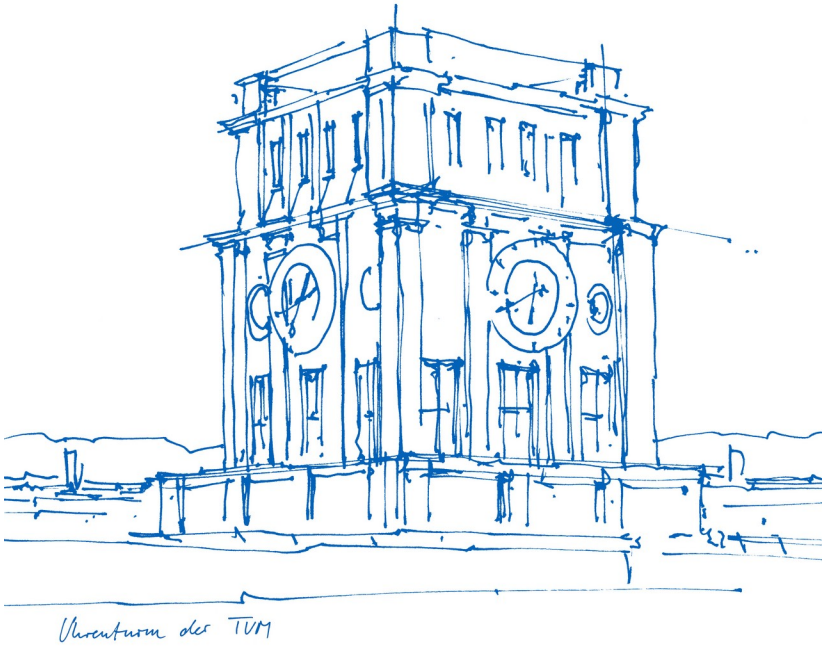
Lehrstuhl für Statik
Ingenieurfakultät Bau Geo Umwelt
Technische Universität München

Master‘s Thesis Presentation

Development of a general OpenFOAM-Coupling-Adapter for the Kratos-CoSimulation Multiphysics solver



Ashish Shankar, Darekar

Chair of Structural Analysis

Technical University of Munich

Munich, 07. February 2022

Examiner : Prof. Dr.-Ing. Kai-Uwe Bletzinger

Assistant advisor : Philipp Bucher

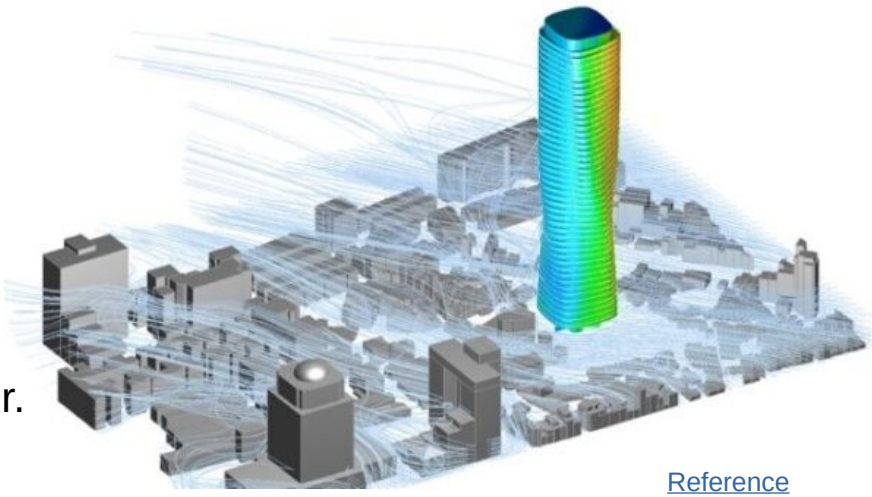
Overview

- Motivation, Introduction, Problem Definition.
- Literature review: Coupling tools and available OpenFOAM adapters.
- Implementation: Kratos-OpenFOAM coupling adapter and OpenFOAM Wrapper.
- Results: Validation and Verification.
- Conclusion and future scope.
- Additional Information.

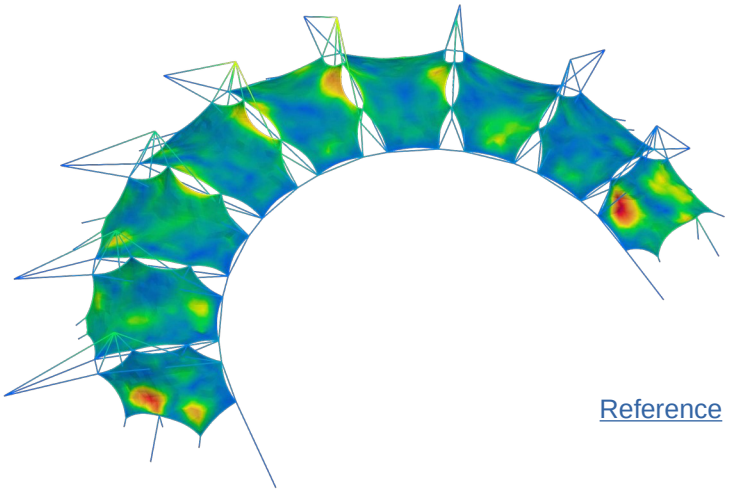
Motivation, Introduction, Problem Definition

Motivation

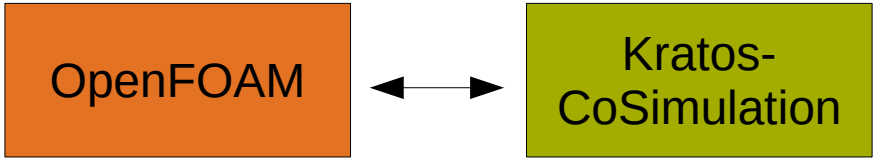
- Solving FSI problem using CoSimulation.
- Fluid solver: OpenFOAM.
- Structural solver: Kratos - StructuralMechanicsApplication or any other.
- Coupling OpenFOAM to CoSimulation.



Reference



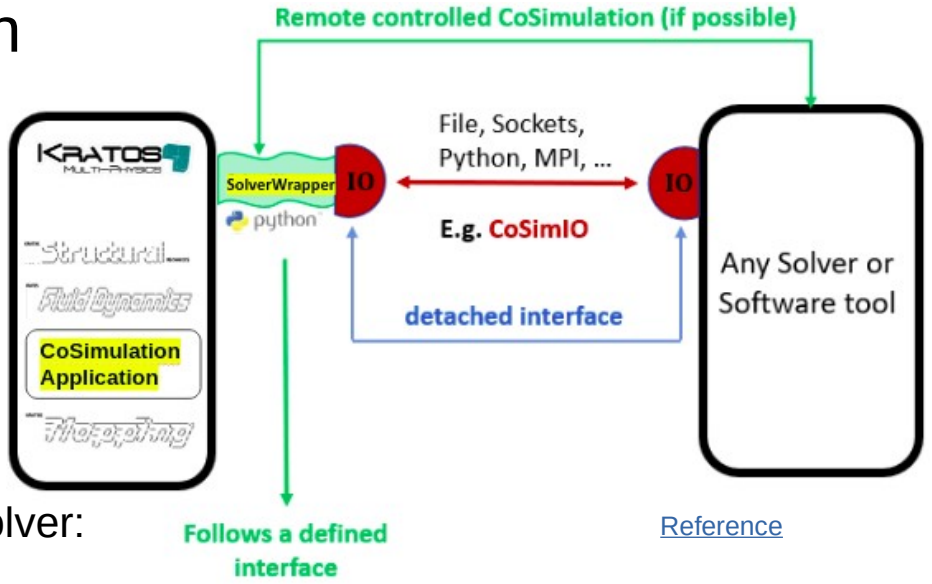
Reference



Introduction

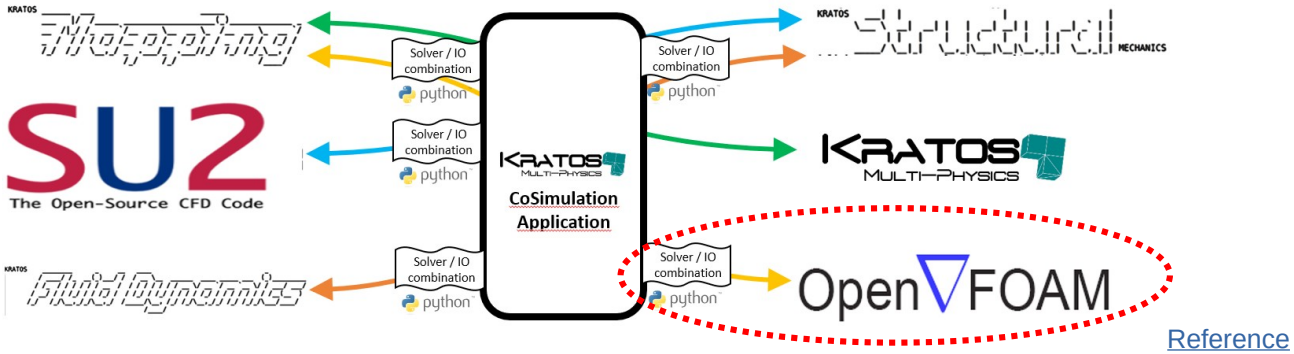
Kratos-CoSimulationApplication

- Features :
- ✔ Partioned approach - based coupling tool
- ✔ Coupling Algorithms
- ✔ Convergence Accelerators
- ✔ Data transfer operations: eg. Mapping



Reference

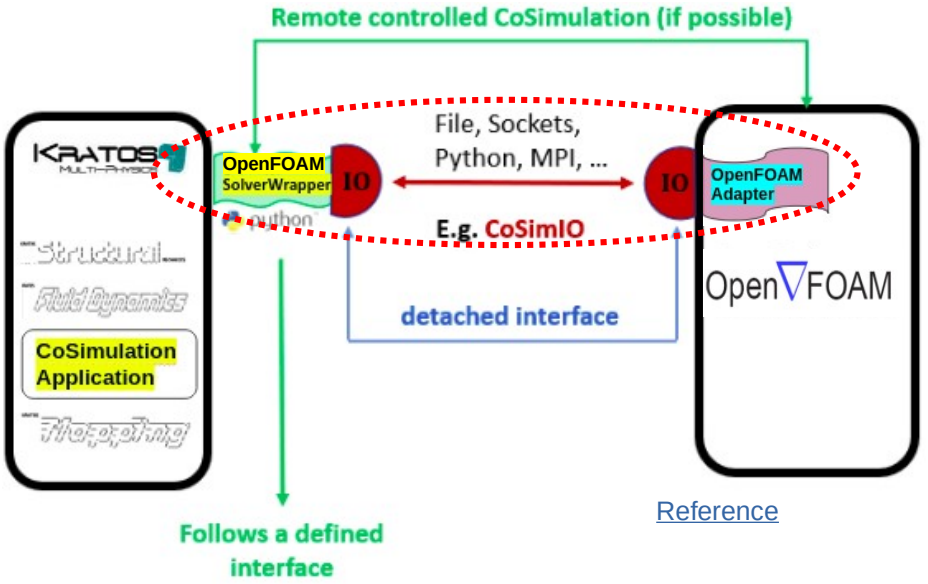
- Components required to couple external solver:
- ✔ Solver Wrapper: Detached-Interface
- ✔ IO: For data exchange between the solvers



Reference

Problem Definition

- FSI :
- ✔ Which coupling strategy to solve?



- Kratos - CoSimulation:

- ✔ Is new external wrapper required?

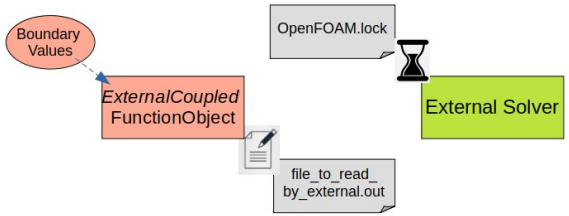
- OpenFOAM :

- ✔ What type of a data is required?
- ✔ How to gather the required data?
- ✔ Is data processing required?
- ✔ How to transfer it to the CoSimulation?

Literature Overview:
Coupling tools and available OpenFOAM adapters

Literature Overview

Coupling OpenFOAM with an external solver

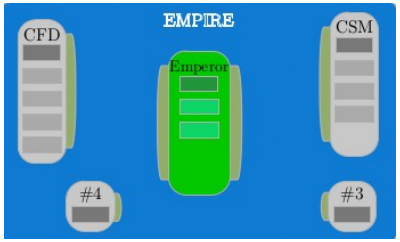


1. ExternalCoupled - OpenFOAM‘s function object

- File-based communication interface.

2. EMPIRE - Enhanced Multi-Physics Interface Research Engine

- Based on client-server model.
- Communication - MPI.
- Modifications in the source code of OpenFOAM solvers.



3. MpCCI - Multiphysics Interfaces:

- Injecting code at runtime using OpenFOAM‘s function object.
- Communication - MPI.

MpCCI – Multiphysics Interfaces

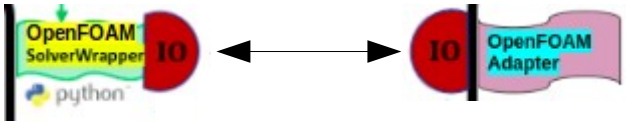
4. PreCICE - Coupling library for partitioned multi-physics simulations

- Retrieves the appropriate data using the objects’ registry.
- Communication - MPI or TCP/IP sockets.

Implementation: Kratos-OpenFOAM coupling adapter and OpenFOAM Wrapper

Solution Strategy and Scope

- FSI :
- ✔ Which coupling strategy to solve? - Explicit (or weak) coupling.



- Kratos - CoSimulation:
- ✔ Is new external wrapper required? - Yes, “OpenFOAM Wrapper“.

- OpenFOAM :
- ✔ What type of a data is required?
- ✔ How to gather the required data?
- ✔ Is data processing required?
- ✔ How to transfer it to the CoSimulation (OpenFOAM Wrapper)?

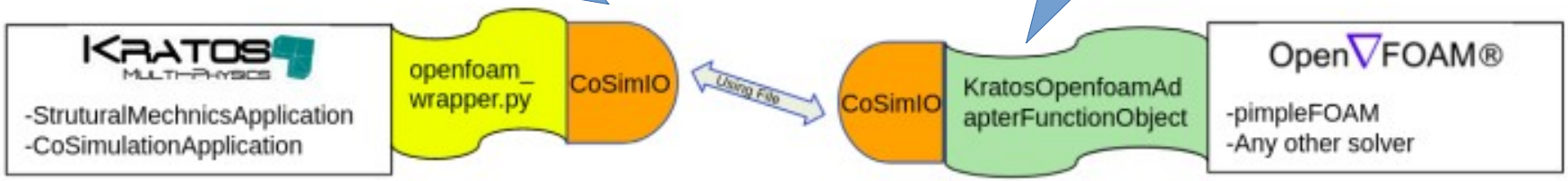
OpenFOAM Adapter

Implementation

Basic design aspects for an Adapter

1. What type of a data is required?
   Explicit coupling : Load and displacement (one time).
2. How to gather the required data?
   OpenFOAM‘s Object Registry, LookupObject methods.
3. Is data processing required?
   Yes, for loads.
4. How to transfer it to the CoSimulation (OpenFOAM Wrapper)?
   CoSimIO : 5 ways of communications available.
   File, Socket, Unix-domain socket, Pipe, MPI.

Function Object

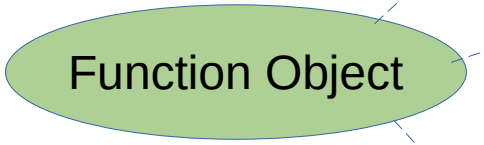


Implementation

Isolating an adapter : Function Object

- No modification : in the solver‘s source code.
- Loaded at runtime, independent of OpenFOAM.
- Built-in and user-defined function objects.
- Public methods, Multiple injection points in solver.
- Easy to configure and extensible.

```
functions
{
    yPlus1
    {
        type yPlus;
        libs ("libfieldFunctionObjects.so");
        patches (leftwall rightwall lowerwall);
    }
    CourantNo1
    {
        type CourantNo;
        libs ("libfieldFunctionObjects.so");
        executeControl      timeStep;
        executeInterval     2;
        writeControl        writeTime;
    }
}
```



No.	Methods	Time of injection
1.	read()	Before beginning of 1st time step
2.	execute()	At each ++ or += of the time-loop
3.	write()	At each ++ or += of the time-loop
4.	end()	When *Time::run()* determines that the time-loop exits
5.	adjustTimeStep()	At the end of *Time::adjustDeltaT()*, if *adjustTime* parameter is true
6.	updateMesh()	When *Time::updateMesh()* is called
7.	meshPoints()	When *Time::movePoints()* is called

OpenFOAM®

Any solver

```
/* Start the solver */

Info<<"\nStarting time loop\n"<< endl;
while (runTime.run()) {
    #include "readTimeControls.H"
    #include "compressibleCourantNo.H"
    #include "setDeltaT.H"

    runTime++;

    /* solve the equations */
    #include "rhoEqn.H"
    while (pimple.loop())
    {
        ...
    }

    runTime.write();
}

/* Finalize */
```

Reference

Implementation

OpenFOAM®

Retrieve the data : ObjectRegistry

- Hierarchical database in OpenFOAM.
- Hash table contains regIOobjects.
- *RunTime_ --> mesh_ --> fieldData_*
- LookupObject<>() methods
  eg. pressure field

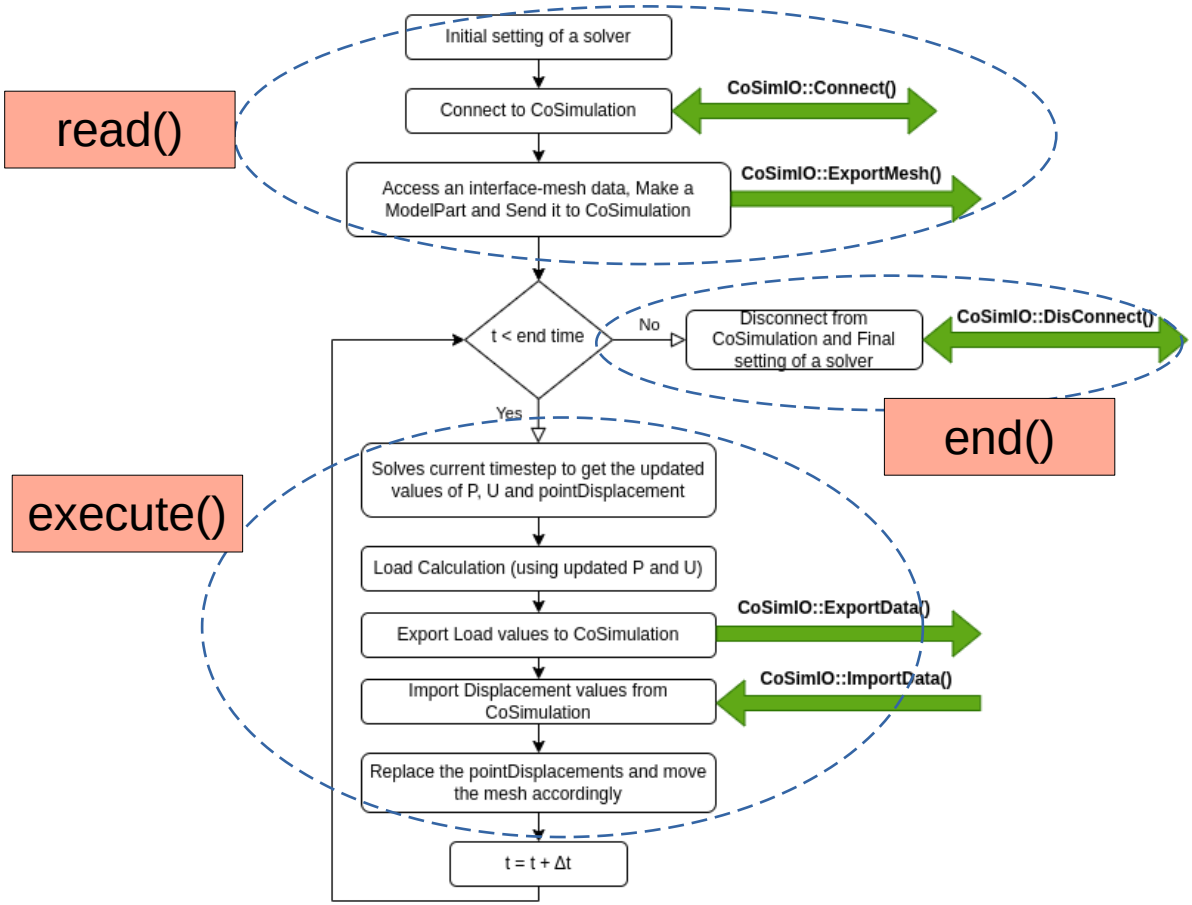
Solver's Data Base

```
runTime                                          //Time          (objectRegistry)
|--->controlDict                                 //IOdictionary  (regIOobject)
|--->FluidMesh                                   //fvMesh        (objectRegistry)
 |-->fvSchemes, fvSolution, transportProperties //IOdictionary  (regIOobject)
 |-->points, faces, owner, neighbour, boundary  //PolyMeshData  (regIOobject)
 |-->pointZones, faceZones, cellZone            //PolyMeshData  (regIOobject)
 |-->T, U, p                                    //volDataFields (regIOobject)
|--->SolidMesh                                   //fvMesh        (objectRegistry)
 |-->fvSchemes, fvSolution, transportProperties //IOdictionary  (regIOobject)
 |-->points, faces, owner, neighbour, boundary  //PolyMeshData  (regIOobject)
 |-->pointZones, faceZones, cellZone            //PolyMeshData  (regIOobject)
 |-->T, U, p                                    //volDataFields (regIOobject)
```

```
// Pressure boundary field
tmp<volScalarField> tp = mesh_.lookupObject<volScalarField>("p");
const volScalarField::Boundary& pb (tp().boundaryField());
```

Implementation

KratosOpenfoamAdapterFunctionObject – Design and algorithm



Implementation

KratosOpenfoamAdapterFunctionObject - Details

1. Accessing coupling interface-mesh in OpenFOAM

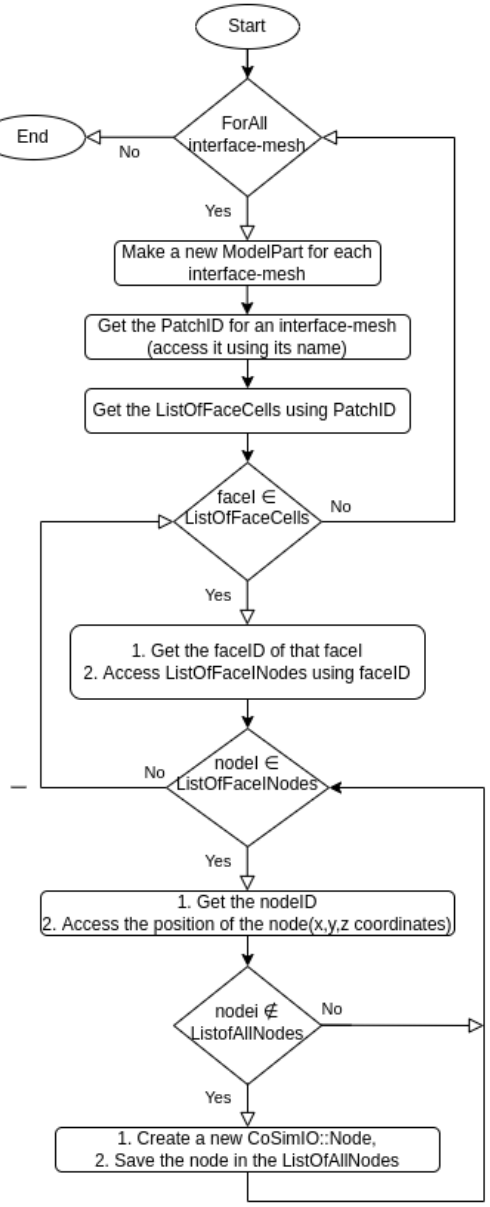
- Export mesh to CoSimulation : using ModelPart.
- ModelPart acts as a mesh container : contains Nodes.
- Use of OpenFOAM‘s ObjectRegistry : *mesh_*
- CoSimIO :: Node is created using APIs.

2. Use of Nodes: Advantages of Mapping

- OpenFOAM creates non-standard entries at boundaries.
- Data transfer – using Nodes.
- Use of point-based mappers.
  Nearest neighbor or barycentric mapper.

3. Displacement related

- Field name: “*PointDisplacement*“.
- Dynamic Mesh handling: setting using *dynamicMeshDict.*



Implementation

KratosOpenfoamAdapterFunctionObject - Details

4. Force or Load calculation

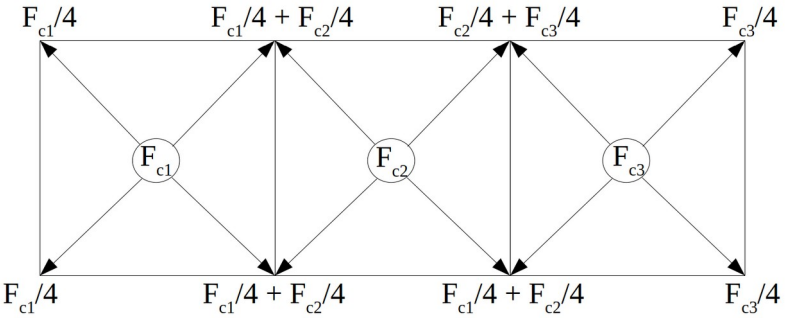
- At any point : Total force = Pressure force + viscous force.
- Pressure, velocity, flow properties : ObjectRegisty and lookupObject<>() methods.

$$F_{total} = F_p + F_v,$$

$$F_p = \sum_i \rho_i . s_{f,i} . (p_i - p_{ref}),$$

$$F_v = \sum_i s_{f,i} . (\mu R_{dev}),$$

- Values present on cell center.
- Conversion from Cell to Nodal value : Interpolation technique.

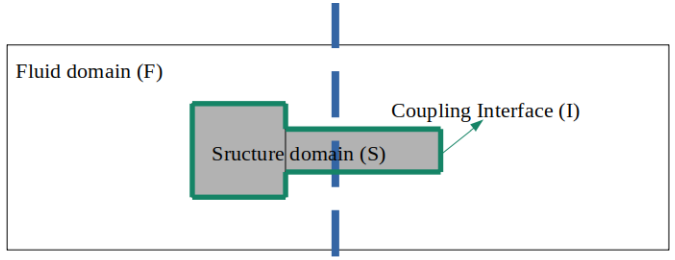


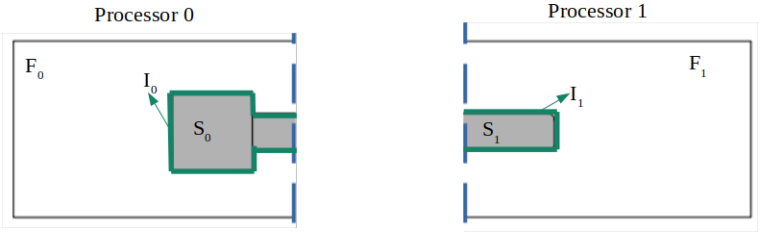
Implementation

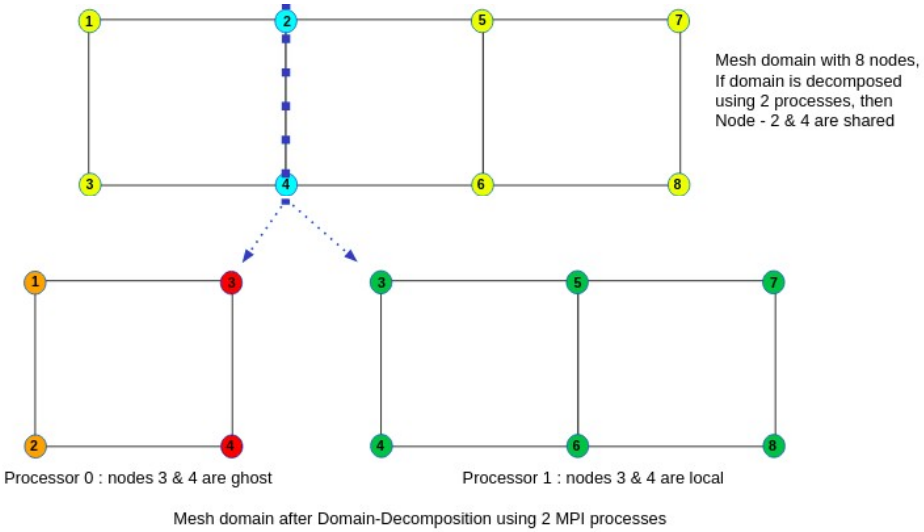
KratosOpenfoamAdapterFunctionObject - Details

5. MPI Parallelism in the adapter

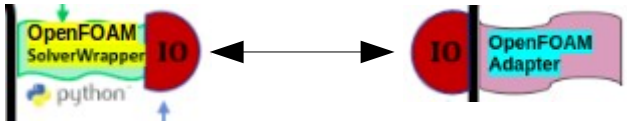
- OpenFOAM, CoSimulation : runs in MPI.
- Using domain-decomposition.
- Communication : subdomain-by-subdomain basis.
- OpenFOAM divides Interface mesh.
- Create ghost and local nodes.
- Load data exchange between the ranks.







Implementation



OpenFOAM solver Wrapper

- External solver wrapper: Communication partner with the OpenFOAM adapter.
- Calls the specific functions of the CoSimulation.
- Important functions that need to be defined:

  1. Initialize():
  - ✔ Read the input files, prepare internal data structures.
  - ✔ Formation of the coupling-mesh.

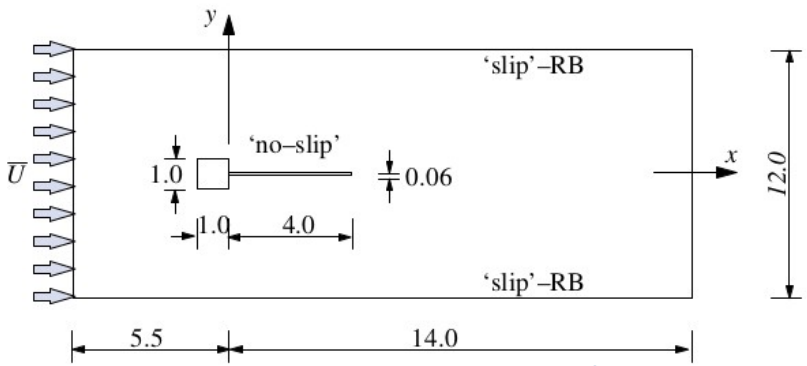
  2. SolveSolutionStep():
  - ✔ **Import** load values and **Export** displacement values during each time step.

Results and Validation

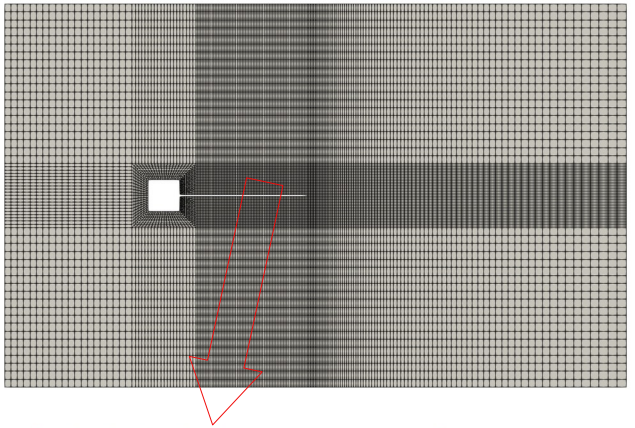
Lehrstuhl für Statik
Ingenieurfakultät Bau Geo Umwelt
Technische Universität München

Validation Case

FSI-Benchmark: Flow-induced vibration of a flexible beam or flap



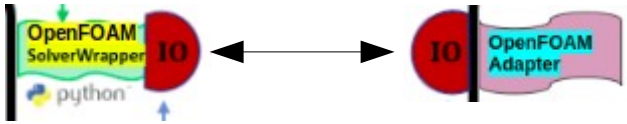
Reference



- Test case setup from W. Wall.
- OpenFOAM : PimpleFoam solver.
- Kratos-StructuralMechanicsApplication.
- CoSimulation : Nearest-neighbor mapping, Initial load ramp-up.

Lehrstuhl für Statik
Ingenieurfakultät Bau Geo Umwelt
Technische Universität München

Validation Case



Configuration files

```
"solvers" :
{
    "Openfoam_Kratos_Wrapper":
    {
        "type" : "solver_wrappers.external.openfoam_wrapper",
        "solver_wrapper_settings" : {
            "import_meshes"     : ["interface_flap"],
            "export_data"       : ["disp_interface_flap"],
            "import_data"       : ["load_interface_flap"]
        },
        "io_settings" : {
            "type"       : "kratos_co_sim_io",
            "connect_to" : "Openfoam_Adapter"
        },
        "data" : {
            "disp_interface_flap" : {
                "model_part_name"   : "interface_flap",
                "dimension" : 3,
                "variable_name" : "DISPLACEMENT"
            },
            "load_interface_flap" : {
                "model_part_name"   : "interface_flap",
                "dimension" : 3,
                "variable_name" : "FORCE"
            }
        }
    }
```

Example of a configuration of the OpenFOAM wrapper (in CoSimulation's configuration file)

```
KratosOpenfoamAdapterFunctionObject1 //User-specific name, can be changed
{
    type KratosOpenfoamAdapterFunctionObject; //Type-name, do not change
    libs ("../KratosOpenfoamAdapterFunctionObject/
    libKratosOpenfoamAdapterFunctionObjectFunctionObjects.so");
    //Path to generated shared library file

    interfaces //Key-word, do not change
    {
        Interface1
        {
            patches                 (flap); //Actual patch-name used by OpenFOAM
            name                    interface_flap;
            importDataIdentifier    (disp_interface_flap);
            exportDataIdentifier    (load_interface_flap);
        }
        Interface2 //More interfaces can be added here in similar way like above
        {...}
    }

    parameters //Key-word, do not change
    {
        rho rho [1 -3 0 0 0 0 0] 0.00118; //Density of a fluid (units are must)
        nu nu [0 2 -1 0 0 0 0 ] 0.1542; //Kinematic viscosity (units are must)
    }

    //Additional parameters
    //debugLevel info;  // For debugging, Default value is info
    //dim 3;            // Dimension of a problem, Default value is 3
}
```

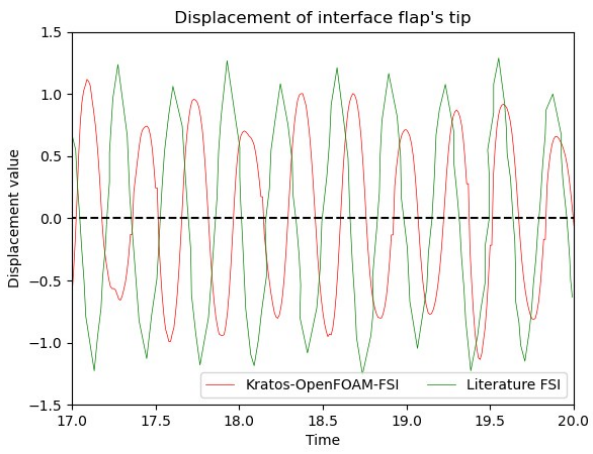
Example of a configuration of the OpenFOAM adapter (in OpenFOAM's *controlDict* case file)

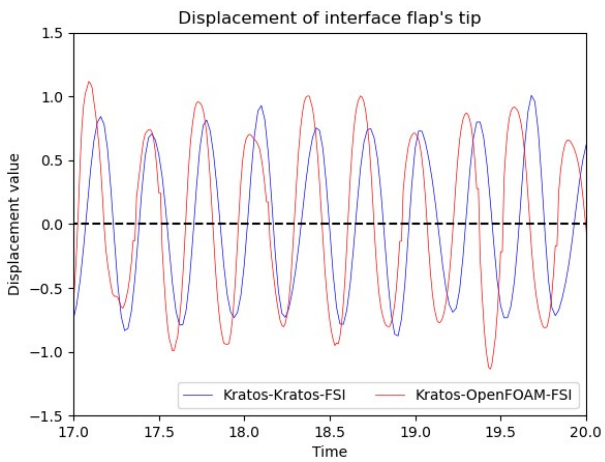
Lehrstuhl für Statik
Ingenieurfakultät Bau Geo Umwelt
Technische Universität München

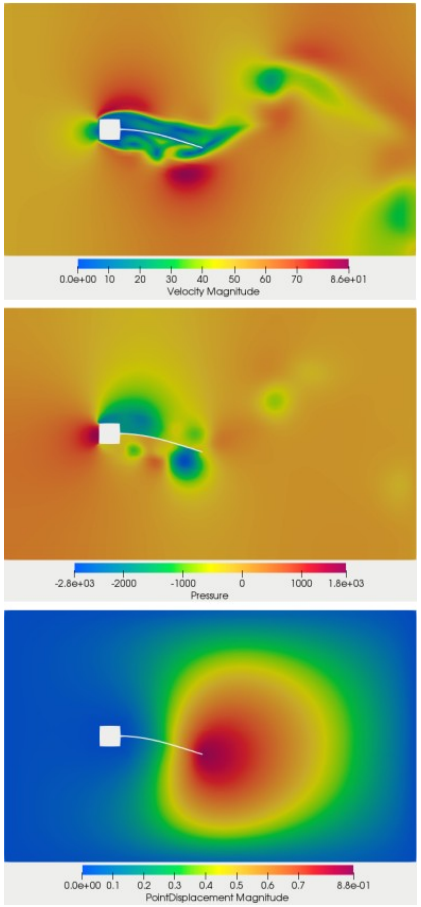
Results

FSI-Benchmark: Flow-induced vibration of a flexible beam or flap

- Flap‘s vibrations comparision.
- Maximum amplitude and frequency values.
- Some discrepancies : Due to fluid solvers' employing different schemes, meshes, and some parameter settings.
- Verification: Simulation performed solely on Kratos.





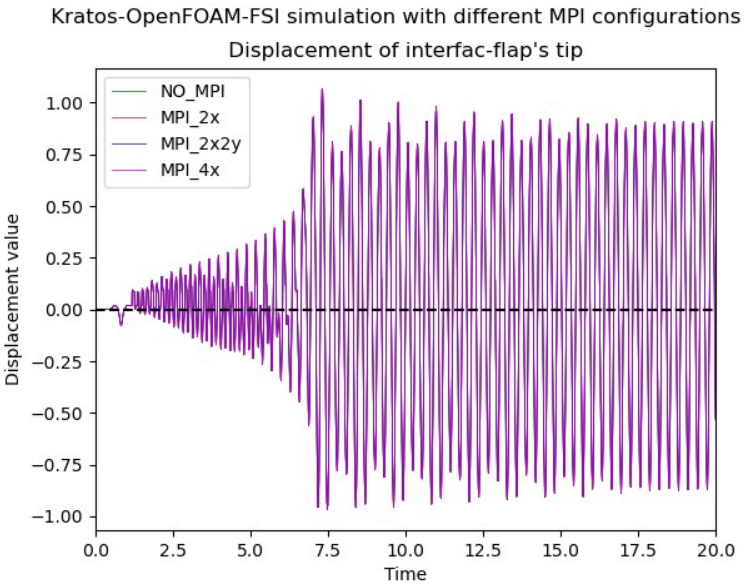


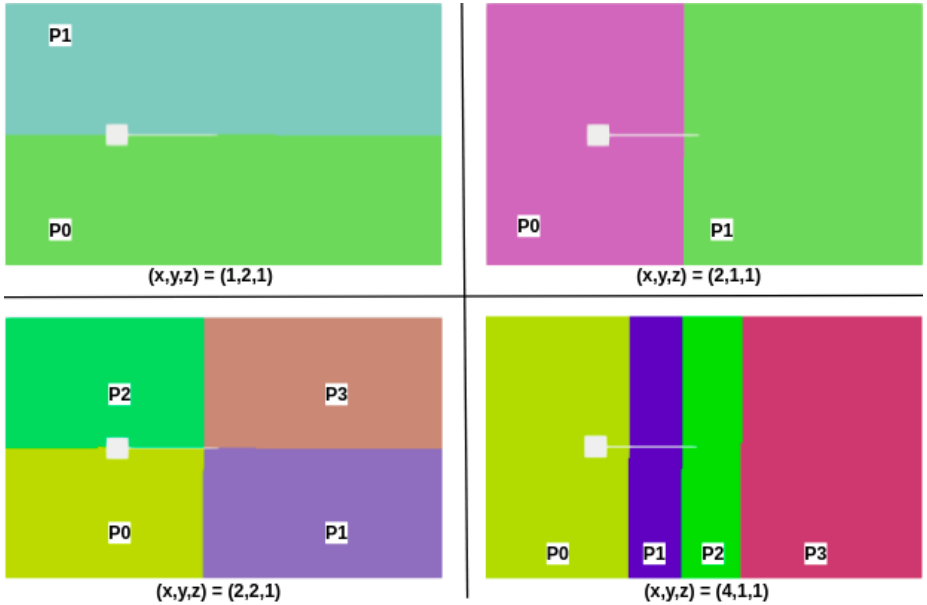
Velocity, Pressure, PointDisplacement

Results

FSI-Benchmark: Flow-induced vibration of a flexible beam or flap

- MPI implementation of the adapter.
- Parameters : *“decomposeDict“* file.
- Flap‘s displacement verified.

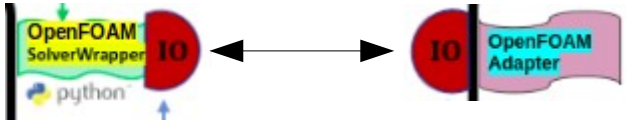




Different domain-decomposition configurations used for MPI-Parallel simulations

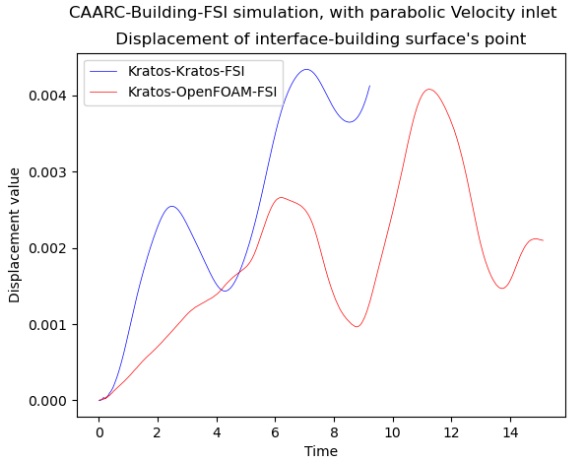
Lehrstuhl für Statik
Ingenieurfakultät Bau Geo Umwelt
Technische Universität München

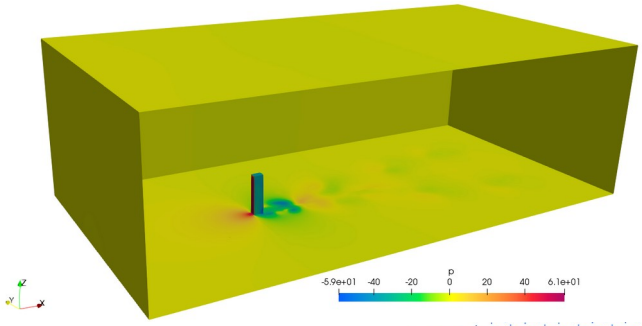
Verification Case



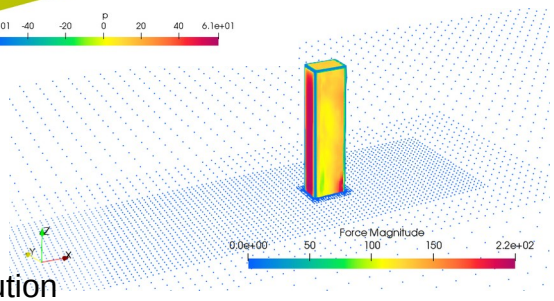
3D FSI case : CAARC Building Simulation

- OpenFOAM : PimpleFoam solver, SnappyHex Mesh, Groovy Velocity inlet B.C, etc.
- Kratos-StructuralMechanicsApplication.
- CoSimulation : Nearest-neighbor mapping.

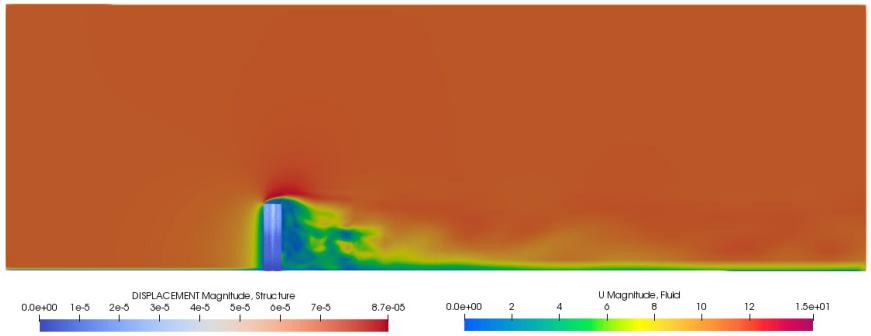




Pressure Distribution



Load Distribution



Displacement(structure) and Velocity(fluid) Distribution

Fig: Plots in CAARC Simulation with constant velocity inlet B.C.

Conclusion and future scope

Conclusion

- The Kratos-OpenFOAM coupling adapter : designed, developed, and validated.
- **KratosOpenfoamAdapterFunctionObject** on OpenFOAM‘s side.
- **Openfoam Wrapper** on CoSimulation‘s side.
- Communication : CoSimIO tool (file-based approach).
- Data access using OpenFOAM‘s ObjectRegistry and LookupClass methods.
- Only Explicit-Gauss-Seidel (or weak) coupling.
- Flexible in use and setup in OpenFOAM‘s *controlDict.*
- Extensible, easy to compile, independent of the OpenFOAM solver.
- Supports MPI-Parallel simulations.
- Supports multiple coupling interfaces.
- Useful for any OpenFOAM solver that supports dynamic meshes.

Future scope

- Implicit (or strong) coupling.
- To validate real-life FSI cases.
- To validate different OpenFOAM solvers, eg. Compressible solvers.
- Extension to other Multiphysics problems, eg. CHT.
- Performance study of MPI-Parallel simulations on a cluster.
- Running on a distributed memory environment.
- Use of other approaches for CoSimIO (MPI, Sockets, Pipe etc).

Thanks
for your attention !!

Ashish Darekar

ashishdarekar14@gmail.com

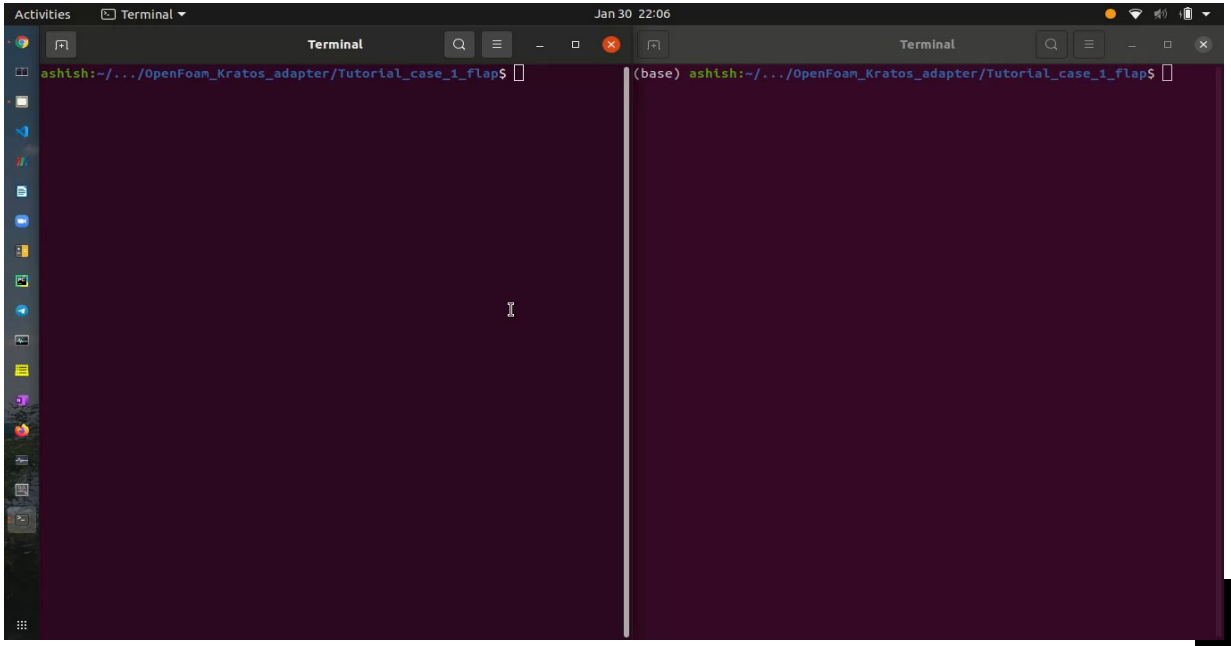
(+49)15129426202

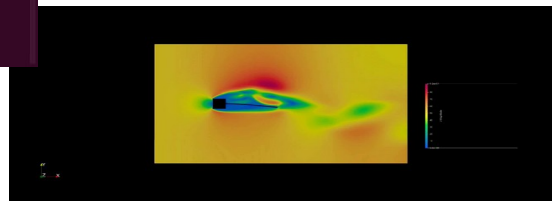
https://github.com/ashishdarekar/Kratos_OpenFOAM_adapter

Additional Information

Validation Case

FSI-Benchmark: Flow-induced vibration of a flexible beam or flap





Running a case using two terminals.
On the Left OpenFOAM and to the right KRATOS-CoSimulation

Build a function object

options

home > ashish > Documents > MS > Master_thesis > Ashish > Thesis > OpenFoam_Kratos_ada

```
EXE_INC = \
    -I$(LIB_SRC)/finiteVolume/lnInclude \
    -I$(LIB_SRC)/meshTools/lnInclude \
    -I$(LIB_SRC)/transportModels/ \
    -I$(LIB_SRC)/transportModels/incompressible/lnInclude \
    -I$(LIB_SRC)/transportModels/compressible/lnInclude \
    -I$(LIB_SRC)/transportModels/twoPhaseMixture/lnInclude \
    -I$(LIB_SRC)/transportModels/interfaceProperties/lnInclude \
    -I$(LIB_SRC)/transportModels/immiscibleIncompressibleTwoPhaseMixture/lnInclude \
    -I$(LIB_SRC)/thermophysicalModels/basic/lnInclude \
    -I$(LIB_SRC)/TurbulenceModels/turbulenceModels/lnInclude \
    -I$(LIB_SRC)/TurbulenceModels/compressible/lnInclude \
    -I$(LIB_SRC)/TurbulenceModels/incompressible/lnInclude \
    -I$(LIB_SRC)/triSurface/lnInclude \
    -I$(MPI_ROOT) \
    -I$(MPI_ROOT2) \
    -I$(HOME)/Documents/MS/CoSimIO/co_sim_io \
    -I$(HOME)/Documents/MS/CoSimIO \
    -L$(HOME)/Documents/MS/CoSimIO/bin

LIB_LIBS = \
    -lfluidThermophysicalModels \
    -lincompressibleTransportModels \
    -lcompressibleTransportModels \
    -lturbulenceModels \
    -lincompressibleTurbulenceModels \
    -lcompressibleTurbulenceModels \
    -limmiscibleIncompressibleTwoPhaseMixture \
    -lspecie \
    -lfileFormats \
    -lfiniteVolume \
    -lmeshTools \
    -lco_sim_io_mpi
```

```
(base) ashish:.../applications/solvers$ grep -r dynamicFvMesh.H
lagrangian/uncoupledKinematicParcelFoam/uncoupledKinematicParcelFoam.C:#include "dynamicFvMesh.H"
lagrangian/DPMFoam/DPMFoam.C:#include "dynamicFvMesh.H"
lagrangian/sprayFoam/sprayFoam.C:#include "dynamicFvMesh.H"
lagrangian/icoUncoupledKinematicParcelFoam/icoUncoupledKinematicParcelFoam.C:#include "dynamicFvMesh.H"
multiphase/interFoam/interMixingFoam/interMixingFoam.C:#include "dynamicFvMesh.H"
multiphase/interFoam/interFoam.C:#include "dynamicFvMesh.H"
multiphase/potentialFreeSurfaceFoam/potentialFreeSurfaceFoam.C:#include "dynamicFvMesh.H"
multiphase/interPhaseChangeFoam/interPhaseChangeFoam.C:#include "dynamicFvMesh.H"
multiphase/compressibleInterFoam/compressibleInterFoam.C:#include "dynamicFvMesh.H"
multiphase/cavitatingFoam/cavitatingFoam.C:#include "dynamicFvMesh.H"
multiphase/multiphaseInterFoam/multiphaseInterFoam.C:#include "dynamicFvMesh.H"
combustion/reactingFoam/rhoReactingFoam/rhoReactingFoam.C:#include "dynamicFvMesh.H"
combustion/PDRFoam/PDRFoamAutoRefine.C:#include "dynamicFvMesh.H"
incompressible/pimpleFoam/pimpleFoam.C:#include "dynamicFvMesh.H"
compressible/rhoCentralFoam/rhoCentralFoam.C:#include "dynamicFvMesh.H"
compressible/rhoPimpleFoam/rhoPimpleFoam.C:#include "dynamicFvMesh.H"
```