Science and Technology Facilities Council

User Guide – Partitioned Multi-physical Simulation Framework

13/05/2020

1 Overview

Partitioned Multi-physical Simulation Framework is developed based on the MUI library and offers a platform where users can carry out multi-physical (mainly fluid-structure interaction) studies with High Performance Computers developed and maintained by the fluid-structure interaction (FSI) team of SCD-STFC. The framework uses partitioned approach to couple two or more physical domains together for a multi-physical simulation. It takes several advantages of the MUI library that:

- Flexible of select solvers for each physical domain;
- Flexible of extend the number of physical domains;
- Good scalability on communications among physical domains for large simulations.

Currently, the framework has implemented the coupling between Fluid solver and Structure for fully coupled FSI simulations:

```
Fluid-Structure Interactions
├── Fluid Solver
│   └── OpenFOAM_V6
├── Structure Solver
│   └── FEniCS_V2019.1.0
└── Communicator
    └── MUI-1.0
```

Figure 1 shows the flow chart of the present framework. The left hand side is the CFD solver for the fluid domain and the right hand side is the CSM solver for the structure domain. In the time step of $t = t + \delta t$ and sub-iteration $iter + 1$, the fluid domain solved the flow field and push the fluid forces at each cell of the fluid structure interface to the structural domain through the MUI library. The displacements at each cell of the interface were determined by the MUI coupling utility based on the fetched value from the structural domain. The calculated displacements of each cell at the interface were then applied to the fluid domain as a Dirichlet boundary condition. While the structural

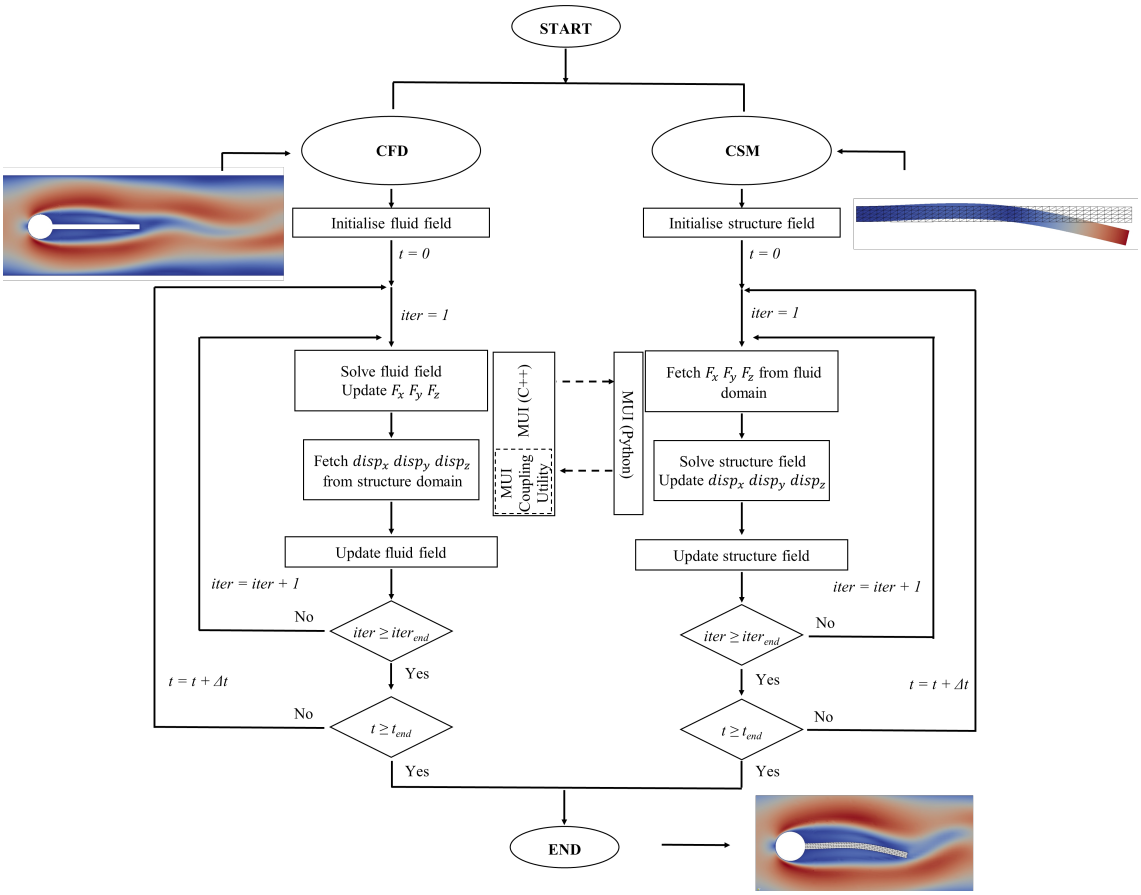

Figure 1: Flow chart of the simulation framework.

domain fetch fluid forces and applied as a Neumann boundary condition of the structure. It further calculated the deformation of the structure and push to the fluid solver. The stress of the structure will then be updated. Both fluid and structure domains were move to the next sub-iteration after the completion of the above actions. Several sub-iterations were needed within each time step until a convergence is reached, i.e. the R_k is smaller enough that below the cretiria.

Apart from the loose coupling, the present framework also implemented both the Fixed Relaxation and the Aitken's δ^2 methods with FSI sub-iterations implemented in both CFD and CSM solvers, as shown in Figure 1. The Radial Based Function (RBF) that has been implemented in the MUI library was used to ensure the forces at the interface are conserved between the two domains.

This framework is under active development to involve more solvers as well as more physical domains. Such infrastructure will make it possible to simulate large multi-physical problems and simulate complicated multi-physical cases.

2 Download

The core code of the Partitioned Multi-physical Simulation Framework can be obtained from the GitLab:

http://hcp004.hartree.stfc.ac.uk/wendi_liu/FSI_Simulation_Framework.git

3 Installation

Please follow the below steps to install the Framework:

3.1 Step 1: Download and install FEniCS V2019.1.0

- Install pre-required modules:
  - CMake V3.10+
  - OpenMPI V2.1+
  - Python V3.6+
  - Eigen V3.3+
  - PETSc V3.7+
  - Boost V1.68+
  - Pybind11 V2.2+
- Follow the instructions in https://fenicsproject.org/download/ to download and install FEniCS_V2019.1.0.

3.2 Step 2: Download MUI library and install the MUI Python wrapper

- Download MUI V1.0 from https://github.com/MxUI/MUI/archive/1.0.tar.gz.
- Install pre-required modules:
  - CMake V3.10+
  - MPI4py V3.0+
  - Python V3.6+
  - OpenMPI V3.1.0
  - numpy V1.13+
  - Eigen V3.3+
  - Petsc V3.7+
  - Pybind11 V2.2+
- source the FEniCS environment.
- Follow the instructions of README.md in the **MUI/wrappers/Python** to install the Python wrapper of the MUI library.

3.3 Step 3: Download the Framework

- Download the Framework from http://hcp004.hartree.stfc.ac.uk/wendi_liu/FSI_Simulation_Framework.git

3.4 Step 4: and Download and install OpenFOAM V6

- Clone a copy of the main OpenFOAM-6 repository (https://github.com/OpenFOAM/OpenFOAM-6) and associated ThirdParty-6 repository (https://github.com/OpenFOAM/ThirdParty-6).
- Extract the patch archive files from **parMupSiF/src/MUI_Utility/MUIpatchArchive-Ope** into the base folder where cloned the two repositories
- Execute the script

```
./patch\_OF6-MUI
```

- Create an alias in the .bashrc file to "sourcing" OpenFOAM. i.e. edit `.bashrc` in the home directory and add

```
alias of6='source␣[path␣to␣where␣cloned␣the␣OpenFOAM-6␣directory]
/etc/bashrc;␣export␣WM\_NCOMPPROCS=8'
```

- Re-source `.bashrc` script using

```
source .bashrc
```

- Trigger the alias by typing

```
of6
```

there should see no errors

- In the ThirdParty-6 folder, place a full copy of MUI library into the new folder `MUI` (these can be sym links but make sure they are not relative paths as a script takes a copy of these files, whenever update MUI the old platform files need to clean and re-run `Allwmake` to update the copy that gets used by the OpenFOAM applications).
- Build the ThirdParty libraries with `Allwmake`
- Build OpenFOAM-6 itself using `Allwmake` in the base of the repository folder (as usual)
- There is a sample coupled case using `laplacianFoam` in **tutorials/basic/laplacianFoam**

3.5 Step 5:Install the OpenFOAM solver `pimpleFSIFoam`

- Go to **`parMupSiF/src/CFD/OpenFOAM/V6/parmupsifLibs`**.
- run

```
./Allwmake
```

to compile the solver `pimpleFSIFoam` and boundary conditions `parmupsifBC`

4 Folder Structure

```
parMupSiF
├── config
│   └── parmupsif_gcc.conf [Configure files for Scafell Pike]
├── demo
│   ├── couplingLibDemo [Demo case for MUI coupling utility]
│   ├── cylinderFlap [Demo case on 2D FSI simulations]
│   ├── FSIBeam [Demo case on 3D FSI simulations]
│   └── Structure_validation [Demo case on the in-house FEniCS
│       solver]
├── docs
│   ├── ToDo_List
│   └── UserGuide
├── src
│   ├── CFD
│   │   └── OpenFOAM
│   │       └── V6
│   │           └── parmupsifLibs [the solver pimpleFSIFoam and boundary
│   │               conditions parmupsifBC]
│   ├── CSM
│   │   └── FEniCS
│   │       └── V2019.1.0
│   │           └── structureFSISolver [the in-house FEniCS structure
│   │               solver]
│   └── MUI_Utility
│       ├── couplingLab [the MUI coupling utility]
│       ├── MUIpatchArchive-MUI_dev [the patch archive files to
│       │   modify MUI library on its RBF sampler]
│       └── MUIpatchArchive-OpenFOAM_6 [the patch archive files
│           to merge MUI library into OpenFOAM V6]
├── thirdparty
│   └── eigen [third party libraries (i.e. Eigen)]
└── README.md
```

5 Demo Cases

The Framework provided several demo cases. Taking the demo case on 3D FSI simulations `FSIBeam` as an example, the folder structure for a simulation case is as follows:

```
FSIBeam
├── caseSetup [contains the original settings of the simulation
│   for both fluid and structure solvers]
│   ├── fluidDomain [contains the original settings of the simulation
│   │   for the fluid solver]
│   └── structureDomain [contains the original settings of the
│       simulation for the structure solver]
├── runCtrl [contains the utilities for the simulation]
├── Allclean [clean all the running results]
├── Allrun [run the simulation]
└── FlexBeam1e6.sh [Job script for Scafell Pike]
```

After running, another folder will be generated to contain the simulation results:

```
FSIBeam
├── caseSetup
├── runCtrl
├── runData [Results folder]
│   ├── fluidDomain [contains the results from the fluid solver]
│   ├── logFiles [contains the log files for the entire simulation]
│   └── structureDomain [contains the results from the structure
│       solver]
│       ├── dataInput [contains the mesh for the structure solver]
│       ├── structureFSISetup [contains the set-up for the structure
│       │   solver]
│       ├── structureResults [contains the results from the structure
│       │   solver]
│       └── structureDomainRun.py [high-level script for the structure
│           solver]
├── Allclean
├── Allrun
└── FlexBeam1e6.sh
```