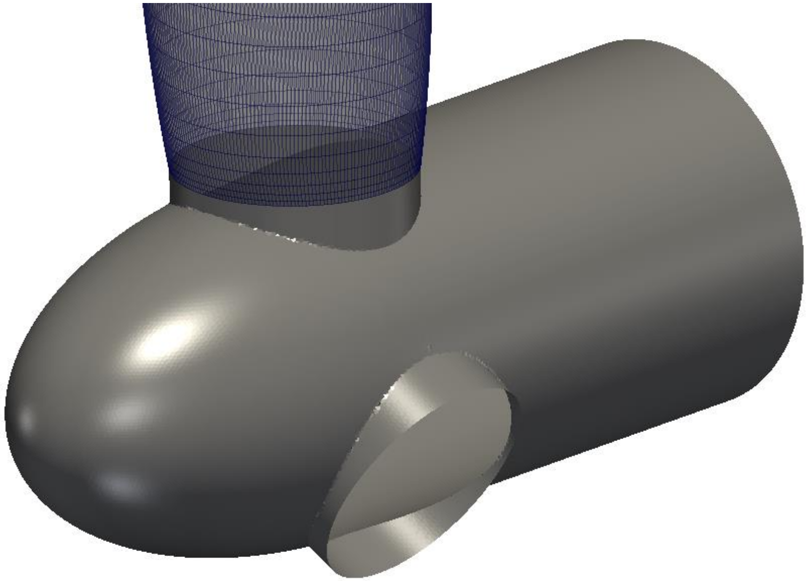


Fraunhofer IWES

Concept of windTurbineMesher

Step 2 - Generation of rotor mesh

- Get blade meshes
- Create hub geometry
  - Three hub types integrated
  - Hub geometry can also be provided
- Create rotor disk stl
- Run snappyHexMesh



Fraunhofer IWES

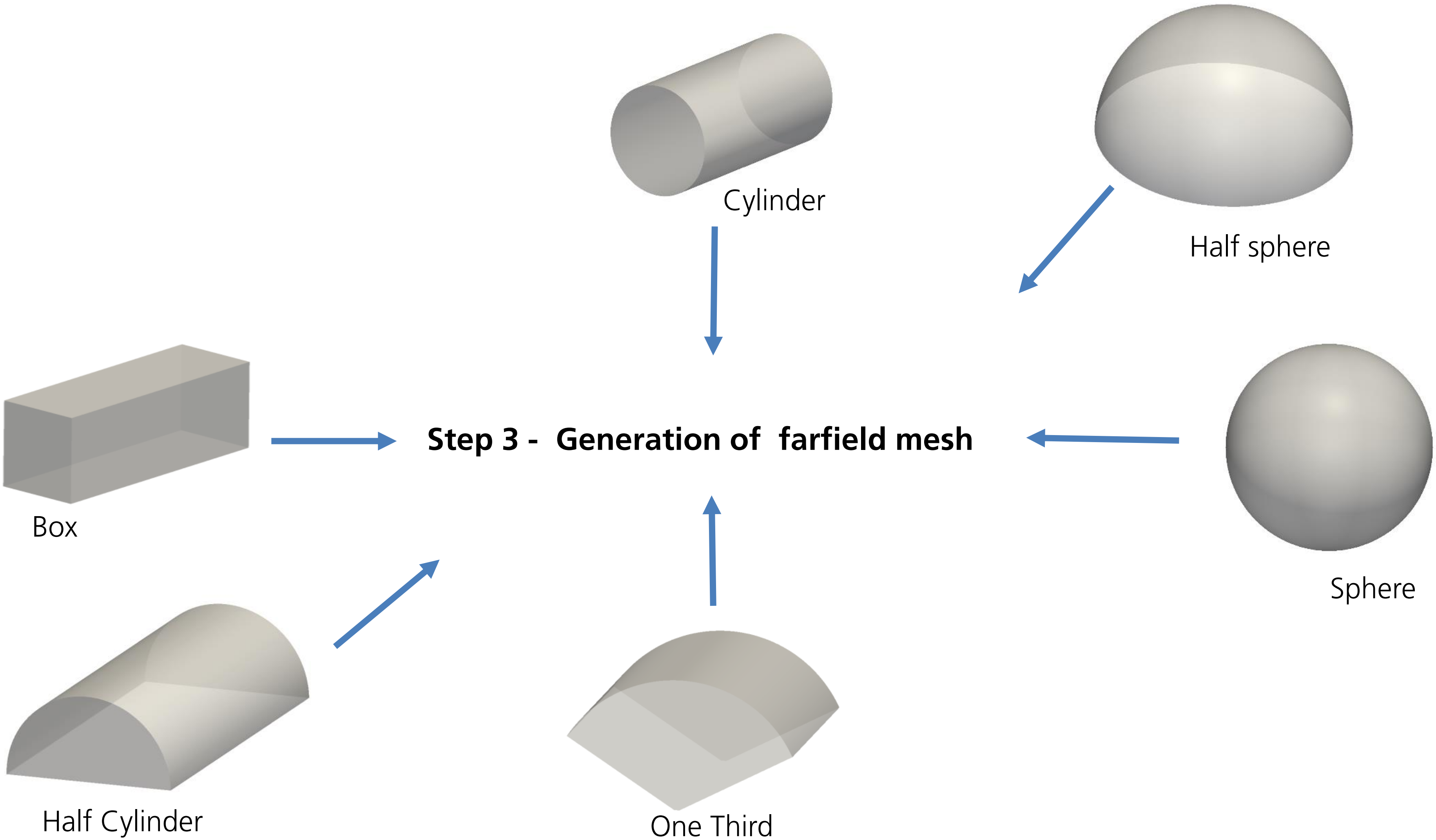
Concept of windTurbineMesher

Step 3 - Generation of farfield mesh

- Add rotor tilt
- Cone angle can be included

Fraunhofer IWES

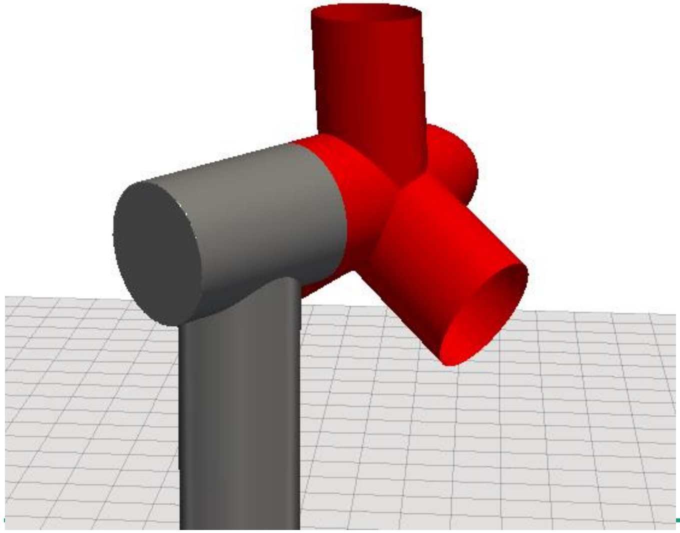
Concept of windTurbineMesher

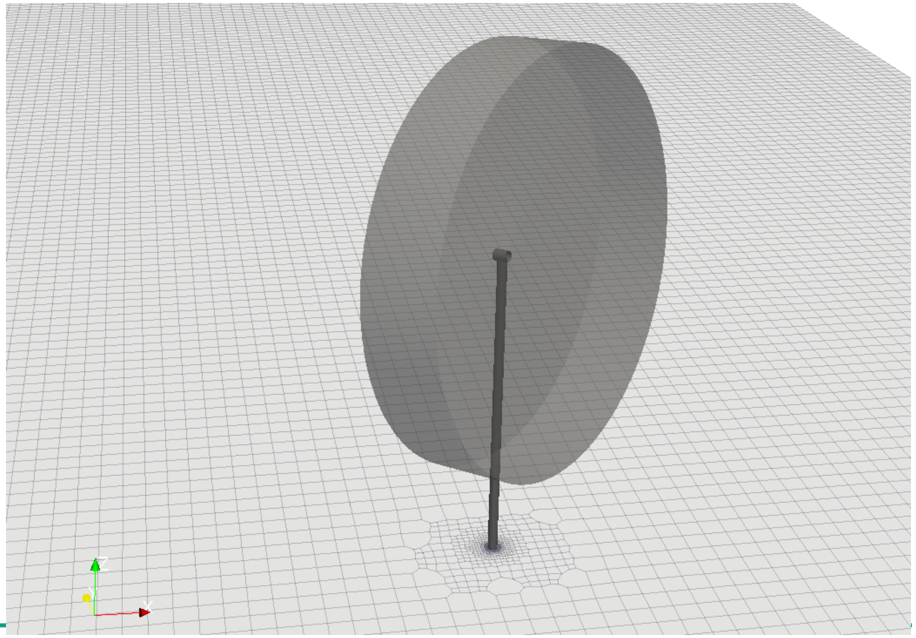


Concept of windTurbineMesher

Step 3 - Generation of farfield mesh

- Generation of tower/nacelle geometry
  - Different type of Nacelle geometry can be imported or selected
- Use snappyHexMesh to generate farfield mesh
- Add yaw angle



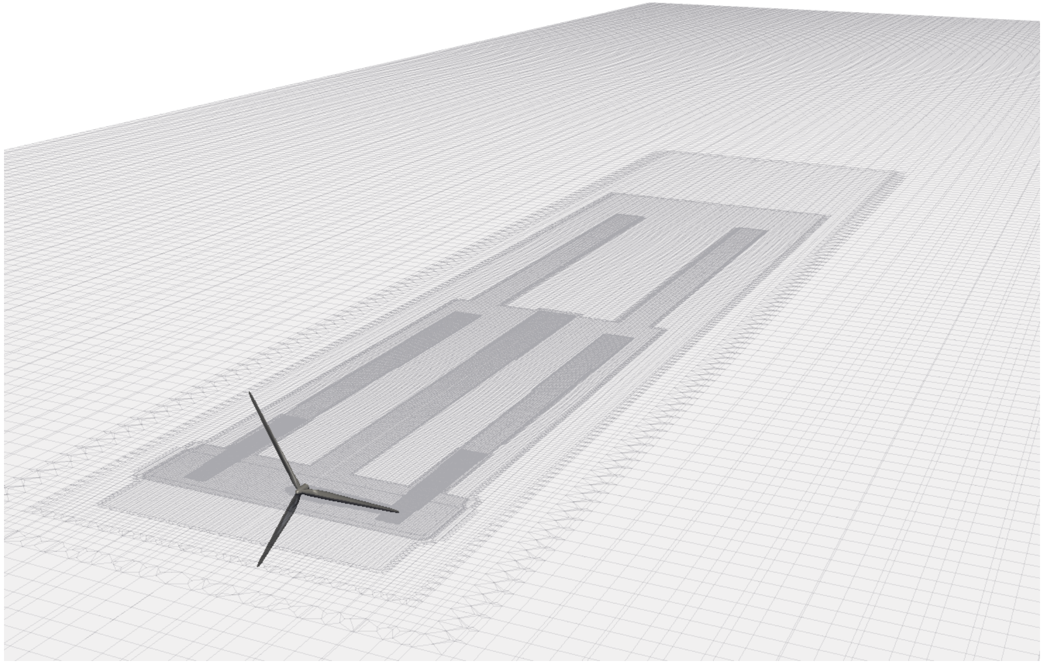


Fraunhofer IWES

Concept of windTurbineMesher

Step 3 - Generation of farfield mesh

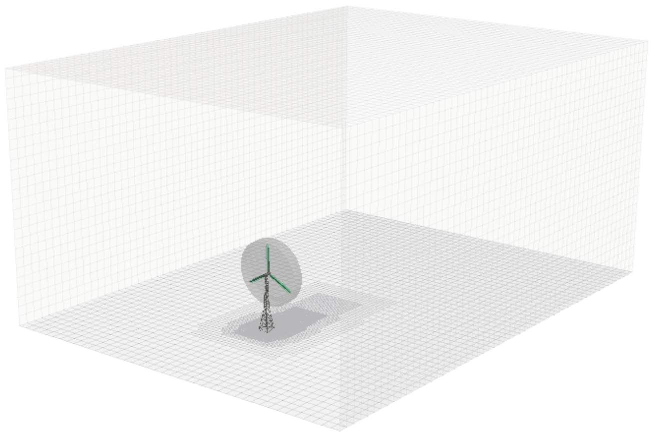
- Wake refinements



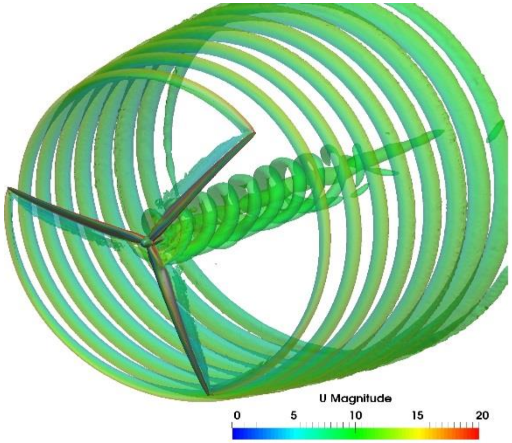
Concept of windTurbineMesher

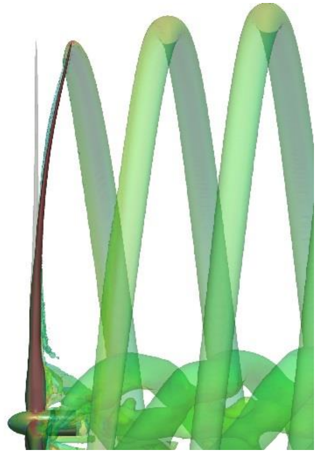
Step 4 - Final mesh assemby

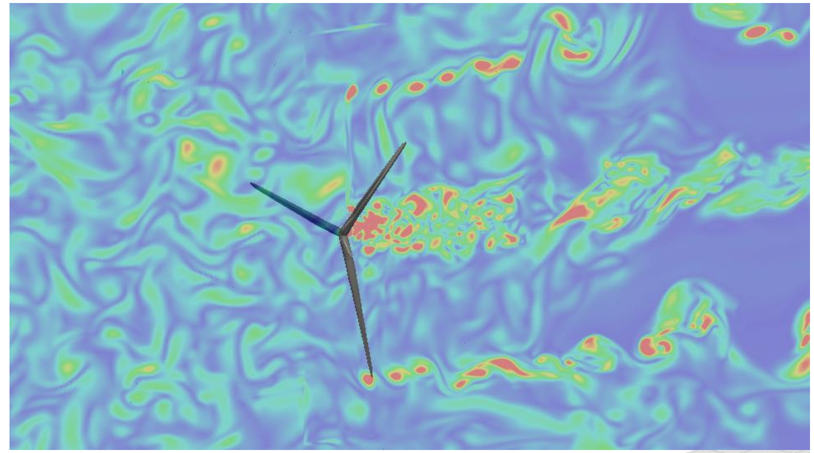
- Merge rotor and farfield mesh (Step 2+3)
- Create 0-Folder
- Setup boundary conditions
- Run checkMesh
- Run foamToVTK

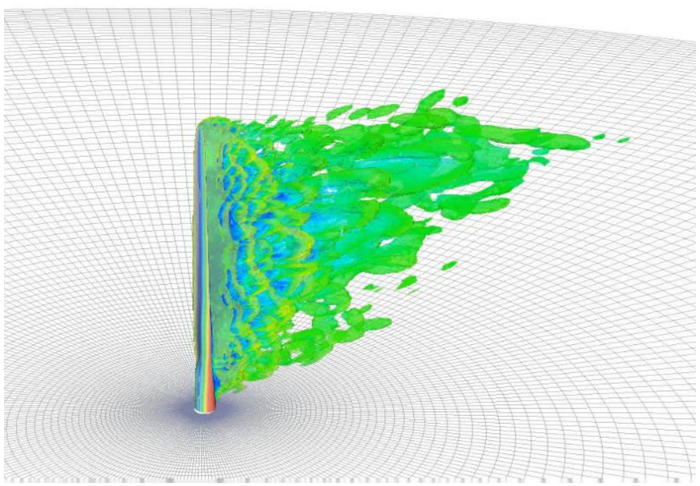


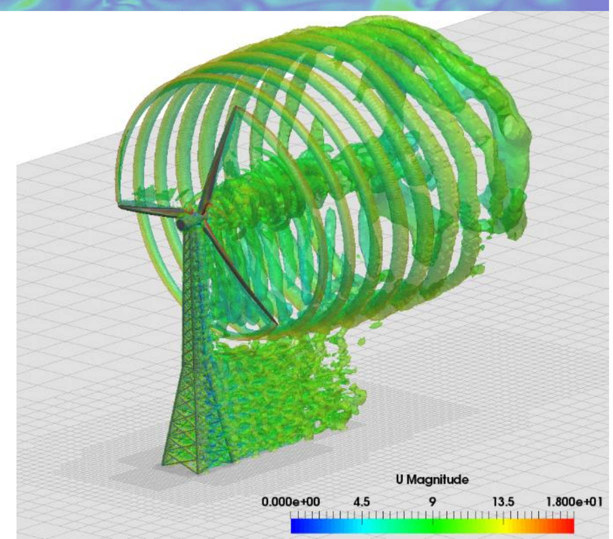
Conducted simulations used for complex cases









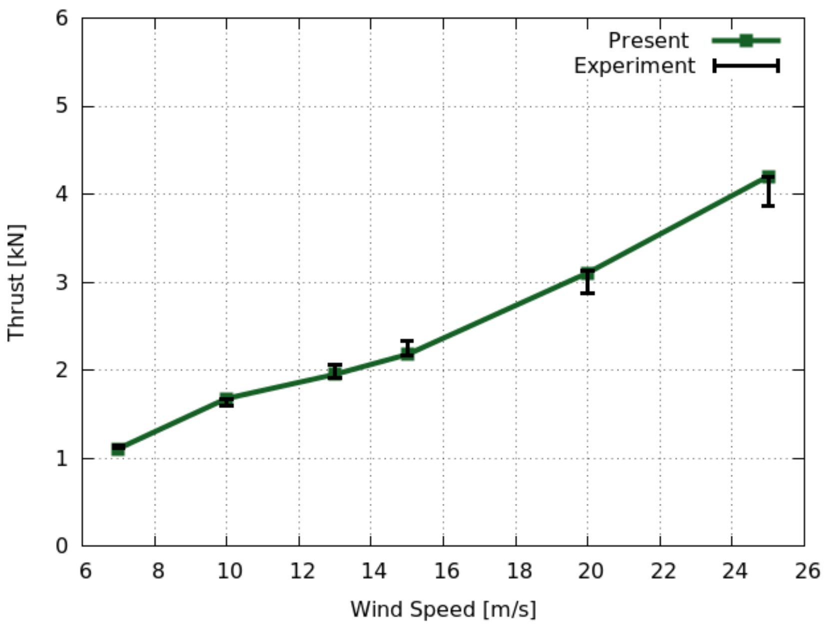


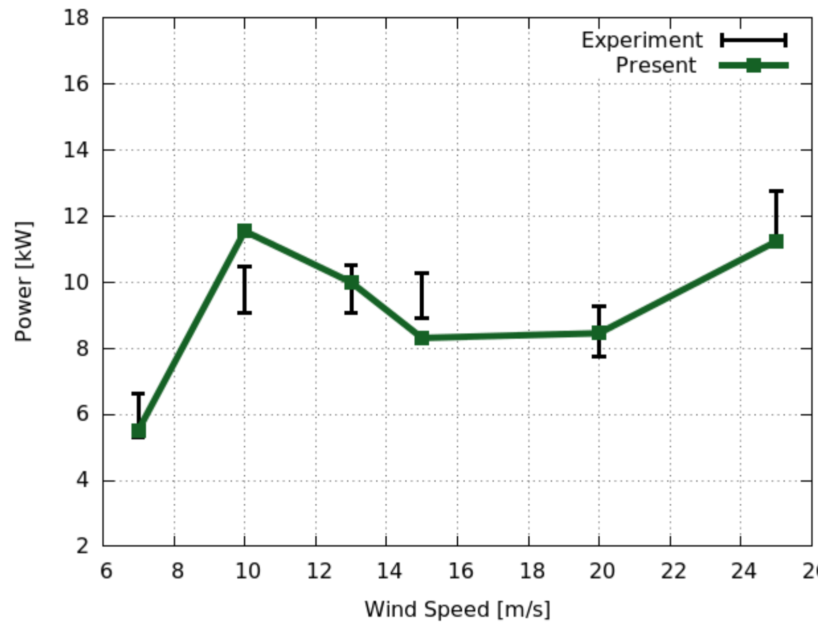
Fraunhofer IWES

Conducted simulations : NREL VI turbine

Validation through experiments

- 10m rotor diameter, stall regulated turbine
- Upwind and downwind measurements in NASA wind tunnel
- Pressure, loads for different sections as experimental data available

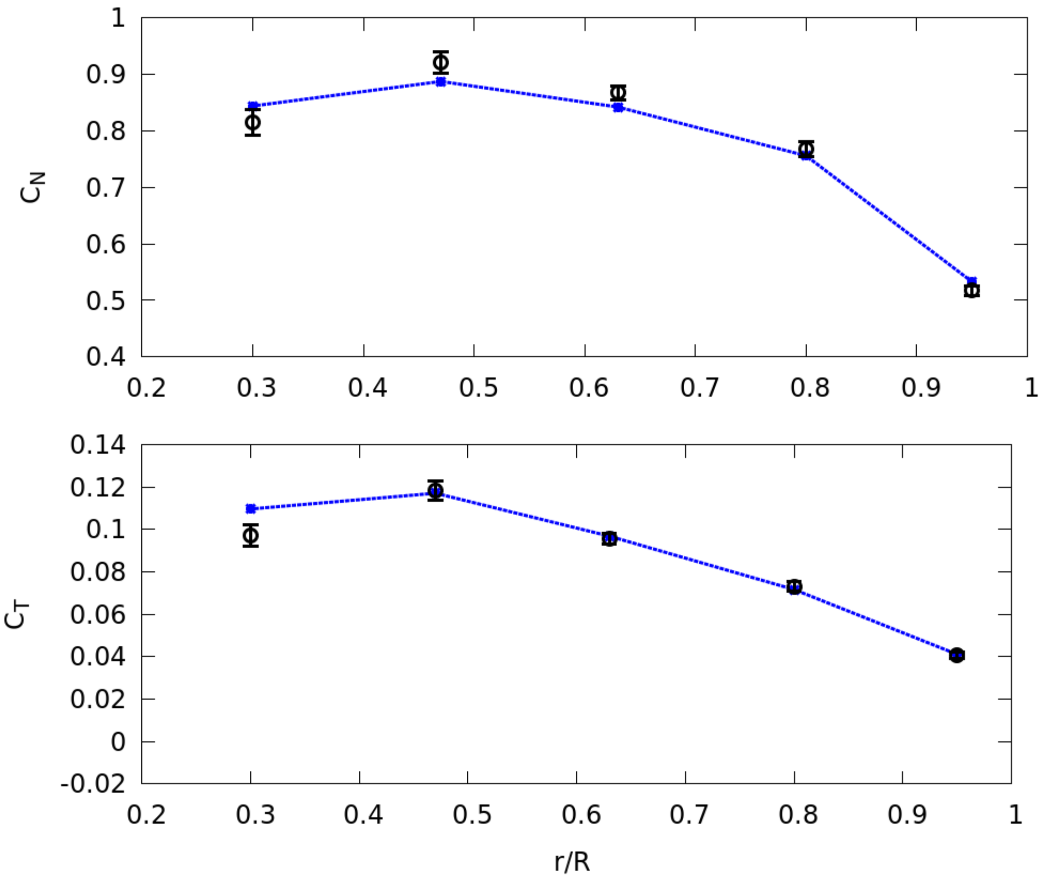




Rahimi, H., Medjroubi, W., Stoevesandt, B. and Peinke, J. (in press) Progress in Computational Fluid Dynamics, ‘Navier-Stokes-based predictions of the aerodynamic behaviour of stall regulated wind turbines using OpenFOAM’,

Conducted simulations : NREL VI turbine

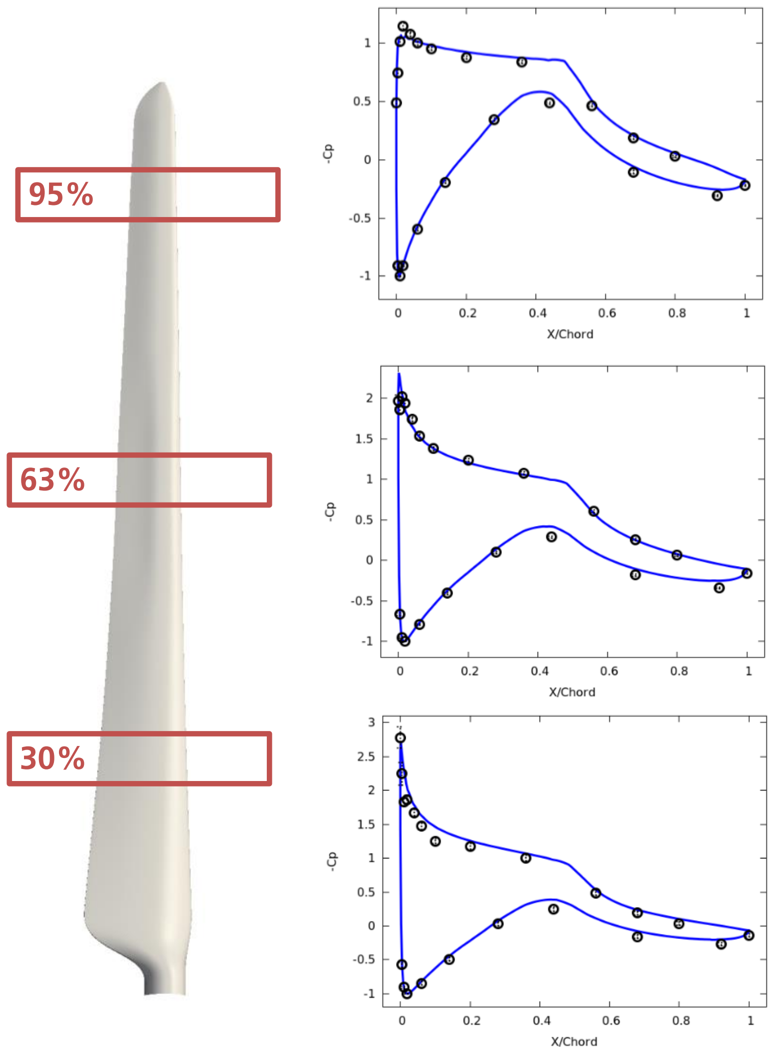
Sectional Forces



Fraunhofer IWES

Conducted simulations : NREL VI turbine

Sectional pressure data



Fraunhofer IWES

Acknowledgements
Fraunhofer IWES is funded by the:

Federal Republic of Germany

Federal Ministry for Economic Affairs and Energy

Federal Ministry of Education and Research

European Regional Development Fund (ERDF):

Federal State of Bremen

- Senator of Civil Engineering, Environment and Transportation
- Senator of Economy, Labor and Ports
- Senator of Science, Health and Consumer Protection
- Bremerhavener Gesellschaft für Investitions-Förderung und Stadtentwicklung GmbH

Federal State of Lower Saxony

Free and Hanseatic City of Hamburg

Federal Ministry for Economic Affairs and Energy

Federal Ministry of Education and Research

European Union
Investing in Bremen's Future
European Regional Development Fund

Bremen

Bremerhaven

Europa fördert Niedersachsen

Lower Saxony

Hamburg

Fraunhofer IWES

This is already the end… ☹

We hope that these slides are helpful!

In case of any questions, please don’t hesitate to contact us.

bernhard.stoevesandt@iwes.fraunhofer.de

Fraunhofer IWES