

Master's Thesis Presentation

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

Ashish Shankar, Darekar

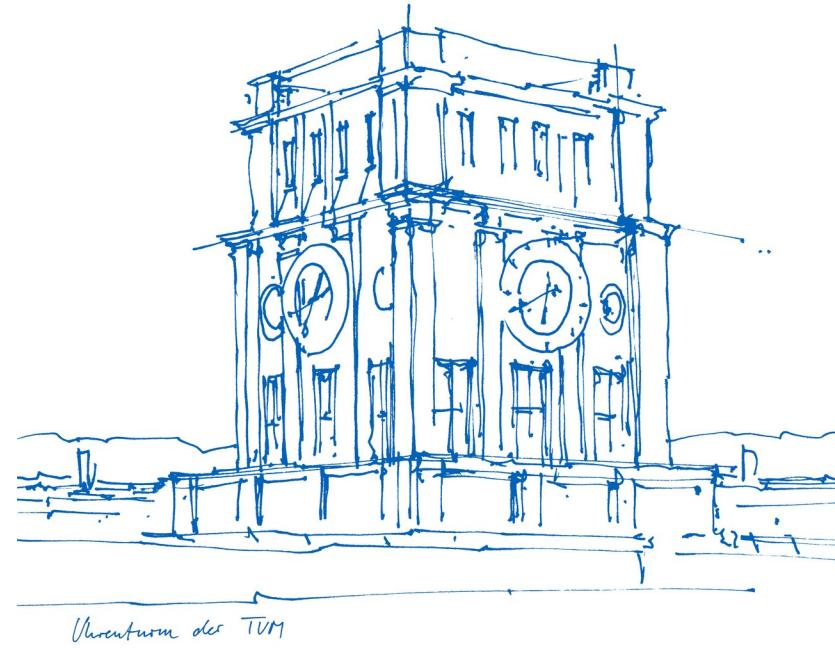
Chair of Structural Analysis

Technical University of Munich

Munich, 07. February 2022

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

Assistant advisor : Philipp Bucher



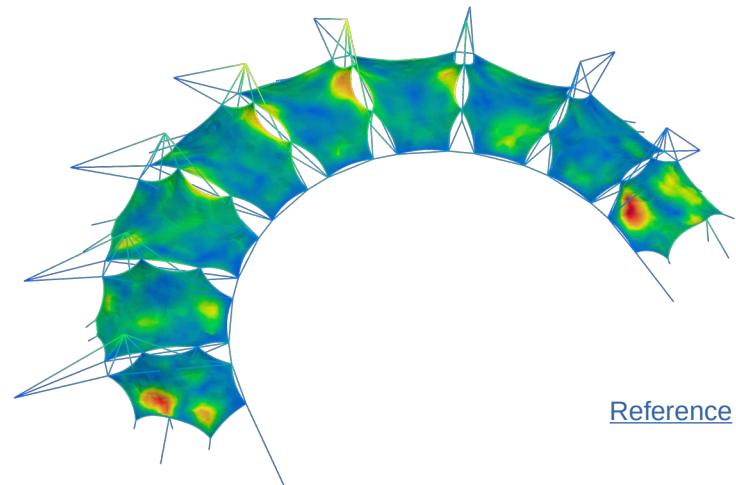
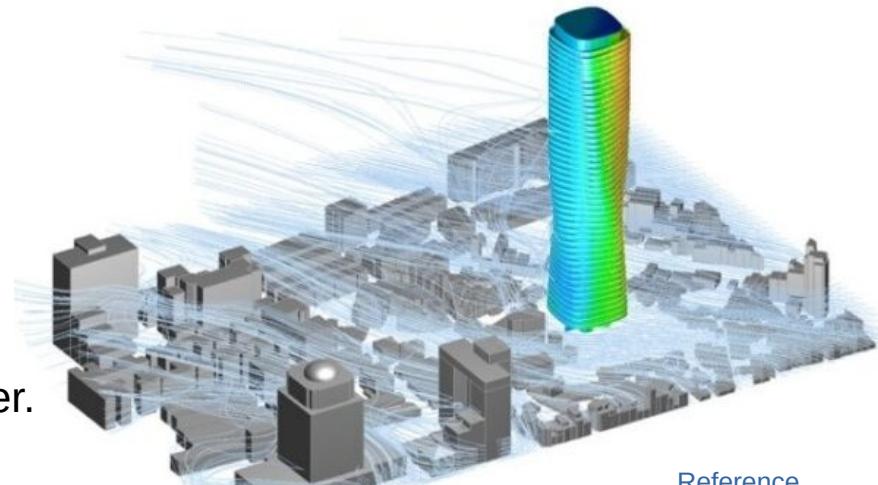
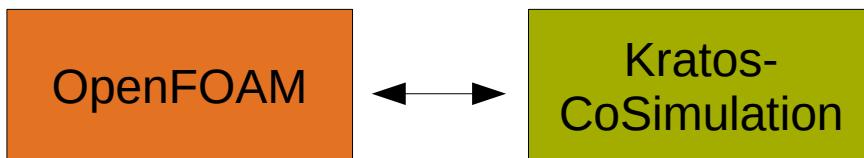
Overview

- 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

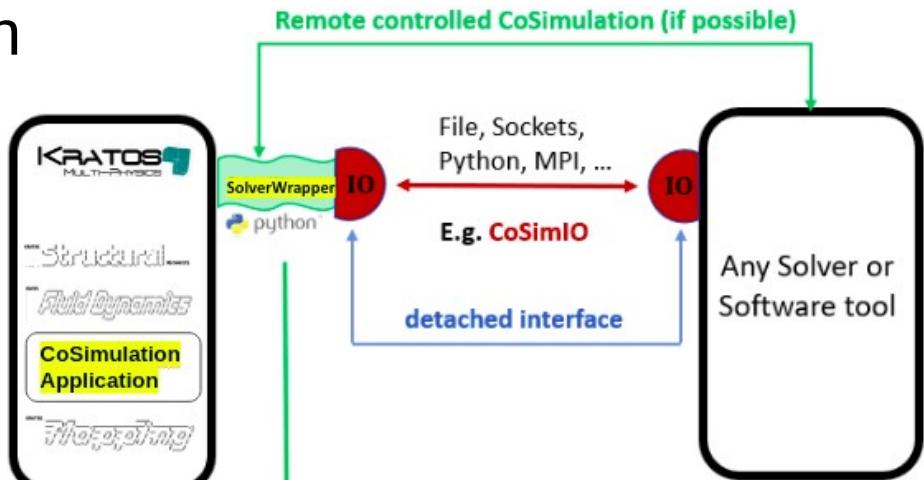
- Solving FSI problem using CoSimulation.
- Fluid solver: OpenFOAM.
- Structural solver: Kratos - StructuralMechanicsApplication or any other.
- Coupling OpenFOAM to CoSimulation.



Introduction

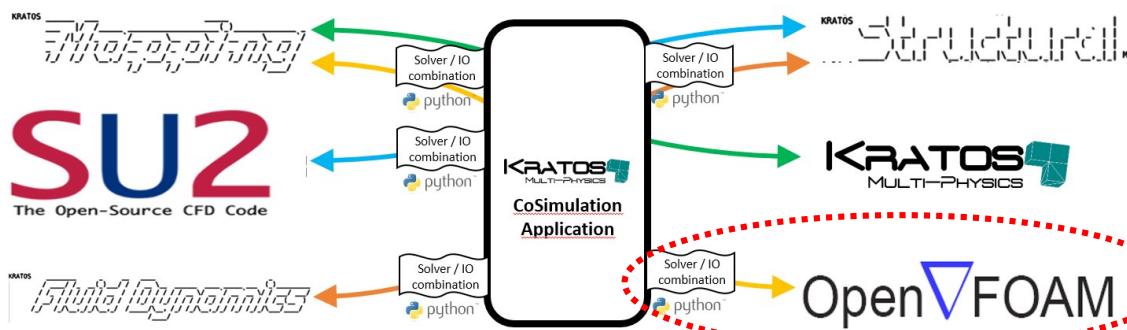
Kratos-CoSimulation Application

- Features :
- ✓ Partitioned 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



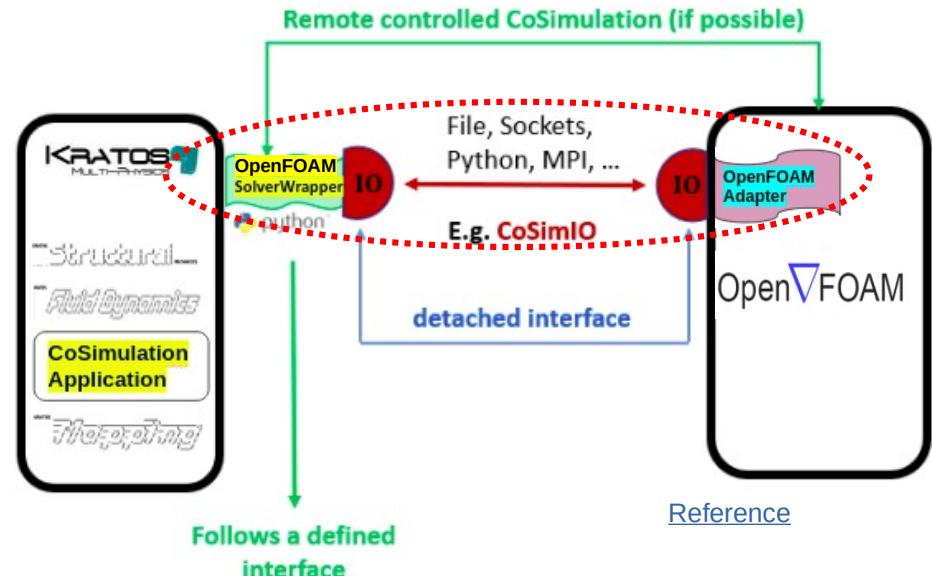
[Reference](#)

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?

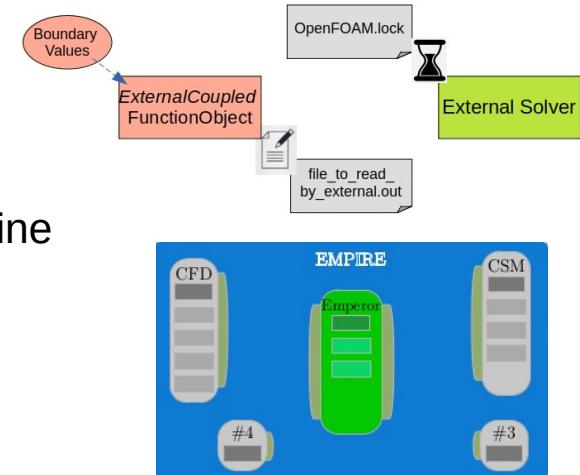
Literature Overview: Coupling tools and available OpenFOAM adapters

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

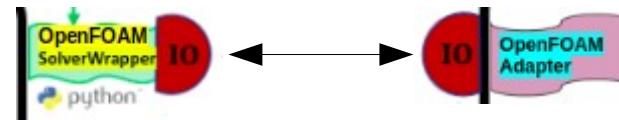
- Retrieves the appropriate data using the objects' registry.
- Communication - MPI or TCP/IP sockets.



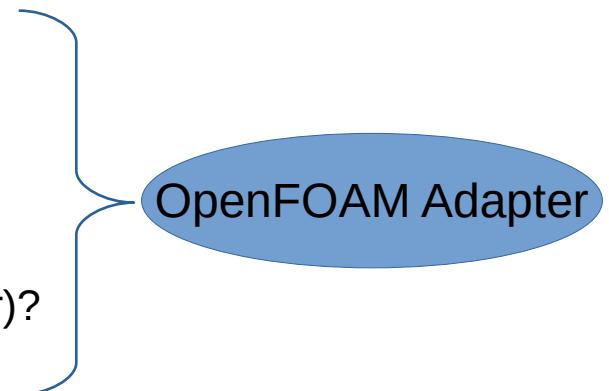
Implementation: Kratos-OpenFOAM coupling adapter and OpenFOAM Wrapper

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)?

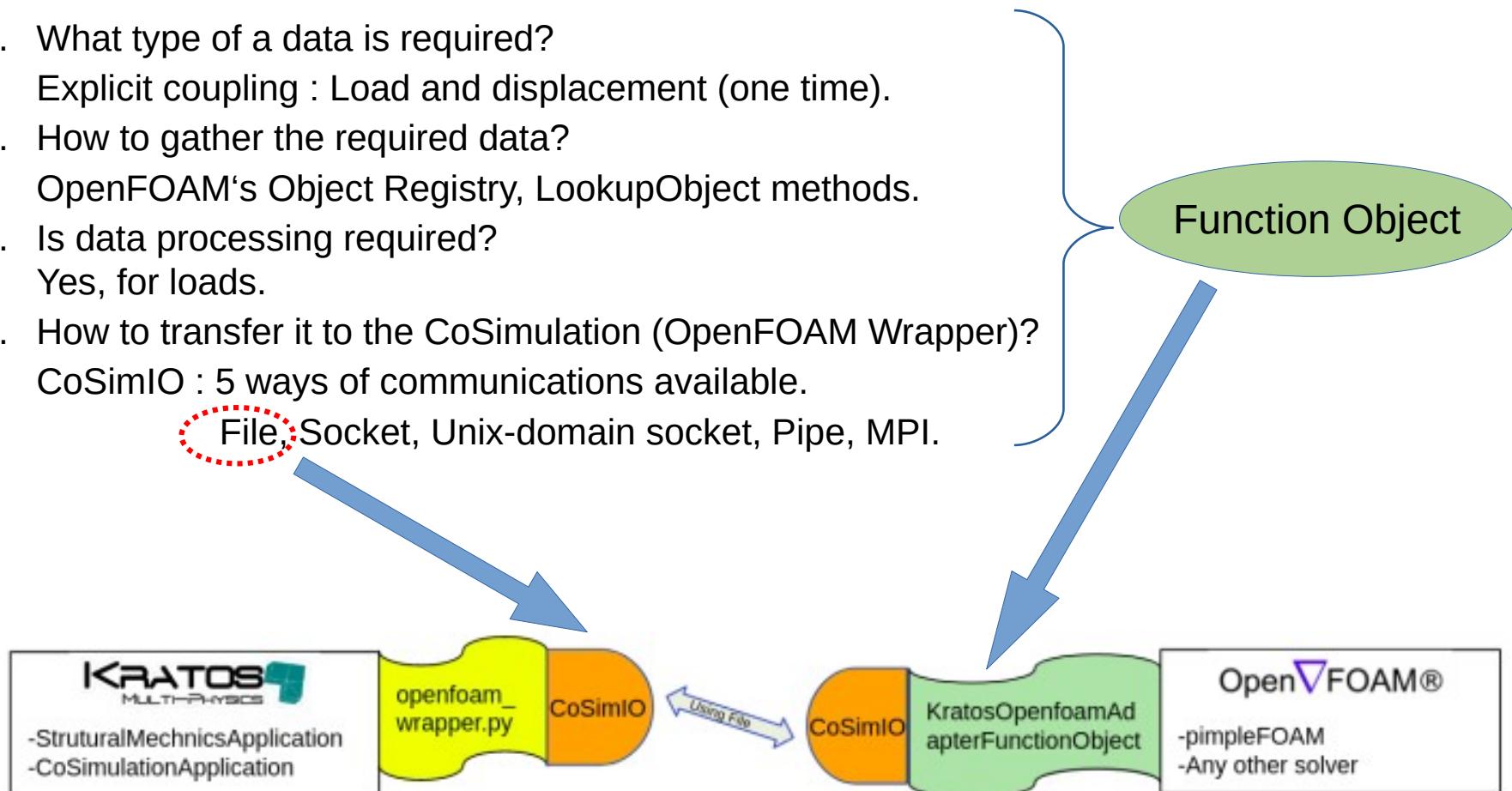


Implementation



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.



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.
- Multiple injection points in the solver.
- Easy to configure and extensible.

```

1 functions
2 {
3     yPlus1
4     {
5         type yPlus;
6         libs ("libfieldFunctionObjects.so");
7         patches (leftwall rightwall lowerwall);
8     }
9     CourantNo1
10    {
11        type CourantNo;
12        libs ("libfieldFunctionObjects.so");
13        executeControl      timeStep;
14        executeInterval     2;
15        writeControl        writeTime;
16    }
17 }
```

Function Object

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

Open∇FOAM®

Any solver

```

1  /* Start the solver */
2
3  Info<<"\nStarting time loop\n"<< endl;
4  while (runTime.run()) {
5      #include "readTimeControls.H"
6      #include "compressibleCourantNo.H"
7      #include "setDeltaT.H"
8
9      runTime++;
10
11     /* solve the equations */
12     #include "rhoEqn.H"
13     while (pimple.loop())
14     {
15         ...
16     }
17
18     runTime.write();
19 }
20
21 /* Finalize */
```

[Reference](#)

Implementation



Retrieve the data : ObjectRegistry

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



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

Open ∇ FOAM®

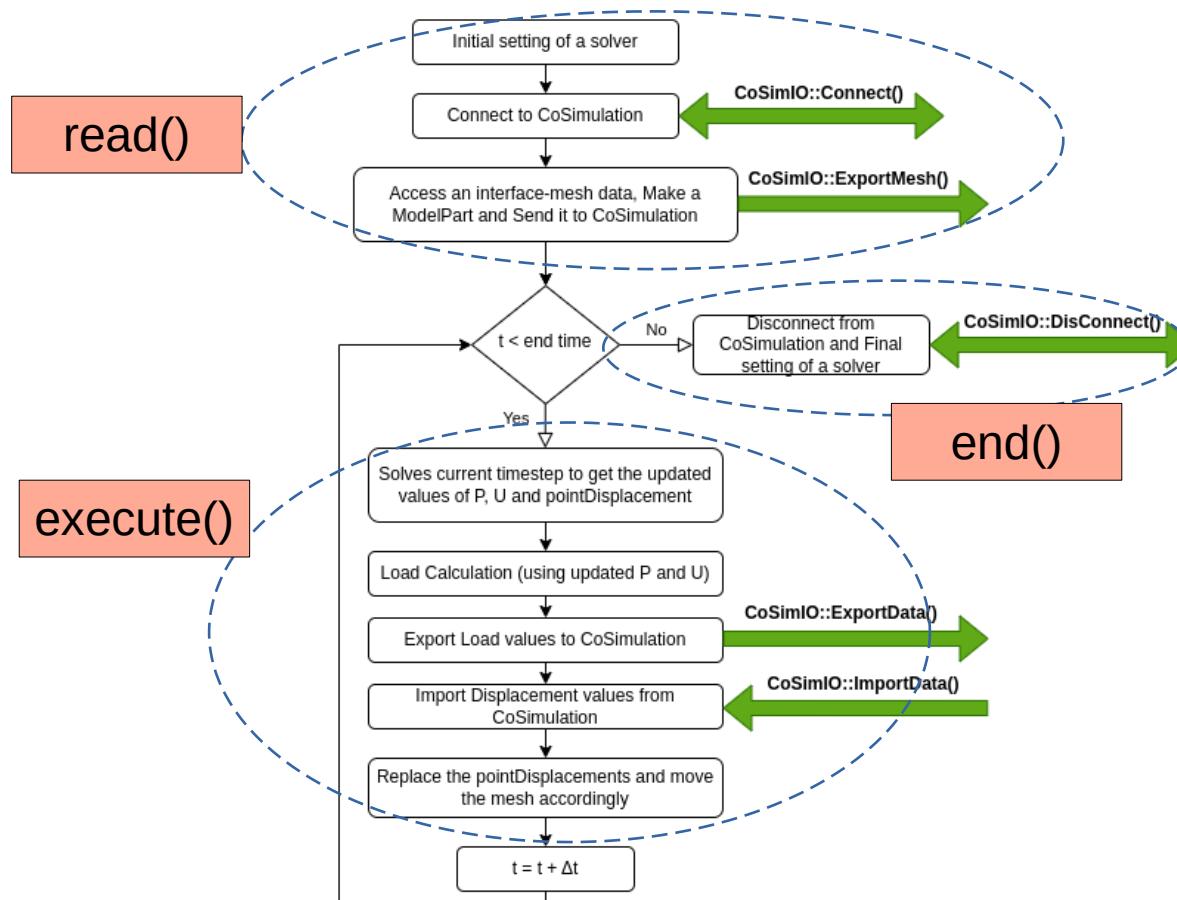
Solver's Data Base

runTime	//Time	(objectRegistry)
2 --->controlDict	//IOdictionary	(regIOobject)
3 --->FluidMesh	//fvMesh	(objectRegistry)
4 -->fvSchemes, fvSolution, transportProperties	//IOdictionary	(regIOobject)
5 -->points, faces, owner, neighbour, boundary	//PolyMeshData	(regIOobject)
6 -->pointZones, faceZones, cellZone	//PolyMeshData	(regIOobject)
7 -->T, U, p	//volDataFields	(regIOobject)
8 --->SolidMesh	//fvMesh	(objectRegistry)
9 -->fvSchemes, fvSolution, transportProperties	//IOdictionary	(regIOobject)
10 -->points, faces, owner, neighbour, boundary	//PolyMeshData	(regIOobject)
11 -->pointZones, faceZones, cellZone	//PolyMeshData	(regIOobject)
12 -->T, U, p	//volDataFields	(regIOobject)

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.

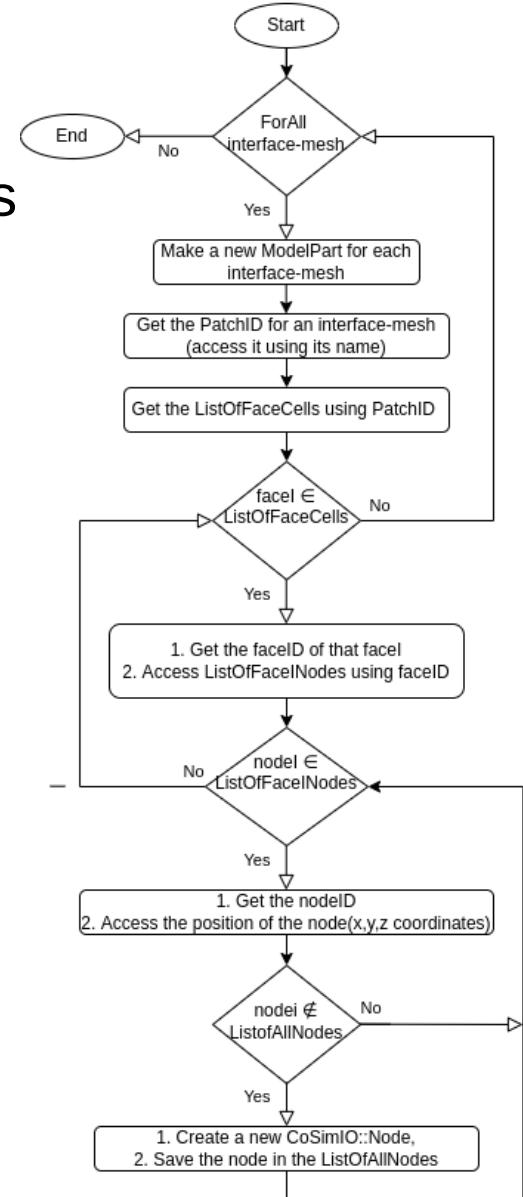
2. Use of Nodes: Advantages of Mapping

- 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

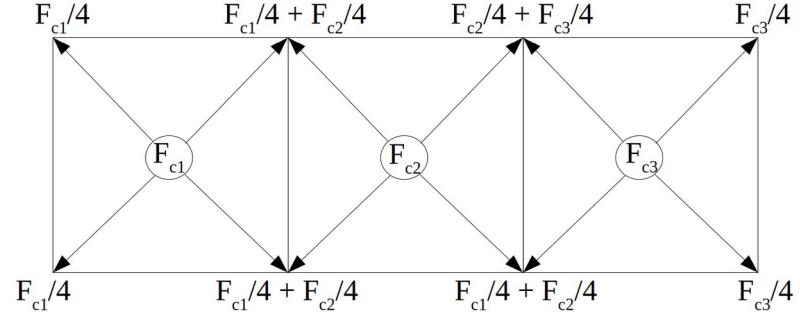
- At any point : Total force = Pressure force + viscous force.
- Pressure, velocity, flow properties : ObjectRegistry and lookupObject<>() methods.

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

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

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

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



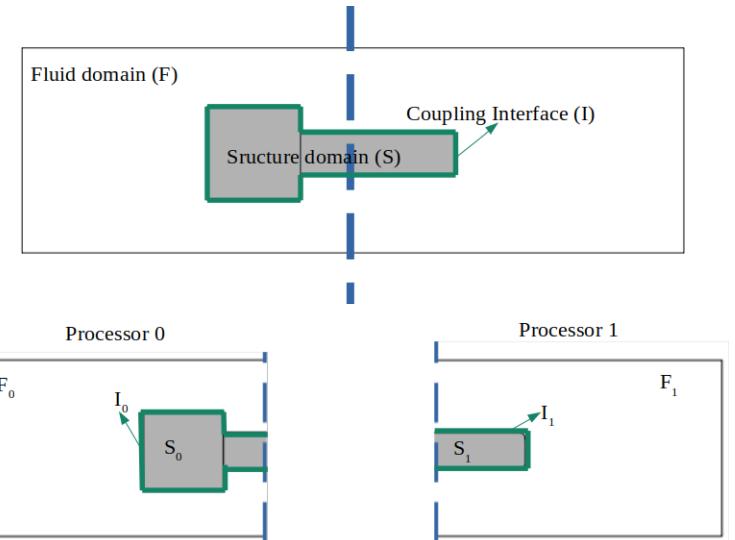
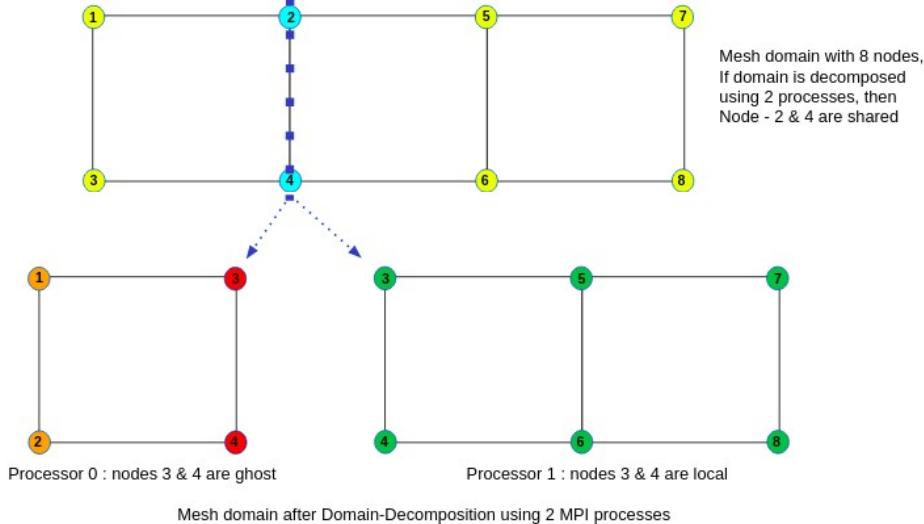
Implementation



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

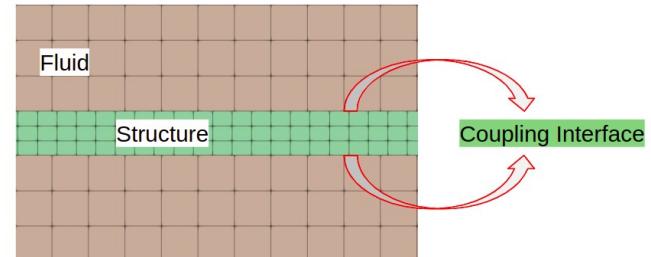
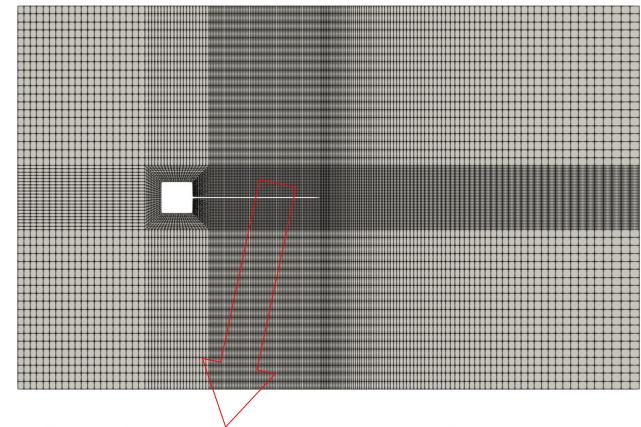
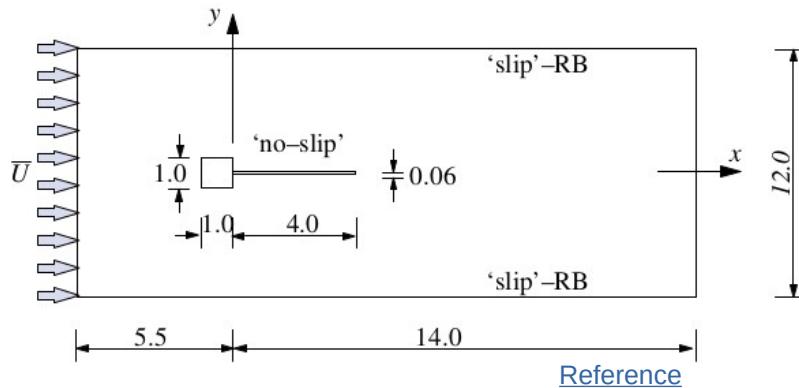
- External solver wrapper: Communication partner with the OpenFOAM adapter.
- Calls the specific functions of the CoSimulation.
- Important functions that need to be defined:
 1. *Initialize()*:
 - ✓ Read the input files, prepare internal data structures.
 - ✓ Formation of the coupling-mesh.
 2. *SolveSolutionStep()*:
 - ✓ **Import** load values and **Export** displacement values during each time step.



Results and Validation

Validation Case

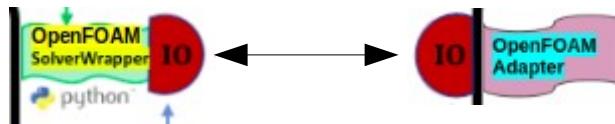
FSI-Benchmark: Flow-induced vibration of a flexible beam or flap



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

Validation Case

Configuration files



```

1 "solvers" :
2 {
3     <--> "Openfoam_Kratos_Wrapper"
4     {
5         "type" : "solver_wrappers.external.openfoam_wrapper",
6         "solver_wrapper_settings" : {
7             "import_meshes" : ["interface_flap"],
8             "export_data" : ["disp_interface_flap"],
9             "import_data" : ["load_interface_flap"]
10        },
11        "io_settings" : {
12            "type" : "kratos_co_sim_io",
13            "connect_to" : "Openfoam_Adapter"
14        },
15        "data" : {
16            "disp_interface_flap" : {
17                "model_part_name" : "interface_flap",
18                "dimension" : 3,
19                "variable_name" : "DISPLACEMENT"
20            },
21            "load_interface_flap" : {
22                "model_part_name" : "interface_flap",
23                "dimension" : 3,
24                "variable_name" : "FORCE"
25            }
26        }
27    }

```

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

```

1 KratosOpenfoamAdapterFunctionObject1 //User-specific name, can be changed
2 {
3     type KratosOpenfoamAdapterFunctionObject; //Type-name, do not change
4     libs ("..//KratosOpenfoamAdapterFunctionObject/
5         libKratosOpenfoamAdapterFunctionObjectFunctionObjects.so");
6         //Path to generated shared library file
7
8     interfaces //Key-word, do not change
9     {
10         Interface1
11         {
12             patches interface_flap; //Actual patch-name used by OpenFOAM
13             name interface_flap;
14             importDataIdentifier (disp_interface_flap);
15             exportDataIdentifier (load_interface_flap);
16         }
17         Interface2 //More interfaces can be added here in similar way like above
18         {...}
19     }
20
21     parameters //Key-word, do not change
22     {
23         rho rho [1 -3 0 0 0 0] 0.00118; //Density of a fluid (units are must)
24         nu nu [0 2 -1 0 0 0 ] 0.1542; //Kinematic viscosity (units are must)
25     }
26
27     //Additional parameters
28     //debugLevel info; // For debugging, Default value is info
29     //dim 3;           // Dimension of a problem, Default value is 3

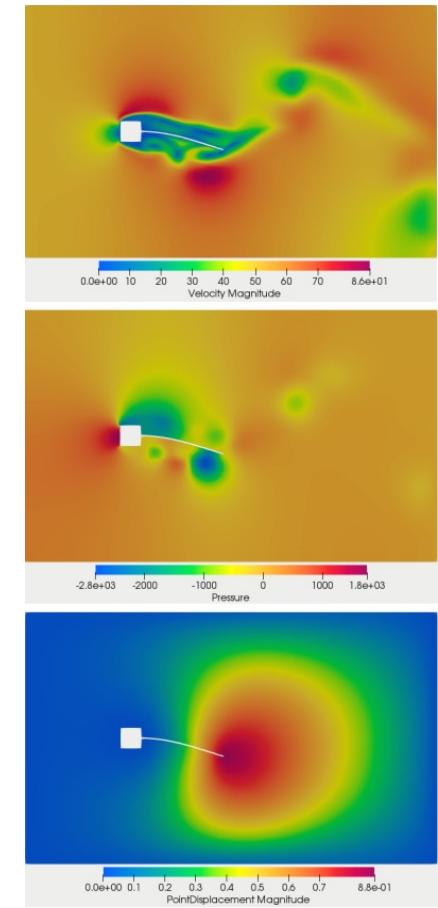
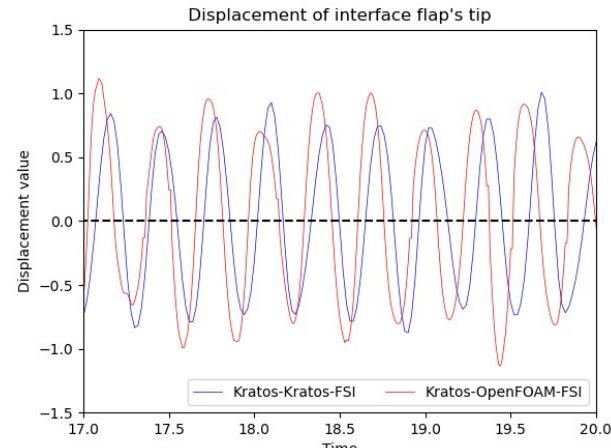
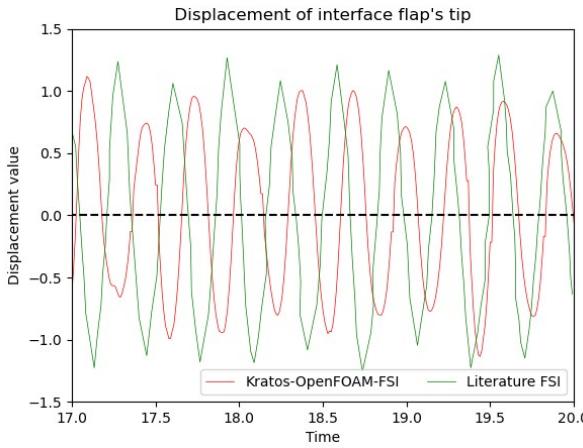
```

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

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.

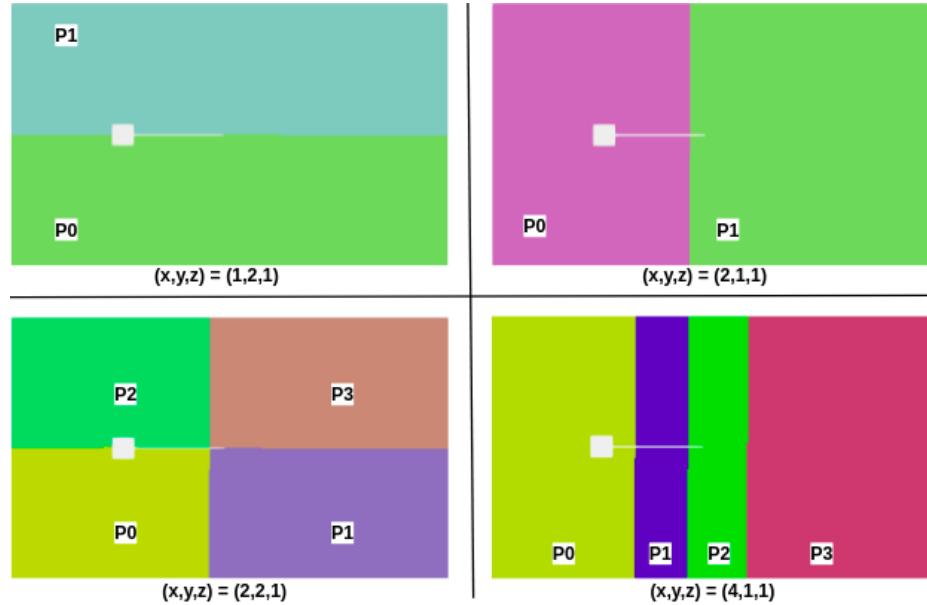
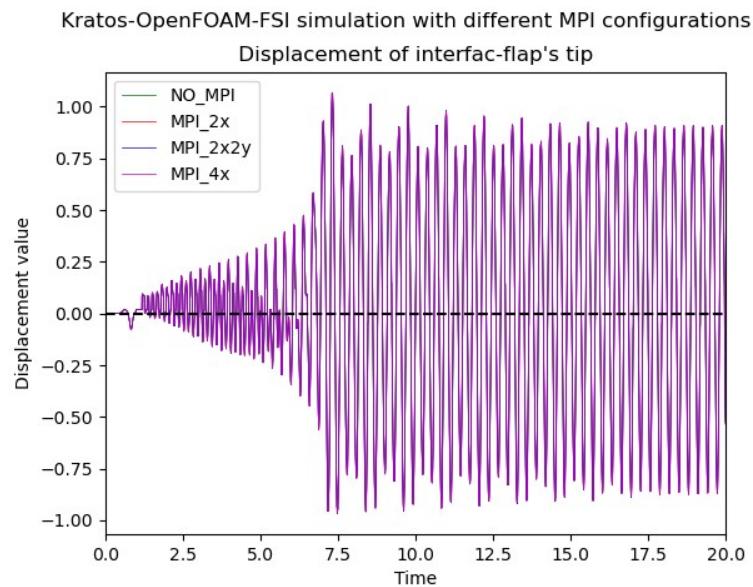


Velocity, Pressure, PointDisplacement

Results

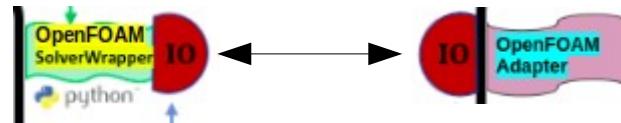
FSI-Benchmark: Flow-induced vibration of a flexible beam or flap

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



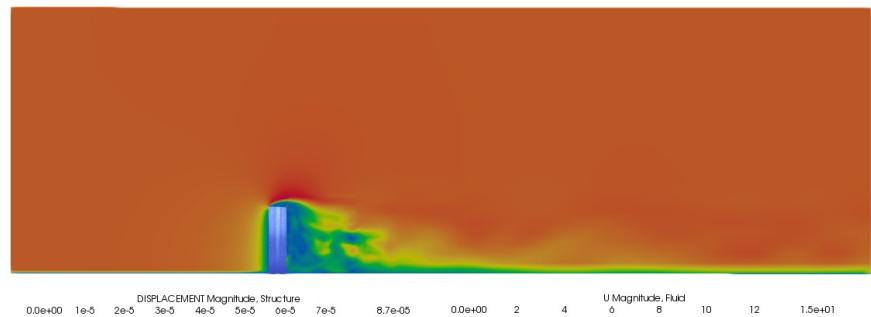
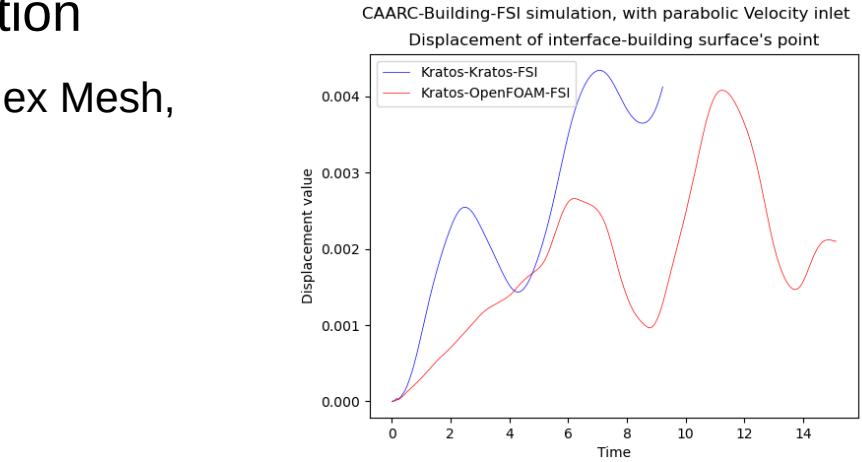
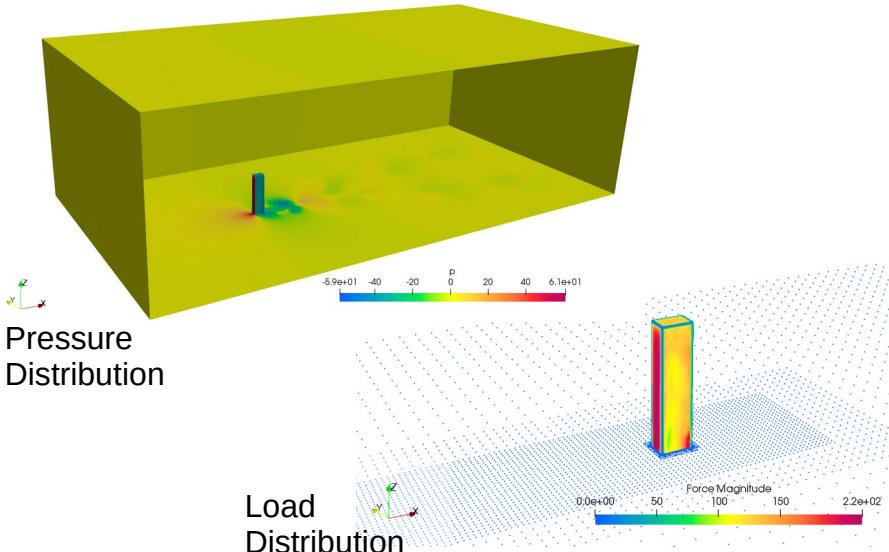
Different domain-decomposition configurations used for MPI-Parallel simulations

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.



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.
- 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 physics problems, eg. CHT.
- Performance study of MPI-Parallel simulations on a cluster.

Thanks
for your attention !!

Ashish Darekar

ashishdarekar14@gmail.com

(+49)15129426202



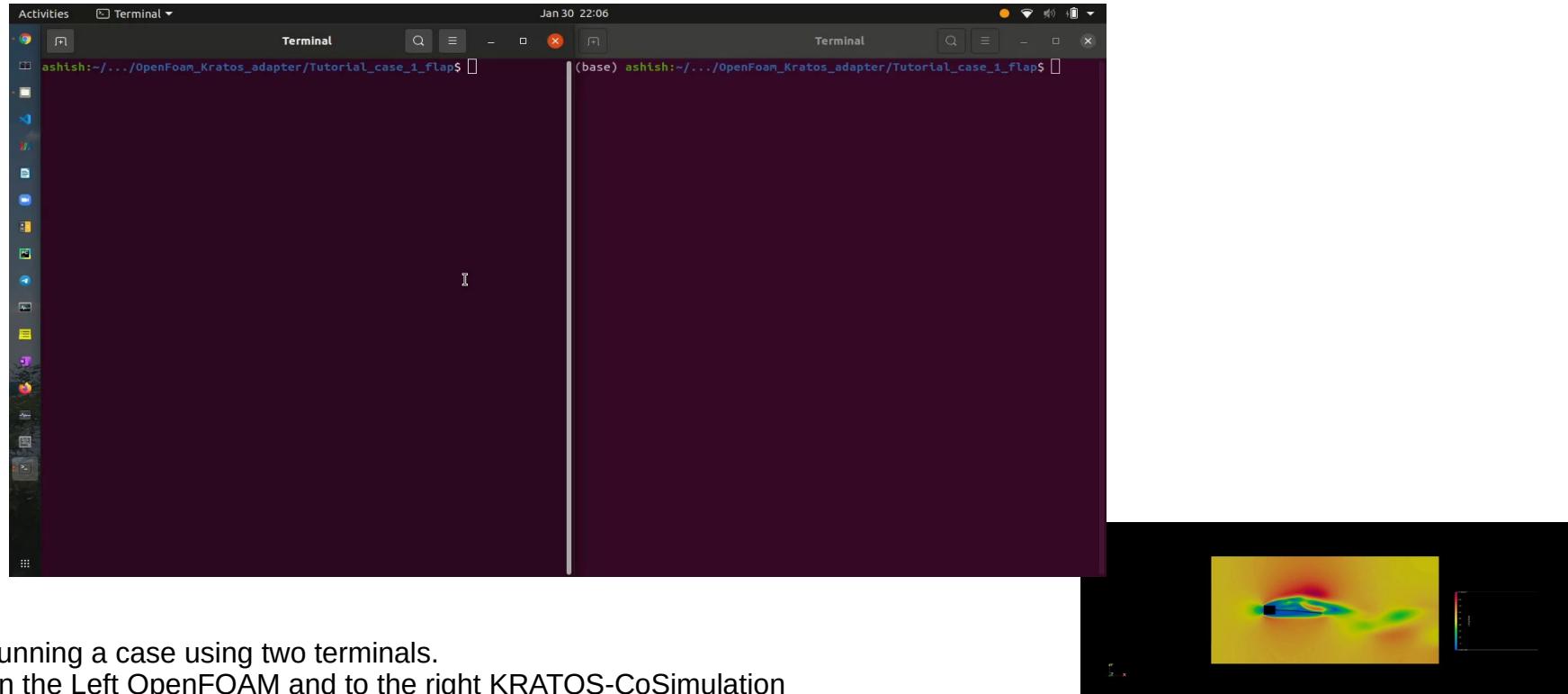
https://github.com/ashishdarekar/Kratos_OpenFOAM_adapter



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

1  EXE_INC = \
2    -I$LIB_SRC/finiteVolume/lnInclude \
3    -I$LIB_SRC/meshTools/lnInclude \
4    -I$LIB_SRC/transportModels/ \
5    -I$LIB_SRC/transportModels/incompressible/lnInclude \
6    -I$LIB_SRC/transportModels/compressible/lnInclude \
7    -I$LIB_SRC/transportModels/twoPhaseMixture/lnInclude \
8    -I$LIB_SRC/transportModels/interfaceProperties/lnInclude \
9    -I$LIB_SRC/transportModels/immiscibleIncompressibleTwoPhaseMixture/lnInclude \
10   -I$LIB_SRC/thermophysicalModels/basic/lnInclude \
11   -I$LIB_SRC/TurbulenceModels/turbulenceModels/lnInclude \
12   -I$LIB_SRC/TurbulenceModels/compressible/lnInclude \
13   -I$LIB_SRC/TurbulenceModels/incompressible/lnInclude \
14   -I$LIB_SRC/triSurface/lnInclude \
15   -I$MPI_ROOT \
16   -I$MPI_ROOT2 \
17   -I$HOME/Documents/MS/CoSimIO/co_sim_io \
18   -I$HOME/Documents/MS/CoSimIO \
19   -L$HOME/Documents/MS/CoSimIO/bin
20
21 LIB_LIBS = \
22   -lfluidThermophysicalModels \
23   -lincompressibleTransportModels \
24   -lcompressibleTransportModels \
25   -lturbulenceModels \
26   -lincompressibleTurbulenceModels \
27   -lcompressibleTurbulenceModels \
28   -limmiscibleIncompressibleTwoPhaseMixture \
29   -lspcgie \
30   -lfileFormats \
31   -lfiniteVolume \
32   -ImeshTools \
33   -lco_sim_io_mpi
34
```