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

Technical Report

Leonard Willeke
Delft University of Technology

Flow dynamics in wind parks are important for many engineering problems, ranging from optimized control to load reduction. This work introduces a preCICE-OpenFAST adapter designed to seamlessly couple OpenFAST with the preCICE library. preCICE facilitates the black-box coupling of simulation programs, enabling multi-physics simulations. The adapter acts as a driver code for OpenFAST, steering the turbine simulation and utilizing preCICE for efficient data exchange with the flow simulation. In a proof of concept, the adapter is employed to couple OpenFAST with OpenFOAM, demonstrating its potential despite limited capabilities. The presented approach lays the foundation for more comprehensive wind park simulations by integrating OpenFAST in the preCICE ecosystem. This research contributes to advancing wind park modeling techniques, providing a framework for studying complex interactions within wind farms.

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) [6]. 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 OpenFAST[1]. 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 and deflections which are imposed on the flow field. The challenge in this setup lies in implementing the data

[1] `https://openfast.readthedocs.io/en/main/`

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 example 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.

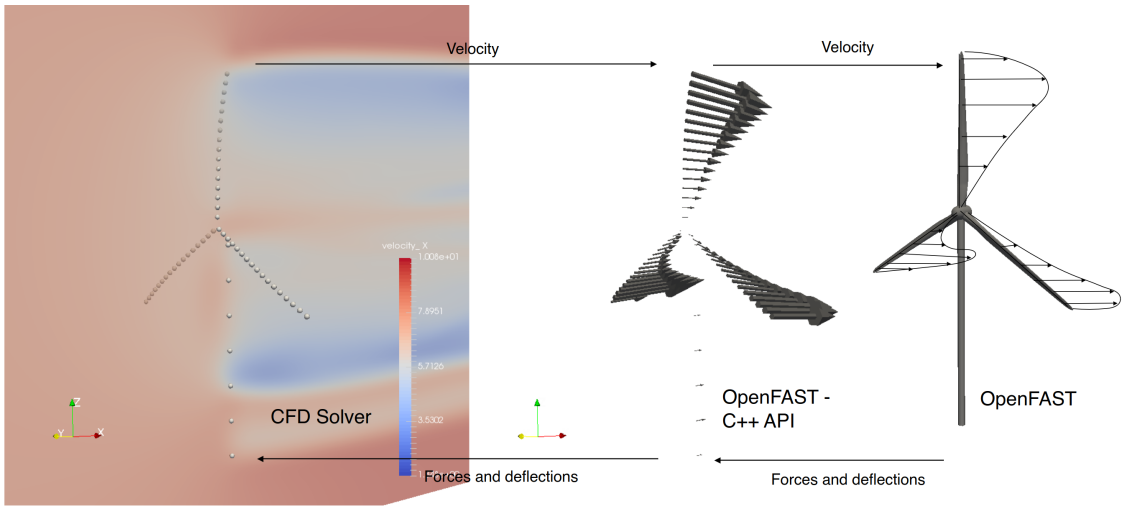


Figure 1. Coupling of OpenFAST to a CFD program. The fluid solver computes the flow dynamics and sends velocity data to OpenFAST. OpenFAST simulates the turbine dynamics and sends back forces and deflections, which are imposed on the flow field. Source: OpenFAST documentation[2]

II. Literature Review

A recent advancement in the coupling of OpenFAST is the tool **AspFAST** [10]. Developed at Delft 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

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

the actuator line model of OpenFAST. Because the coupling code[3] 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 [9] 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 algorihtms 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 the open-source CFD software 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 coupling tool similar to preCICE is **foamFAST** [11]. This project uses the commercial coupling tool for partitioned multi-physics simulations MpCCI, developed and sold by the Fraunhofer SCAI. As 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. MpCCI adapters exist for OpenFAST and OpenFOAM and have been tested on several benchmark cases.

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 seperately 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 abundance of existing coupling tools, 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. The black-box approach ensures a flexible simulation setup [7] with many coupling options. Explicit and implicit coupling schemes are available, as well as multi-coupling to connect more than two participants. 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. preCICE is scalable and performs on personal laptops and HPC applications alike.

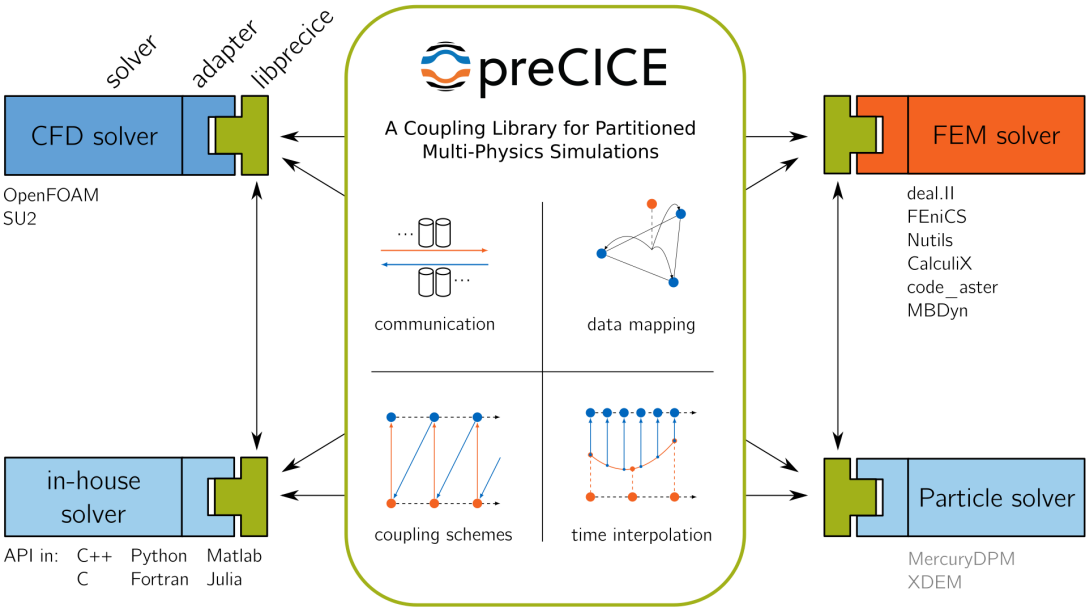


Figure 2. preCICE overview [3]

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 nature allows users to access, modify, and distribute the source code. Originally developed in the late 1980s,

[4] https://www.openfoam.com/

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. The software architecture of the adapter follows a modular design, allowing for flexibility and customization in the coupling process. Besides the core functionalities, the adapter consists of three distinct modules 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.

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. More information on how to install, use, and modify the adapter can be found in the official documentation[5].

4. OpenFAST

OpenFAST[6] 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. 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.

[5] `https://precice.org/adapter-openfoam-overview.html` (visited on 06.01.2024)

[6] `https://openfast.readthedocs.io/en/main/`

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. Simultanously, 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

```
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.

the main time loop (line 16-28) 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 finalized (lines 30-31). The software is developed and documented on GitHub[7].

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 reference model [8], 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 GitHub[8].

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 land-based 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 [10], 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 (Dummy) 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. This setup underlines the flexibility with which preCICE allows the coupling of participants in different setups and constellations.

B. Coupling OpenFAST with OpenFOAM

The motivation for this case is to perform a realistic simulation with OpenFAST and a CFD tool. To increase complexity, OpenFAST computes a land-based NREL 5MW turbine with a moving rotor. As fluid solver, OpenFOAM is used. The CFD program computes the fluid domain around the turbine to model the impact of the turbine on the wind field. The preCICE-OpenFOAM adapter is used for the communication with OpenFOAM. The coupling scheme is identical to the previous case. This case allows to explore the coupling in a realistic setup.

OpenFOAM computes a rectangular space surrounding the turbine. A subdomain of this space, defined

[7] https://github.com/LeonardWilleke/openfast-adapter

[8] https://github.com/LeonardWilleke/openfast-adapter/tree/main/cases

[9] https://precice.org/tooling-config-visualization.html

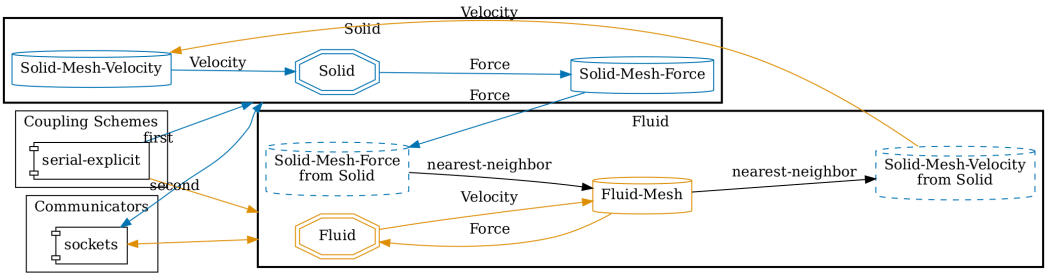


Figure 4. Coupling scheme between OpenFAST, named Solid, and the dummy Fluid solver. The figure was created from the *precice-config.xml* using the *config visualizer tool*[9].

as a *cellSet* in OpenFOAM and shown in Figure 5b, is coupled to OpenFAST. Force data from the OpenFAST mesh seen in Figure 5a are mapped to this subdomain, while velocity data is transferred back. The mapping between the two different domains is performed by preCICE. Although a first data exchange is achieved, the coupling is physically not correct: The data sent from OpenFAST is applied in OpenFOAM not as force, but as pressure. This is a workaround to test how field values are handled in the OpenFOAM adapter, which was not designed to apply force as a field value as well.

The broader goal for this example case is to implement the Actuator-Line Method to represent the turbine in the OpenFOAM fluid field. This would ensure a correct coupling. However, the preCICE-OpenFOAM adapter is not designed to implement this method. How to map between the line mesh in OpenFAST and the volume mesh in OpenFOAM as shown in Figure 5 therefore remains an open question.

V. Challenges

What are the next steps in the coupling of OpenFAST via preCICE? This section aims to give an overview of the upcoming tasks to guide interested developers in their endeavour.

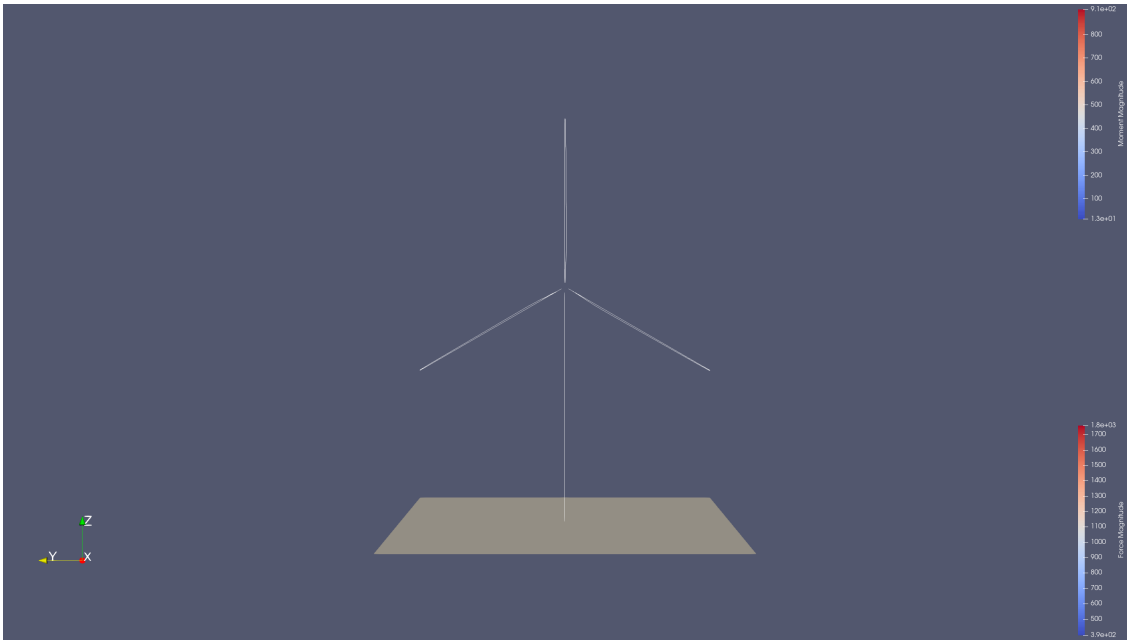
A. Mapping turbine data between actuator lines and volume meshes

The most pressing issue is to implement a correct mapping. As seen in chapter IV, the mapping depends not only on the preCICE-OpenFAST adapter, but also on the preCICE adapter of the CFD solver. In our case, we face problems because the OpenFOAM adapter was not designed for the implementation of ALM. In particular, the volume coupling functionalities are not sufficient. It is possible to retrieve velocity data from a volume. But it is not possible to write force or pressure gradient data on a volume. However, this functionality is needed to implement a bidirectional coupling for this FSI case. I see two solutions to this problem:

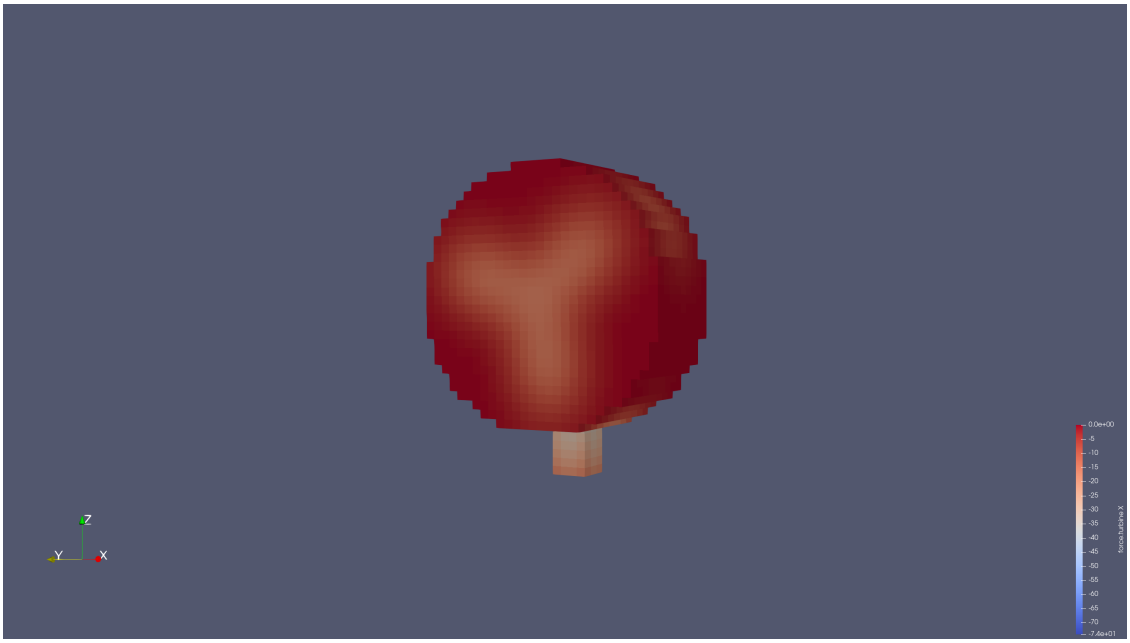
Solution 1: Map in the OpenFAST adapter

The code[10] from AspFAST provides a possible solution. The tool to connect OpenFAST and GRASP is published under a MIT license and can therefore be re-used without limitations. It implements useful mapping functions in the main script *aspast.cpp*. To communicate from OpenFAST to the CFD solver, the functions *calcBodyForce*, *uniformBodyForce* and *diskBodyForce* are used. To understand the mapping from CFD to OpenFAST, look into the functions *sampleVelocity* and *calcVelocity*. Including similar functions in the OpenFAST adapter could help to address mapping problems. It is still unclear how the mapped force values would then be transferred to a volume mesh in OpenFOAM.

[10] `https://gitlab.com/whiffle-public/aspfast` (visited on 28.12.2023)



(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.

Solution 2: Modify the OpenFOAM adapter

Another solution is to add the missing functionality in the OpenFOAM adapter. For the coupling setup, the Fluid-Fluid (FF) module of the adapter is used. For a correct coupling, OpenFOAM needs to include the updated field values of force or pressure gradient from OpenFAST. But the OpenFOAM adapter only supports

the exchange of pressure as field value. To change this, we need to adapt the class *PressureGradient.C* of the FF module. A code section to write pressure gradient data on a cellset should be added. Have a look in the class *Pressure.C* of the FF module to get an idea how to access and write field values in OpenFOAM in general.

Inside OpenFOAM, the pressure gradient is stored as the boundary condition of the velocity field U. Therefore, we need to import the velocity field U in the class *PressureGradient.C* and use appropriate commands from OpenFOAM to set its boundary condition. Setting the boundary condition on a cellset and not on a wall or inlet might be tricky.

As an additional remark, the OpenFOAM library turbinesFoam[11] [2] might be useful. It implements the actuator line method with different solvers like pimpleFoam. 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 in OpenFAST and map the results to OpenFOAM, not do the whole computation in OpenFOAM.

B. Improve code maturity

To railguard the further development, a regression test should be implemented. The dummy fluid solver from chapter IV.A can be used as a simple coupling participant to check calculations. The preCICE-FMI runner[12] may serve as an example on how to write the test, place it in the GitHub repository and execute it automatically with GitHub workflows.

Once a mature coupling to OpenFOAM is achieved, the coupling should be verified. Previous work [10] provides a benchmark case to cross-verify the coupling of a single NREL 5MW turbine with five other research LES codes.

C. Enable coupling with multiple turbines

A future version of the adapter may include the coupling of multiple turbines in OpenFAST with one OpenFOAM instance. The OpenFAST C++ API allows to run multiple instances of OpenFAST to simulate wind park scenarios with FAST.FARM. But how do you communicate the results of multiple turbines to preCICE, and how do you organize the coupling to the CFD solver? This scenario increases the complexity in both adapters involved in the coupling setup. Again, a look into the AspFAST code could provide insight into how to deal with those challenges.

A different option may be the use of the preCICE Micro manager[13]. It is a tool to manage many micro simulations and couple them to one macro simulation. However, it must be said that the manager was not designed with large-scale FSI simulations in mind.

D. Explore coupling scenarios with other CFD solver

The current work is focused on the coupling of OpenFAST with OpenFOAM for the simple fact that it is the only suitable CFD solver in the preCICE ecosystem. The coupling of other tools like GRASP, NaluWind or YALES2 via preCICE might be interesting to create a simulation environment where CFD solvers could be swapped easily. Simulation results could be compared mor readily and the individual strengths of each program leveraged depending on the simulation case. However, this setup comes with the additional effort of developing preCICE adapters for the other CFD programs. As the tools above are already coupled to OpenFAST with mature, verified tools, this should only be done if it adds real benefit to the research community.

[11] `https://github.com/turbinesFoam/turbinesFoam` (visited 14.12.2023)

[12] `https://github.com/precice/fmi-runner/tree/main` (visited on 06.01.2024)

[13] `https://precice.org/tooling-micro-manager-overview.html` (visited on 06.01.2024)

VI. Conclusion

A first preCICE-OpenFAST adapter to couple the wind energy engineering tool to CFD solvers was presented. 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 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 vertixal-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] M. Churchfield et al. *An advanced actuator line method for wind energy applications and beyond*. NREL/CP-5000-67611. March 2017.

[7] B. Gatzhammer. "Efficient and Flexible Partitioned Simulation of Fluid-Structure Interactions". PhD thesis. Technical University of Munich, 2014, pp. 1–183.

[8] J. Jonkman et al. *Definition of a 5-MW Reference Wind Turbine for Offshore System Development*. NREL/TP-500-38060. 2009.

[9] 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.

[10] 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.

[11] M. Weber. *fastFoam: An aero-servo-elastic wind turbine simulation method based on CFD*. Master thesis. 2017.