

A preCICE-OpenFAST adapter to couple OpenFAST to CFD simulation tools

Leonard Willeke
Delft University of Technology

- Flow dynamics in wind parks are important for many engineering problems from optimized control to load reduction
- High-fidelity flow computation, eg in OpenFOAM, is desirable to understand the farm flow dynamic
- Low-fidelity simulation of the wind turbines is often sufficient
- OpenFAST allows to simulate one or multiple turbines in low fidelity
- I present a preCICE-OpenFAST adapter to couple OpenFAST to the coupling library preCICE
- preCICE couples simulation programs in a black-box manner to achieve multi-physics simulations
- The adapter serves as driver code for OpenFAST to steer the simulation and calls preCICE for the data exchange
- As a first proof of concept, a coupling to OpenFOAM with limited capabilities is presented

I. Introduction

The realistic representation of the flow around wind turbines and larger wind parks can have a valuable impact on many fields. In engineering, it enables the optimization of turbine design and control to minimize structural loads and maximize power outputs. Economic assessments of the cost of wind energy production rely on a thorough understanding of how much energy a wind park will produce, and how often load-induced maintenance and replacements are necessary. Policy makers and planners are interested in the local environmental effects from acoustics to microclimates to regulate the placement of wind parks effectively.

To gain understanding of the aerodynamic effects of wind turbines, simulation tools are invaluable as they allow to study numerous conditions and arrangements without the need of complex experiments. However, the simulation of fluid dynamics at the scale of wind turbines or whole wind parks comes with a considerable trade-off. Detailed, high-fidelity simulations may capture the physics of the problem accurately, but are computationally very expensive. An alternative can be found in low-fidelity engineering models which model the physical phenomena in a simplified way to represent the most important aspects of the problem at hand. A popular engineering model for the fluid-structure interaction of wind turbines is the Actuator Line Method (ALM). The turbine tower and blades are represented as slender beams or lines, reducing the number of computational nodes. Nevertheless, it allows the detailed study of blade deformations and wake effects. A widely-used tool implementing the Actuator Line Method is the wind turbine simulation software OpenFAST¹. Coupling OpenFAST with a high-fidelity CFD solver allows to accurately solve the wind field at moderate computational cost. The CFD program computes a detailed large-scale flow field and exchanges the local flow velocity with OpenFAST (see Fig. 4). OpenFAST computes the turbine dynamics and sends back forces

¹<https://openfast.readthedocs.io/en/main/>

and deflections which are imposed on the flow field. The challenge in this setup lies in implementing the data exchange and mapping while ensuring numerical robustness of the simulation.

One solution to this problem is the open-source tool preCICE. preCICE is a coupling library for multi-physics simulations and allows to couple different programs to perform a partitioned simulation. It takes care of the communication and data mapping between the tools and implements different coupling schemes. To connect a program to preCICE, an additional piece of software called adapter is necessary. Once a simulation tool is connected to preCICE, it can be coupled to any other program in the preCICE environment, reducing the time-to-solution. The main idea of this work is therefore to develop a preCICE-OpenFAST adapter to couple OpenFAST to CFD simulation tools via preCICE.

To provide context, chapter II provides a literature review on the existing couplings between OpenFAST and several CFD solvers. The software packages used as well as the concept of the newly developed module are then described in Chapter III. Chapter IV outlines two test cases on which the software is applied, followed by a description of the development challenges the software still faces to reach maturity in chapter V. The final chapter VI concludes on the current capabilities and future possibilities of this approach.

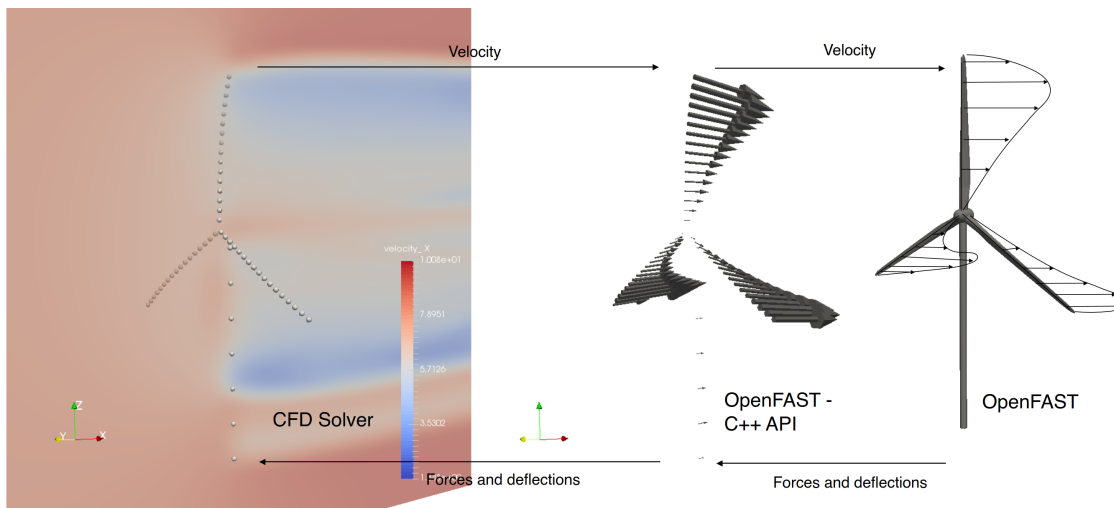


Figure 1. Coupling of OpenFAST to a CFD program. The fluid solver computes the flow dynamics and sends velocity data to OpenFAST. OpenFAST simulates the turbine dynamics and sends back forces and deflections, which are imposed on the flow field. Source: OpenFAST documentation²

II. Literature Review

A recent advancement in the coupling of OpenFAST is the tool **AspFAST** [9]. Developed at the Delf University of Technology and the weather forecasting company Whiffle, it connects OpenFAST to the closed-license Large Eddy Simulation (LES) code GRASP. As a binder code, AspFAST serves as an intermediary between the GPU-driven GRASP and the CPU-driven OpenFAST. It accesses OpenFAST through a C++ API, essentially operating as a driver code. GRASP is accessed through a specialized API that handles the transition from CPU to GPU operations. The core functionalities of AspFAST include the data exchange and mapping between GRASP and OpenFAST. This is done for crucial data such as force, velocity, and the position of turbine blades. Notably, the code implements the mapping between the volume-based velocity grid of GRASP and

²https://ganesh-openfast.readthedocs.io/en/latest/_images/actuatorLine_illustrationViz.pdf(visited on 07.11.2023)

the actuator line model of OpenFAST. Because the coupling code³ is open-source, it can serve as a blueprint for anyone interested in coupling OpenFAST via the C++ interface. The software architecture of AspFAST is not far from our idea for the preCICE-OpenFAST adapter: AspFAST is a plug-in for the CFD solver GRASP while being the driver code for OpenFAST, serving as the "adapter" between the two. A big difference lies in the fact that AspFAST also has to implement the mapping algorithms, which are executed by preCICE in our approach.

The **ExaWind** project [8] is based on the open-source, massively parallel and incompressible flow solver NaluWind [1]. It is a wind-focused adaption of the general flow solver NaluCFD and strives to deliver flow simulations in exascale. Hence, great emphasis lies on the scalability and simulation speed of the software through advanced solver and coupling algorithms as well as the use of GPUs. The coupling to OpenFAST is implemented inside NaluWind and not as additional plug-in. An extensive documentation with insights into the theory and verification is given.

Another coupling based on solver modification is the Simulator for Wind Farm Applications or **SOWFA** [5] from NREL. It couples OpenFOAM to OpenFAST by providing an additional class *horizontalAxisALM* in OpenFOAM. It defines functions to call OpenFAST and enables the data exchange. This class can be included directly in any OpenFOAM solver by modifying the OpenFOAM source code, so that the solver calls OpenFAST during each time step.

A software solution relying on a similar coupling tool to preCICE is **foamFAST** [10]. This project uses the commercial coupling tool for partitioned multi-physics simulations MpCCI, developed and sold by the Fraunhofer SCAI. Similar to preCICE, MpCCI is connected to different programs via adapters. Once the adapter is written, you can couple your tool to any other tool in the MpCCI environment. A difference in software architecture is the fact that MpCCI acts as the master algorithm: It calls the solvers instead of being called by them like preCICE. During the project, a MpCCI adapter for OpenFAST was developed and an existing OpenFOAM adapter modified to perform the coupling between the two.

The reviewed literature shows two different concepts to undertake the coupling. The first idea is to **modify the CFD solver source code**, as the ExaWind project has done with the NaluWind solver and SOWFA with OpenFOAM. While convenient for developers who are familiar with the software, this approach can prove hard to maintain when the underlying solver software is updated.

The second concept leaves the CFD solvers intact and uses **an additional software layer** to connect the simulation tools. One way to implement this is the library approach where an additional module or library is called by the original CFD solver. An example is AspFAST, where the stand-alone plug-in code is maintained separately from the solver. The second way for implementation is the framework approach, used by the coupling tool MpCCI in foamFAST. Here, the additional software acts as a master algorithm calling the solver.

Given the previous work, why should we continue to develop a preCICE-OpenFAST adapter? Our approach combines some of the positive aspects of previous projects. First, preCICE follows the library approach, leaving the original programs untouched and enabling a fast time-to-solution. Using preCICE allows to rely on advanced and tested coupling algorithms without the need to re-implement them or buy a licensed software. Establishing a preCICE-OpenFAST adapter opens up the road to connect OpenFAST not only to OpenFOAM, but to any CFD solver coupled to preCICE. Furthermore, preCICE creates a clean and easy to use simulation environment and simplifies setting up and maintaining simulations.

Nevertheless, it should be noted that OpenFOAM is currently the only CFD solver connected to preCICE which can be used for a coupling with OpenFAST. Coupling any other tool, such as the presented GRASP or NaluCFD, will require the development of additional adapters.

³<https://gitlab.com/whiffle-public/aspfast> (visited on 20.12.2023)

III. Software description

A. Software components

1. *preCICE*

preCICE [3] is an open-source coupling library for partitioned multi-physics simulations. An overview of its concept is shown in Figure 2. In *preCICE* terminology, the coupled simulation programs are called *solvers* or *participants*. *preCICE* connects different solvers to perform a partitioned simulation. It takes care of different coupling aspects such as data mapping and communication. Different coupling schemes are possible [6] by the choice of which solvers are coupled, what data is exchanged and which coupling algorithms are used. Explicit and implicit coupling schemes are available, as well as the option to connect more than two participants via multi-coupling. In addition, *preCICE* provides acceleration algorithms to reach a decent computation time for large simulations and time interpolation for the coupling of solvers with different time steps. The coupling scheme is defined in the *precice-config.xml*, the only global file which is accessed by all participants.

For each solver, a specific *adapter* is necessary to communicate with *preCICE*. The adapter is an additional software layer which can take many forms, depending on the solver. For example, the OpenFOAM adapter is a C++ function object, an independent library, while the FEniCS adapter is a Python module. Additional language bindings in Fortran, Julia and Matlab allow the development of adapters in these languages as well.

Additional remarks: library approach, scalability, HPC

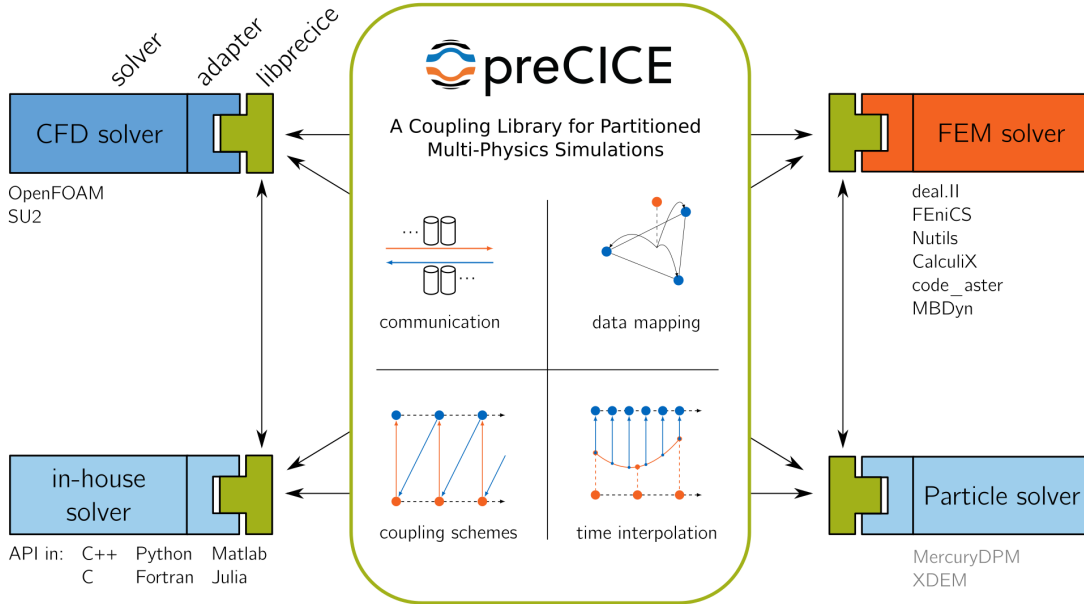


Figure 2. *preCICE* overview [3]

2. *OpenFOAM*

*OpenFOAM*⁴, an acronym for Open Field Operation and Manipulation, is a powerful open-source computational fluid dynamics (CFD) software package. This versatile tool provides a comprehensive suite of solvers and utilities for simulating and analyzing fluid flow, heat transfer, and related phenomena. Its open-source

⁴<https://www.openfoam.com/>

nature allows users to access, modify, and distribute the source code, fostering collaboration and innovation within the CFD community. Originally developed in the late 1980s, OpenFOAM has since evolved into a robust and widely-used simulation platform. It employs a finite volume method, making it particularly suitable for simulating complex fluid dynamics scenarios in various industries, including aerospace, automotive, energy, and environmental engineering.

A key strength of OpenFOAM lies in its flexibility and extensibility. Users can customize simulations by modifying the source code or by utilizing a wide range of pre-existing solvers and libraries. This adaptability makes OpenFOAM a preferred choice for researchers, engineers, and scientists seeking to tackle diverse fluid dynamics challenges. With a strong emphasis on parallel computing, OpenFOAM is capable of leveraging high-performance computing resources to accelerate simulations, enabling the analysis of large-scale and intricate fluid flow problems. Its user-friendly interface, coupled with extensive documentation and a supportive community, makes OpenFOAM an accessible tool for both beginners and experienced practitioners in the field of computational fluid dynamics.

3. preCICE-OpenFOAM Adapter

The preCICE-OpenFOAM adapter [4] is designed to facilitate robust and efficient coupling between the preCICE library and OpenFOAM to enable multi-physics simulations. At its core, the software architecture of the adapter follows a modular design, allowing for flexibility and customization in the coupling process. The adapter consists of three distinct modules tailored for fluid flow (FF), fluid-structure interaction (FSI), and conjugate heat transfer (CHT) coupling scenarios. Each module is optimized to handle the requirements of its respective physics domain.

In the context of fluid flow coupling, the adapter incorporates modules that leverage preCICE to enable seamless communication between different fluid solvers. This modular architecture empowers users to mix and match specialized fluid dynamics solvers as needed, while preCICE maintains a standardized interface for consistent coupling. The coupling of two solvers can be performed at a shared boundary surface or on a shared volume.

For fluid-structure interaction coupling, the architecture orchestrates the interaction between OpenFOAM's fluid dynamics solvers and external structural simulation codes. This involves careful coordination to ensure accurate and efficient data exchange, enabling comprehensive simulations of scenarios involving dynamic interactions between fluids and structures.

In the realm of conjugate heat transfer coupling, the adapter's architecture facilitates the exchange of thermal information between diverse solvers within OpenFOAM and external heat transfer codes. This functionality is crucial for scenarios where temperature fields influence fluid dynamics or structural behavior, allowing for a comprehensive understanding of phenomena such as electronics cooling or combustion.

The overall design emphasizes modularity, extensibility, and adherence to established standards, making the preCICE-OpenFOAM adapter an adaptable tool for a wide range of multi-physics simulations.

4. OpenFAST

OpenFAST⁵ is a versatile and open-source simulation tool developed for the analysis and design of wind turbines. This software, maintained by the National Renewable Energy Laboratory (NREL) in collaboration with the wind energy community, provides a comprehensive platform for simulating the aerodynamics, structural dynamics, and turbulence effects associated with wind turbine systems.

OpenFAST is designed to meet the evolving needs of the wind energy industry, offering a modular and extensible architecture that allows users to tailor simulations to specific requirements. Its capabilities span a range of applications, including load analysis, controller design, and overall turbine performance assessment.

⁵<https://openfast.readthedocs.io/en/main/>

The software incorporates advanced modeling techniques for aerodynamics, structural dynamics, and control systems, ensuring accurate representation of wind turbine behavior in various operating conditions. With a focus on high-performance computing, OpenFAST can leverage parallel processing to handle complex simulations efficiently, facilitating the analysis of large-scale wind turbine systems. The tool supports the simulation of various turbine configurations, including land-based and offshore turbines with fixed or floating platforms. This flexibility makes it a valuable tool for addressing the diverse challenges associated with wind energy projects.

On a software level, OpenFAST is a glue code for different modules which compute the various physical domains. For example, the separate module AeroDyn is used to compute the aerodynamics of a simulation case. Another module called ElastoDyn computes the structural response of the aerodynamic loads. Both tools are then coupled in runtime in the framework. OpenFAST can be interfaced with a C++ API to drive the simulation and couple the simulation to an external CFD tool.

B. Software concept

The main idea of this work is to develop a preCICE-OpenFAST adapter, a new piece of software that connects OpenFAST and preCICE. It acts as a driver code towards OpenFAST, leveraging the C++ API of the simulation program. Simultaneously, it calls preCICE to communicate data and receive steering commands for the simulation. The software is configurable on runtime by two *.yaml* input files. *preciceInput.yaml* contains information about the coupling setup such as which variables are exchanged, which meshes are used and where the global *precice-config.xml* can be found. *openfastInput.yaml* adds metadata for the wind turbine simulation and points towards the OpenFAST *.fst* file.

The software concept is presented in simplified form in Figure 3. First, the header files for important libraries are included and variables for data handling are created (lines 1-5). The script then utilizes the C++ API of OpenFAST and preCICE to instantiate and initialize both tools (lines 8-14). The *FAST* object allows the access of OpenFAST while the *participant* object is connected to preCICE. The coupling library controls the main time loop (line 1628) which performs the coupled simulation. First, velocity data is retrieved from the CFD solver and stored locally (line 18). The information is then passed on to OpenFAST (line 20) and updated on the internal meshes of the simulation tool. OpenFAST is called to compute the wind turbine dynamics for the next time step (line 22). The resulting blade and tower force data is extracted and stored (line 24). The updated force data is passed on to the CFD solver (line 26), after which both solvers advance in time (line 28). This loop is repeated until the simulation time is completed. Finally, preCICE and OpenFAST are destructed (lines 30-31). The software is developed and documented on GitHub⁶.

IV. Example cases

To drive the adapter development and test the software at its current state, two example cases were implemented. The OpenFAST simulation of a single wind turbine, namely the NREL 5MW model [7], is coupled. The first case implements the coupling to a dummy fluid solver. While performing no realistic computations, the dummy is handy to get an insight into the different data structures, mesh setups, and mapping options. The second case couples OpenFAST to the CFD program OpenFOAM. It can be seen as a first proof-of-principle with a successful data exchange via preCICE. However, more work needs to be done to ensure a correct and robust simulation. Both cases are part of the repository on GitHub⁷.

⁶<https://github.com/LeonardWilleke/openfast-adapter>

⁷<https://github.com/LeonardWilleke/openfast-adapter/tree/main/cases>

```

1      #include <OpenFAST.H>
2      #include <precice.hpp>
3
4      vector<double> readData;
5      vector<double> writeData;
6
7      // OpenFAST Setup
8      fast::OpenFAST FAST;
9      ...
10     FAST.init()
11     // preCICE Setup
12     participant = precice.participant(...);
13     ...
14     participant.initialize();
15     // main time loop
16     while participant.isCouplingOngoing():
17         // Get read data from preCICE
18         participant.readData(readData, ...);
19         // Set read data in OpenFAST
20         FAST.setVelocity(readData, ...);
21         // Compute next time step
22         FAST.step();
23         // Get write data from OpenFAST
24         FAST.getForce(writeData, ...);
25         // Send write data to preCICE
26         participant.writeData(writeData, ...);
27         // Advance preCICE in time
28         participant.advance(dt)
29
30     participant.finalize()
31     FAST.end()

```

Figure 3. Concept of the OpenFAST adapter: The script utilizes the C++ API to execute OpenFAST and calls preCICE to couple the simulation. For conciseness, API calls are simplified.

A. Coupling OpenFAST with a dummy CFD solver

The motivation for this case is to provide a simple and fast way to test different functionalities and gain insight, not to perform realistic simulations. OpenFAST computes a NREL 5MW turbine with a fixed rotor to avoid mesh problems on the moving blades. The fluid solver is implemented as a simple C++ script. It creates a mesh that can be used to map data from OpenFAST and write constant values back. The dummy does not perform any calculations. To perform the coupling, the script calls preCICE. This case allows to explore the mapping and data exchange in a minimal setup.

OpenFAST employs two internal meshes of the turbine surface with different vertices. Force data is stored on one mesh, while flow velocity is stored on another. Both meshes can be accessed by the C++ API. However, it is also possible to exchange both variables on the velocity mesh only and call an internal mapping function in OpenFAST to map the force data to the force mesh. Inspired by the coupling setup in [9], I want to access both meshes and use preCICE for the mapping. This results in the elaborous coupling scheme seen

in Figure 4. In total, three meshes are employed: The Fluid solver has one mesh *Fluid-Mesh* used for both the force and velocity data while the Solid solver (OpenFAST) uses two meshes. *Solid-Mesh-Velocity* is the velocity mesh of OpenFAST, on which the velocity data from the Fluid solver is mapped. *Solid-Mesh-Force* is the force mesh of OpenFAST, from which force data is mapped back to the Fluid solver.

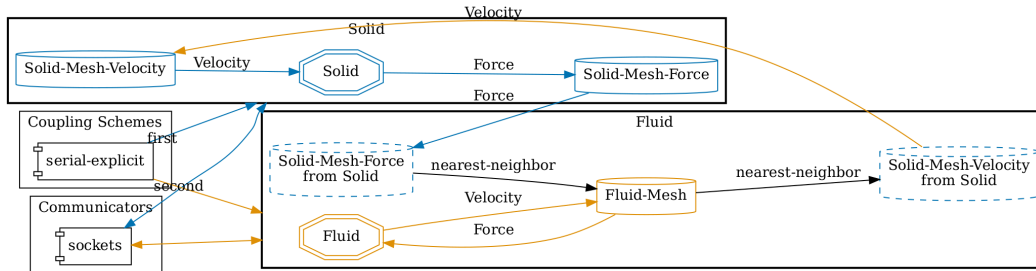


Figure 4. Coupling scheme between OpenFAST, named Solid, and the dummy Fluid solver. The figure was created from the *precice-config.xml* using the *config visualizer tool*⁸.

B. Coupling OpenFAST with OpenFOAM

The second example case implements the coupling of OpenFAST to the CFD tool OpenFOAM. It employs the same coupling scheme and mesh use as the previous case. However, the Fluid mesh is now more complex. The main obstacle for a successful coupling is

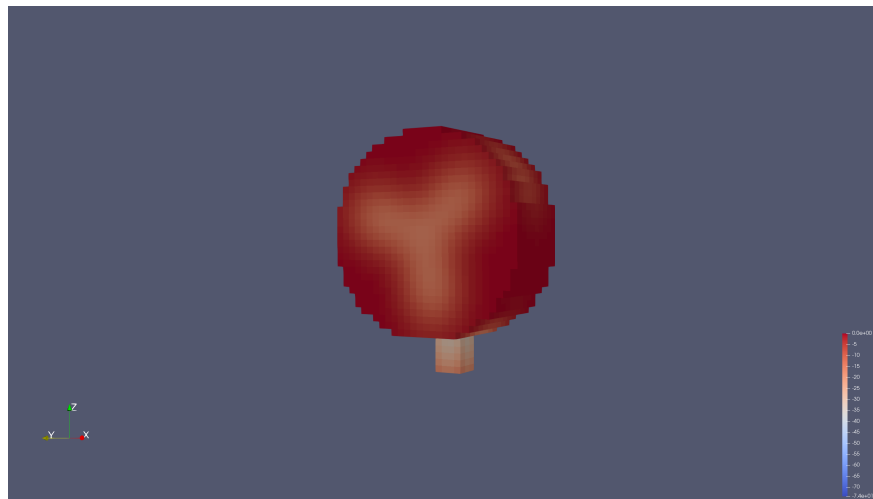
Case cfd-turbine

- Explain the setup
 - Couple OpenFAST to OpenFOAM
 - The coupling scheme and mesh use is identical to the dummy
 - Now we are dealing with a different mesh on the fluid solver
 - Main obstacle: The OpenFOAM adapter is not designed to implement a coupling with a solver using the actuator line method. How to map between the line mesh in OpenFAST (Figure 5a) and the volume mesh in OpenFOAM (Figure 5b)?
- Explain how the current setup works
 - Define a cellSet inside the OpenFOAM domain (Figure 5b) to which OpenFAST is coupled
 - Write the rotor and tower data to this subdomain
 - Read velocity data from this subdomain
 - The mapping from line to volume and back is done by preCICE, but probably wrong
- How should the turbine be represented in OpenFOAM?
 - Volume mesh / cellSet
 - Surface mesh / patch
 - ALM implementation (eg with turbinesFoam)

⁸<https://precice.org/tooling-config-visualization.html>



(a) Line representation of the turbine in OpenFAST



(b) Volume representation of the flow field immediately around the turbine in OpenFOAM

Figure 5. Mesh differences between OpenFAST and the CFD solver OpenFOAM. OpenFAST uses the Actuator Line Model to represent blades and towers in the flow field, while OpenFOAM calculates the whole flow field.

V. Challenges

Current limitations of the adapter

- Only for one turbine
- Only for onshore turbine
- Only proof of concept: Data is exchanged in a sensible way, no knowledge about the physical correctness

What are the next steps in the coupling of OpenFAST via preCICE?

Task 1: Mapping turbine data between actuator lines and volume meshes

- Problem: The OpenFOAM adapter is not designed for the implementation of ALM

- We need to get velocity from a volume section → possible
- We need to set force or pressure gradient on a volume section → not possible
- Solution 1: Do the mapping in the OpenFAST adapter
 - Something very similar has been done by AspFAST (LINK)
 - Includes smearing of the actuator data which is good
 - Possible Problem: I add volume data to the mesh in the OpenFAST adapter that the OpenFOAM adapter is not able to write to FOAM afterwards → think about
- Solution 2: Modify the OpenFOAM adapter
 - Adapt the PressureGradient.C class of the FF module
 - Include code to write pressure gradient on a cellset
 - Take structure from Pressure.C class
 - Problem: How to write the pressure gradient on a field in OpenFOAM?
 - Pressure gradient seems to be the boundary condition of the velocity field U → this is settable
 - Requires the correct import of the velocity field and the correct retrieval of its boundary field → give the commands you know
 - Not sure if you can also set the pressure gradient on the pressure field itself but don't think so
 - Open: How to do smearing if necessary

Task 2: Bring the OpenFAST adapter to maturity

- Write a regression test (using the dummy fluid solver)
- Create a first test case with documentation
- Verify simulation results against simulations done with AspFAST and other tools ([9] gives some benchmark cases)

Task 3: Enable coupling with multiple turbines

- OpenFAST C++ API allows to run multiple instances of OpenFAST for wind park scenarios
- How do you connect them to preCICE?
- How do you define the different coupling scenarios?
- Possibly use the MacroMicro manager of preCICE to deal with the coupling of multiple domains

Task 4: Explore coupling scenarios with other CFD solver

- Are there currently other CFD solvers coupled to preCICE that would be interesting?
- Otherwise interesting candidates are: YALES2, GRASP, NaluWind
- Most of them have native coupling tools for OpenFAST already

Additional remarks

- The OpenFOAM library turbinesFoam⁹ [2] might be useful. It implements the actuator line method with different solvers like pimpleFoam in OpenFOAM. The solvers are modified to perform the ALM computation. It is not clear yet how this could be of use, as we want to perform the ALM computation of the turbine to OpenFAST and map the results to OpenFOAM, not do the whole computation in OpenFOAM.

VI. Conclusion

A first preCICE-OpenFAST adapter to couple the wind energy engineering tool to CFD solvers was presented in this work. The coupling of an OpenFAST simulation of a single wind turbine with a dummy solver and OpenFOAM was discussed. Although a proof of concept was achieved, showing that preCICE can be used

⁹<https://github.com/turbinesFoam/turbinesFoam> (visited 14.12.2023)

for the coupling to OpenFOAM in principle, some challenging tasks remain. The coupling of OpenFAST to an arbitrary CFD solver was not discussed and poses more questions. This work may serve as a starting point. The presented preCICE-OpenFAST adapter has the potential to be developed into a viable open-source alternative for the coupling of OpenFAST to different CFD solvers.

Acknowledgments

I am grateful to Prof. Axelle Viré who accepted me as a visiting researcher and made this work possible. Many thanks to her and Evert Ewald for the discussions and hints during our meetings. Extended thanks to Benjamin Uekermann, who enabled my visit at TU Delft in the first place and provided feedback along the way. Lastly, I want to thank the wonderful crowd of PhD researchers in the wind energy department who made my time in Delft memorable.

References

- [1] S. Ananthan et al. *Nalu-Wind and OpenFAST: A high-fidelity modeling and simulation environment for wind energy*. SAND2019-3687R. 2019.
- [2] P. Bachant, A. Goude, and M. Wosnik. “Actuator line modeling of vertical-axis turbines”. In: *arXiv* (published online 22 Sep. 2018).
- [3] G Chourdakis et al. “preCICE v2: A sustainable and user-friendly coupling library [version 2; peer review: 2 approved]”. In: *Open Research Europe* 2.51 (2022). doi: 10.12688/openreseurope.14445.2.
- [4] Gerasimos Chourdakis, David Schneider, and Benjamin Uekermann. “OpenFOAM-preCICE: Coupling OpenFOAM with external solvers for multi-physics simulations”. In: *OpenFOAM® Journal* 3 (Feb. 2023), pp. 1–25. doi: 10.51560/ofj.v3.88. URL: <https://doi.org/10.51560/ofj.v3.88>.
- [5] M. Churchfield, S. Lee, and P. Moriarty. *Overview of the Simulator for Wind Farm Applications (SOWFA)*. Presentation. 2012. URL: <https://www.nrel.gov/wind/nwtc/assets/pdfs/sowfa-tutorial.pdf>.
- [6] B. Gatzhammer. “Efficient and Flexible Partitioned Simulation of Fluid-Structure Interactions”. PhD thesis. Technical University of Munich, 2014, pp. 1–183.
- [7] J. Jonkman et al. *Definition of a 5-MW Reference Wind Turbine for Offshore System Development*. NREL/TP-500-38060. 2009.
- [8] M. A. Sprague, C. Ananthan S. Vijayakumar, and M. Robinson. “ExaWind: A multifidelity modeling and simulation environment for wind energy”. In: *Journal of Physics: Conference Series* (2019). doi: 10.1088/1742-6596/1452/1/012071.
- [9] E. Taschner et al. “A new coupling of a GPU-resident large-eddy simulation code with a multiphysics wind turbine simulation tool”. In: *Wind Energy* (2023), pp. 1–21. doi: 10.1002/we.2844.
- [10] M. Weber. *fastFoam: An aero-servo-elastic wind turbine simulation method based on CFD*. Master thesis. 2017.

Appendix

Possible points to include in the Appendix

- Files from the OpenFOAM adapter with hints on how to modify them
- Files from AspFAST with hints on how to reuse the code for our mapping