Development of OpenFOAM-based libraries for massive parallel simulation of compressible turbulent flows in topologically changing meshes

Federico Piscaglia, Andrea Montorfano
Dipartimento di Energia, Politecnico di Milano

POLITECNICO DI MILANO

Outline

TOPICS:

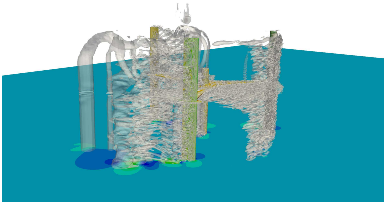
- automatic Mesh Motion with Topological Changes
  - `slidingInterface`, `layerAdditionRemoval`, `attachDetach`
  - parallelisation of Topological Changes
  - constrained decomposition
- novel compressible topoSolvers: `topoDyM` family
- scale-adaptive RANS/LES turbulence modeling: DLRM
- examples of applications to Engineering problems
- conclusions

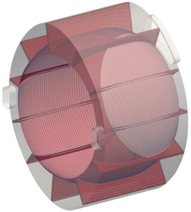
FRAMEWORK OpenFOAM®-2.4.x + in-house developed library

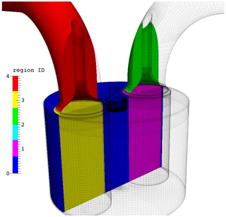
Prescribed mesh motion with topological changes

Domain shape is changing during the simulation according to a prescribed motion law:

- Motion is known and independent of the solution, usually only prescribed at boundaries
- Definition of moving mesh involves point position and mesh connectivity for every point and cell for every time-step of the simulation. This is typically defined with reference to a pre-processor or parametrically in terms of motion parameters (crank angle, valve lift curve, etc.)
- Solution-dependent mesh changes can be performed without affecting the motion: eg. mesh refinement





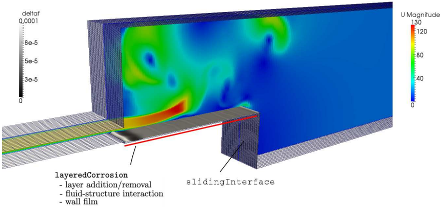


Solution-dependent mesh motion

External shape of the domain is unknown and a part of the solution:

- by definition, it is impossible to pre-define mesh motion;
- in all cases, it is the motion of the boundary that is known or calculated;
- automatic mesh motion determines the position of internal points based on boundary motion

Example: combustion of hybrid rocket motors (Aerospace)



Joint research with the *SPLab (Space Propulsion Laboratory)*, Aerospace Science and Technology Department, Politecnico di Milano (**Prof. Luciano Galfetti**)

Compressible dynamic solver for unsteady flows

Extended flow solver for dynamic moving grids (`rhoPimpleDynFoam`):

$$\frac{\partial}{\partial t}\int_V \rho u_i dV + \int_S \rho u_i \left(\vec{v} - \vec{v}_b\right)\vec{n} dS = \int_S \left(\tau_{ij}\vec{i}_j - p\vec{i}_i\right)\vec{n} dS + \int_V \vec{b}_i dV$$

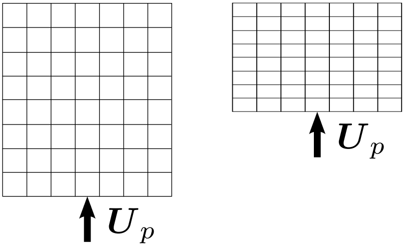
- when the location of the grid is known as a function of time, solution of the NS equations must account for convective fluxes, using the relative velocity components at the cell faces
- convective term is the “link” between the conservation equations and the mesh motion solver
- conservation of mass and energy with moving faces is not necessarily ensured if the grid velocity is not used to calculate mass fluxes

Some extensions to the basic solver are needed:

- a correction is applied to mass fluxes over the faces to enforce mass conservation for strong deformations of the grid within the timestep
- enhanced calculation of the contribution of the relative fluxes in the mesh motion solver, to make the solver formulation independent by the mesh motion strategy applied
- improved control on under-relaxation factors with transient solvers applied to problems with moving grids

Mesh motion with **cell deformation/stretching** or with **multiple meshes (remapping)**:

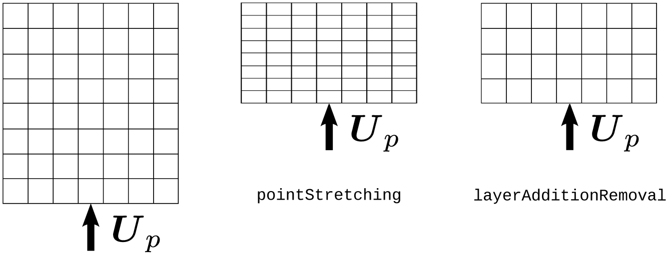
- variations in the mesh size Δx affect the discretization error and the numerical accuracy;
- if the cutoff length $\bar{\Delta} x$ is tied to Δx, mesh refinement will also induce a decrease in $\bar{\Delta}$ and make the subgrid model less influential on the results;
- how filtering behaves as cells increase in their mesh size is not trivial



- A dynamic variation of the filter size requires to find an **error estimate** and a **bound** for the algorithm, in order to guarantee a sufficient resolution of the grid to resolve the main turbulent scales of the engine

Topological changes in deforming grids

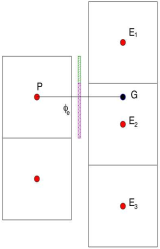
Development of a new accurate methodology for LES based on the so-called `topologyModifiers` to keep the filter size unchanged during mesh motion.



FEATURES:

- variations in the mesh size Δx are limited and localized on a small number of cells: filter width almost constant despite the mesh is moving!
- overall number of cells varies during the engine cycle
- SGS model less influential on the results since it is not interacting with mesh motion

AMI



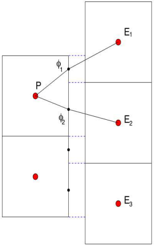
$$\phi_e = \phi_G \alpha + (1-\alpha)\,\phi_P$$

$$\alpha = \frac{x_e - x_P}{x_G - x_P}$$

$$\phi_G = \sum_i \alpha_i \phi_{E_i}$$

$$\alpha_i = \frac{A_i}{A}$$

slidingInterface



$$\phi_{e_1} = \phi_{E_1}\alpha_1 + (1-\alpha_1)\,\phi_P$$

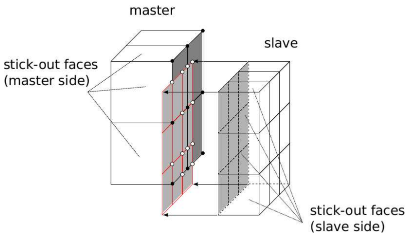
$$\phi_{e_2} = \phi_{E_2}\alpha_2 + (1-\alpha_2)\,\phi_P$$

$$\alpha_i = \frac{x_{e_i} - x_P}{x_{E_i} - x_P}$$

For extreme cases of mesh motion, changing point positions is not sufficient to accommodate boundary motion and preserve mesh quality.

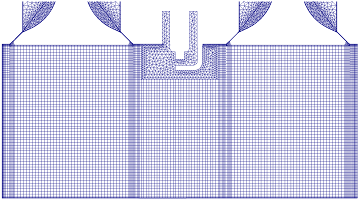
- Topological Changes on Polyhedral Meshes:
- Definition of a topological change: number or connectivity of points, faces or cells in the mesh changes during the simulation
  - `slidingInterface`, `layerAdditionRemoval`, `attachDetach`
- Motion can be handled by the FVM with no error (moving volume), while a topological change requires additional algorithmic steps

`dynamicMesh`: non-conformal interfaces

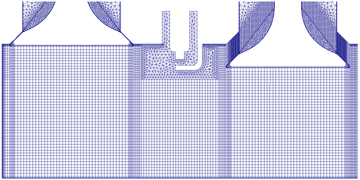


- **SUPERMESH APPROACH** (`AMI`): vertex-based solution using a bounded Galerkin projection over a virtual triangulated surface mesh
  - allows to avoid updates in mesh change every time step;
  - optimized solution with full overlap of fluid regions;
  - not stable with partial overlap of mesh regions.
- **TARGET MESH APPROACH** (`slidingInterface`): coupling/decoupling of mesh regions based on polyhedral cells support.
  - updating the mesh topology updated anytime the algorithm operates;
  - no specific treatment for interpolation of fluid-dynamic quantities over the supermesh.

Adaptive LES of ICE



Time: -66 CA-deg ATDCE



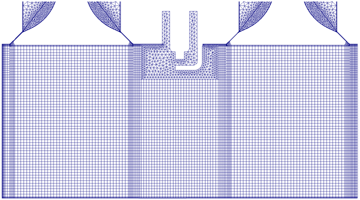
Time: 302 CA-deg ATDCE

Examples of application on the TCC Engine. Dynamic addition/removal of cell layers during:

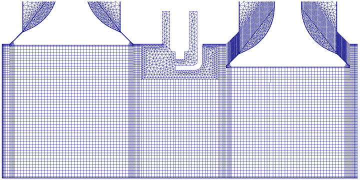
- *piston motion*: cell layer over the piston head is removed/added during compression/expansion
- *valve motion*: cells are added/removed both in the valve seat and on the valve bottom

A wide set of tools for case setup (automatic extraction and automatic decomposition over multiple processors of `faceSets/cellSets`) for dynamic layerAR has been implemented.

Adaptive LES of IC engines



Time: -66 CA-deg ATDCE



Time: 302 CA-deg ATDCE

- grids at different time steps differs only for layers of hexahedral cells;
- initial *skewness* and *non-orthogonality* preserved during the whole engine cycle;
- almost no cell deformation and no re-meshing are present;
- SGS filter cell size does not change during the engine cycle

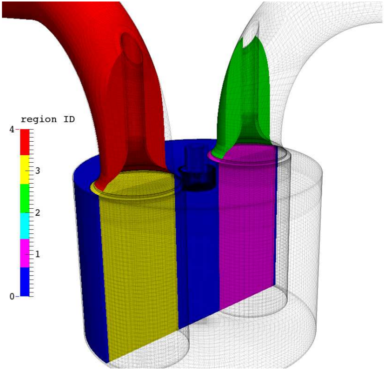
Moving mesh based on topological changes

ADVANTAGES:

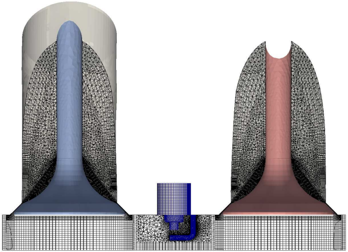
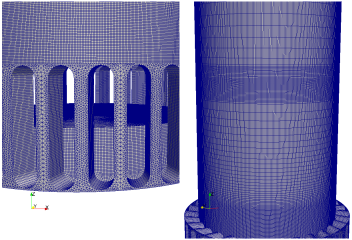
- decoupling of mesh morphology in different mesh regions
- initial mesh quality preserved during the all simulation
- high quality results
- faster convergence of the solver
- significantly reduced simulation time

DRAWBACKS:

- requires advanced numerical solvers
- not-easy initial setup



Adaptive LES of IC engines

WHAT IS NEEDED:

Automatic Mesh Motion with Parallel Topological Changes

- `slidingInterface` to handle non-conformal interfaces
- `layerAdditionRemoval` of multiple layers of cells
- `attachDetach` to simulate valve opening/closure event

Solver

- `motionSolver` for topologically changing meshes
- `topoEngineFoam` to handle topologically changing and moving meshes

`dynamicMesh` class: novel development

Dynamic mesh handling is available and well established in the official version of OpenFOAM:

- Grid points moved by means of an automatic mesh motion solver
- Mesh to mesh interpolation
- `AMI` (Arbitrary Mesh Interpolation)
- `layerAdditionRemoval` (basic features)
- `snappyHexMesh`: automatic mesh generator

Implementation and operation of **dynamic mesh handling BASED ON TOPOLOGICAL CHANGES** is strictly dependent on the mesh handling strategy of the code:

- **foam-extend-3.1** (released by the *Extend Community*): *mesh definition contains all the topological changes performed during the simulation* as a set of faces, cells and points labeled as "inactive".

- **OpenFOAM** (released by the *OpenFOAM Foundation*): *mesh definition contains the topology of the current calculation only*. Additional information about the topological changes is stored separately → official releases by OpenCFD are not configured to allow for the decoupling of the mesh through an interface.

`dynamicMesh` class: novel development

Dynamic mesh handling is available and well established in the official version of OpenFOAM®:

- Grid points moved by means of an automatic mesh motion solver
- Mesh to mesh interpolation
- `AMI` (Arbitrary Mesh Interpolation)
- `layerAdditionRemoval` (basic features)
- `snappyHexMesh`: automatic mesh generator

In ... ES
is

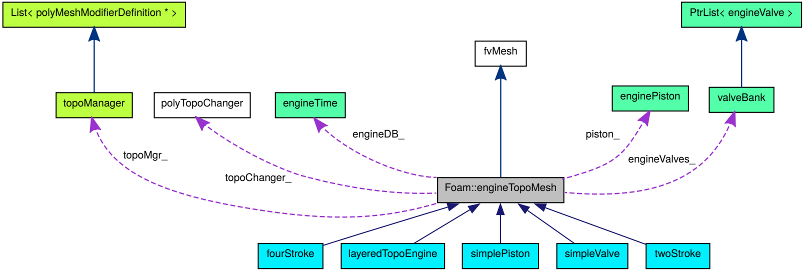
Author's choice: implementation compatible with OpenFOAM® by OpenCFD®

REQUIRED FEATURES:

- very general implementation → must be compatible:
  - to any distribution of OpenFOAM® (released by the Foundation) also for non-engine applications (spray nozzles, extAero, renewable energy, etc);
  - with any solver/application/utility (VOF, multiphase,...) already available in OpenFOAM®, without modifications.

NOTE: a *similar* theory (but completely different technology, implementation and operation) is available in *foam-extend-3.1*; **porting is not possible**.

dynamicMesh class: requirements (1)



FLEXIBILITY

- point motion algorithm must be as general as possible;
- extension to 'new' components must require little programming;
- implementation must be transparent to the final user

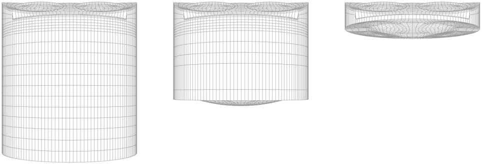
EFFICIENCY (→ paper available soon for download at https://imem.cray.com/)

- small overhead on the overall computation
- must run in parallel and provide good scalability

`dynamicMesh` class: requirements (2)

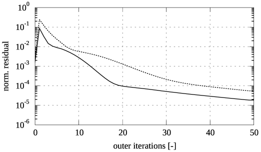
ACCURACY

- cell quality must not degrade as grid changes;



CA = -180° ATDC　　CA = -90° ATDC　　CA = 0° ATDC

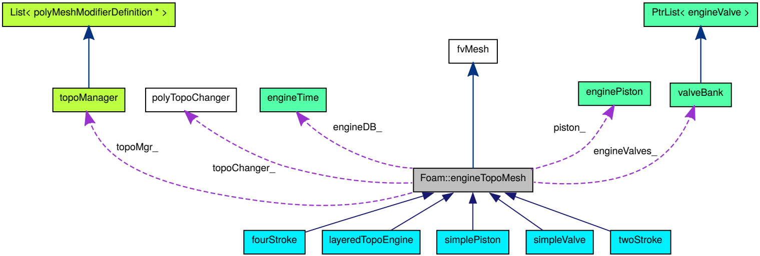
- it must coupled with a robust and accurate flow solver



SPEED

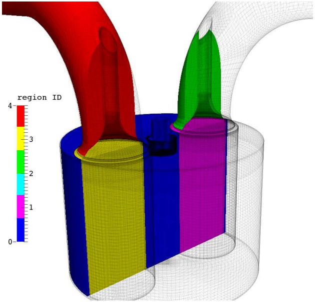
- it must be very fast!

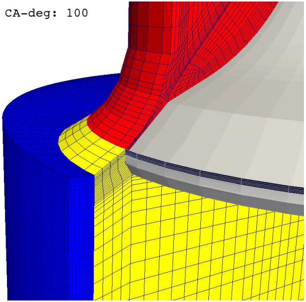
dynamicMesh **class: inheritance and collaboration**



- management of topoChanges (definition, parallelization/synchronization, variables interpolation) implemented at low-level (class `topoManager`)
- `engine` class: any extension can be easily done by adding new physical components (valves, ports)
- implementation of new 'components' requires only the **point motion law**
- mesh motion functionality supported by ALL the solvers of the code

Test case: TCC engine





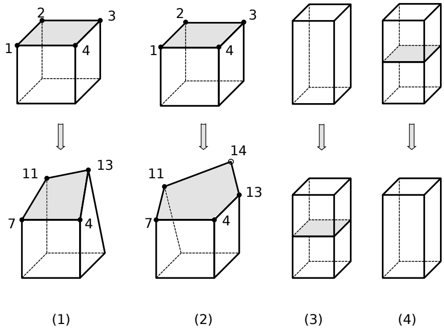
http://www.sandia.gov/ecn/engines/engineFlows/TCCEngine/engGeo.php

- spark-ignition engine, set up at the University of Michigan
- a two-valve head and a pancake-shape combustion chamber
- optical data available through the Engine Combustion Network (ECN)

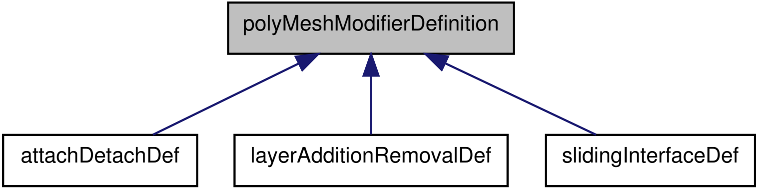
The TCC optical engine represents a perfect test-case for the validation of:

- algorithms for moving mesh (`slidingInterface`, `layerAdditionRemoval`)
- models and solvers for LES

Dynamic grids with topological changes



1. Point motion without topological change
2. Point insertion (removal)
3. Face insertion (cell split)
4. Face removal (cell merge)



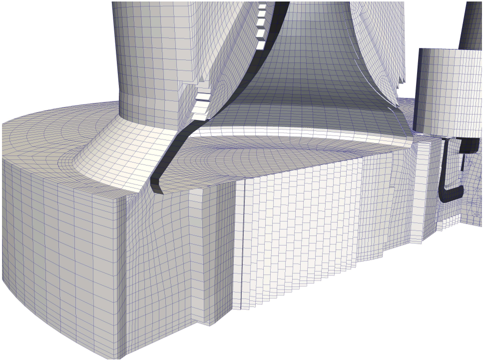
Topology modifiers

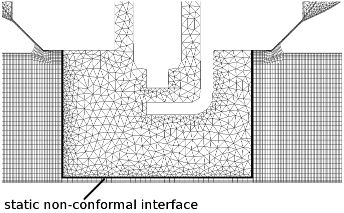
1. sliding interface
2. layer addition/removal
3. attach/detach boundary

Revised structure of the dynamic mesh class

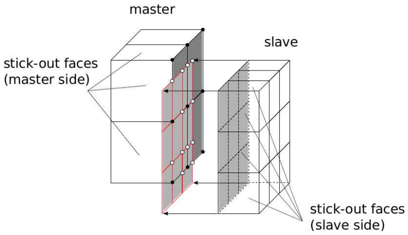
- topological changes completely transparent to the mesh motion solver
- automatic *case setup* with topological changes
- `topoManager` class: very flexible and fully object-oriented implementation

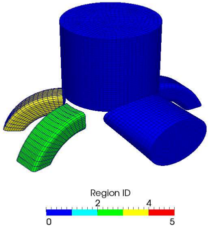
Disconnected, but adjacent, mesh domains can be stationary or move relative to one another.





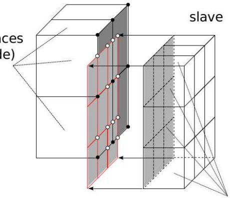
- tested on **IC Engines**, **rotating machinery**, **external aerodynamics** and on several simulations involving **partially/fully overlapping mesh regions**.
- only possible way to simulate relative motion between partially overlapping regions connected by non conformal-interfaces.

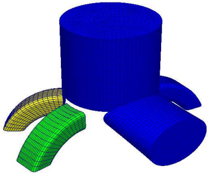




`slidingInterface` is a technique that allows simulation across disconnected, but adjacent, mesh domains, that can be stationary or move relative to one another.

- When coupling the interface, old topology must be stored for later decoupling;
- **Decoupling of the interface is not supported in the official distribution of the code**, that does not retain any information about removed entities: hence, interface coupling is always irreversible;

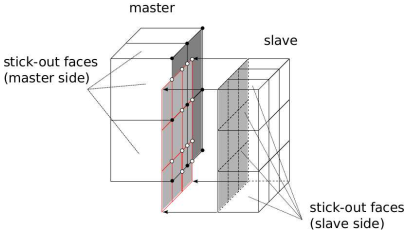


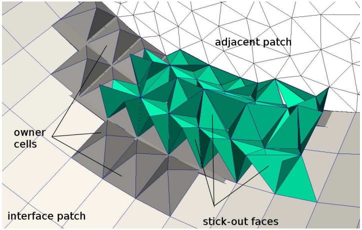


`slidingInterface` is a technique that allows simulation across disconnected, but adjacent, mesh domains, that can be stationary or move relative to one another.

- mesh regions are connected through `non-conformal interfaces`, preserving the overall mesh quality;
- Connection algorithms works automatically during mesh motion;
- Second order interpolation of face-fluxes over the interface;
- fully parallelised and integrated into boundary patch classes.

The present implementation works only with the mesh definition of the code released by the OpenFOAM® Foundation. (→**SAE 2013-24-0027**)

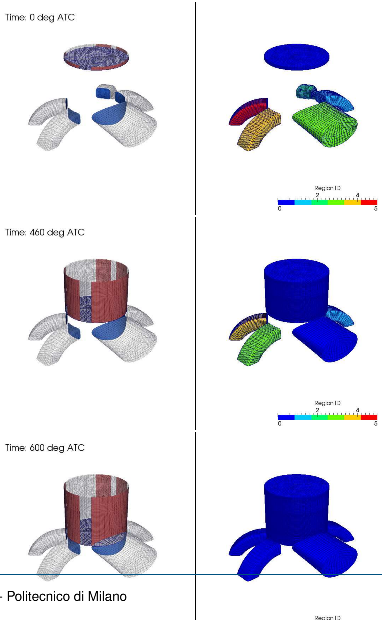




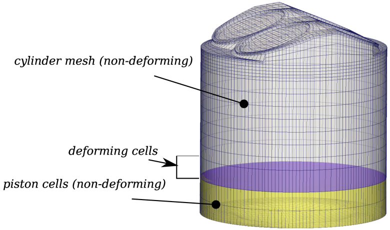
Recent development and improvements:

- improved robustness of the algorithm when non-conformal interfaces are generated through Third-Party software
- improved calculation of mesh fluxes during point merging/splitting, for enhanced conservation of the variables
- novel algorithm for the calculation of `stickOut` faces, based on their `sharedPoints` → improved stability with very complex/hybrid non-conformal interfaces (example: spark-plug)
- **low-level definition of topology modifiers** → topological changes are transparent to the user during novel mesh motion solver implementation

slidingInterface: two stroke engines



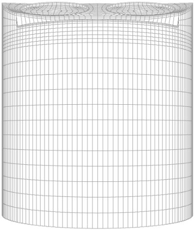
```
layerAdditionRemoval
{
  piston
  {
    layerFacesName pistonFaces;
    cellSetName   pistonCells;
    minLayerThickness 0.5e-3;
    maxLayerThickness 1.2e-3;
  }
}
```



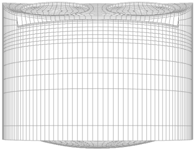
The `layerAdditionRemoval` mesh modifier:

- adds/removes dynamically one or more layers of cells as a consequence of a moving boundary (piston, valves)
- only one layer of cells undergoes actual deformation: **global mesh quality is preserved**
- Deforming layer does not have to lie on the moving boundary: constant aspect ratio of near-wall cells

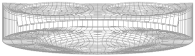
layerAdditionRemoval



CA = -180° ATDC



CA = -90° ATDC

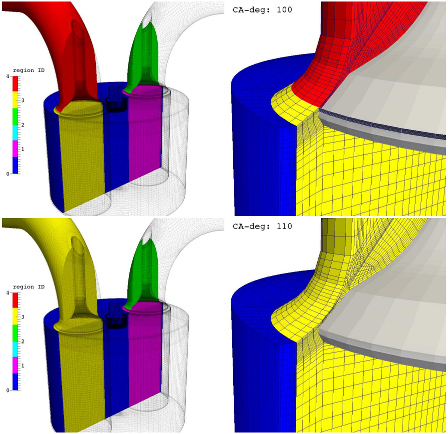


CA = 0° ATDC

Extensions to the official version of the `layerAdditionRemoval` class:

- **variable topology-driven time-stepping** to ensure grid consistency
- `decomposePar`: extension of the `scotch` algorithm for **automatic decomposition** of the mesh, to comply with the constraints of the topology modifiers
- run-time update of faceZones crossing a cellZone where layerAR is triggered
- synchronization of layerAR through processor boundaries
- **checking for boundary proximity**: automatic deactivation near the physical mesh boundaries to prevent topological inconsistency

`attachDetach` mesh modifier is applied to simulate the valve closure event and it consists in a reversible interface between two conformal mesh regions.



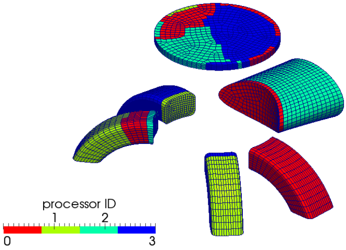
Enhancements:

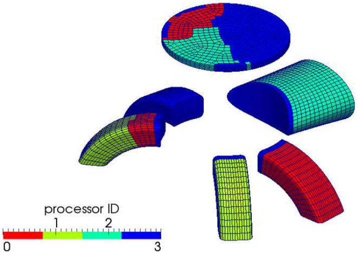
- extension: face matching calculated on the basis of point projection (strategy based on the `slidingInterface` algorithm)

Parallel decomposition

Topological changes cannot occur across a processor interface:

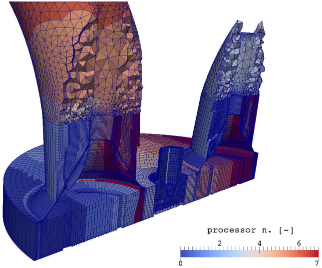
- each sliding patch pair must be on the same processor mesh
- the same constraint applies for attachDetach
- in layerAR region all processor patches must be perpendicular to cutting faces
- Domain decomposition has to be complemented with new algorithms to account for the added constraints





Extended version of `decomposePar`

Parallel decomposition



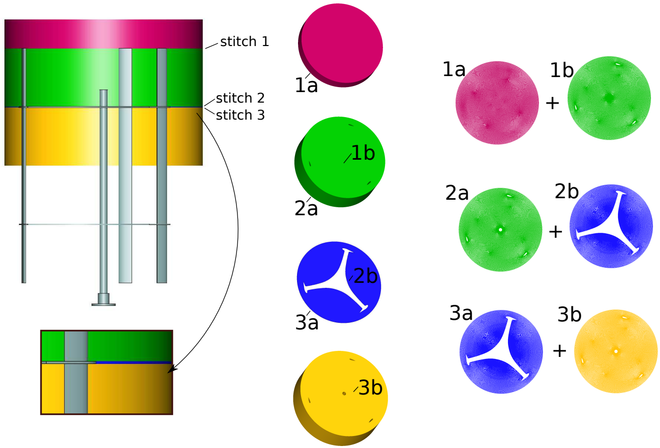

```
singleProcessorFaceSets
(
    (sliding-exhValveA 1)
    (sliding-exhValveB 1)
    (sliding-inValveA 2)
    (sliding-inValveB 2)
    (exhValve-detachFaces -1)
    (inValve-detachFaces -1)
);

layerARdecomp
{
    cylinderCells
    {
        faceSet  pistonFaces;
    }
}
```

Constrained decomposition

- Best results using automatic decomposition algorithms (METIS, Scotch)
- Modified **decomposePar** with layer AR decomposition
- Fully automatic setup provided by shell scripts

Vertical-Axis Wind Turbines (VAWT)

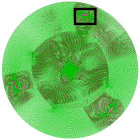
- Axial blocks have non-conformal interfaces (greater flexibility)
- Static non-conformal interfaces generated by **stitching**

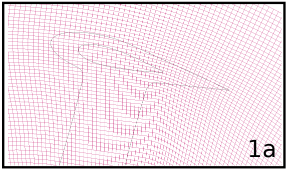


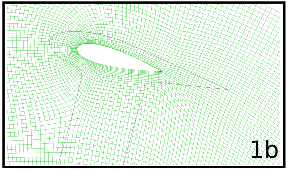
Stitch 1: turbine upper part

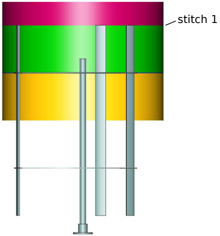
Stitch 1

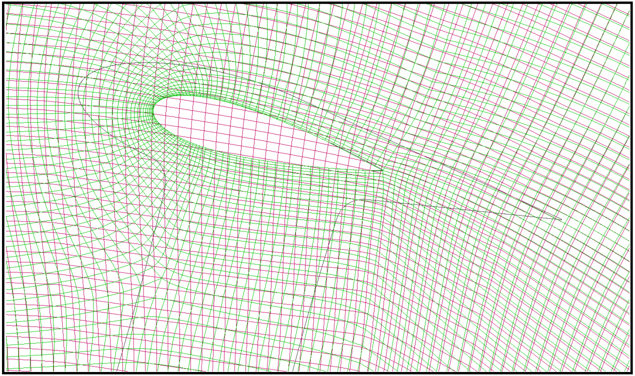
Detail: Blade





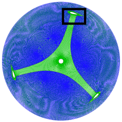


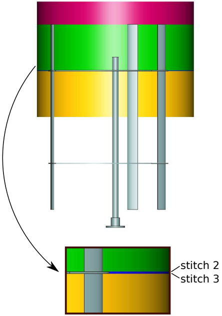


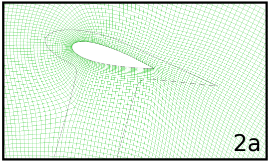


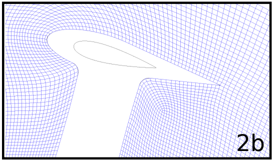
Stitch 2: blade-link plate

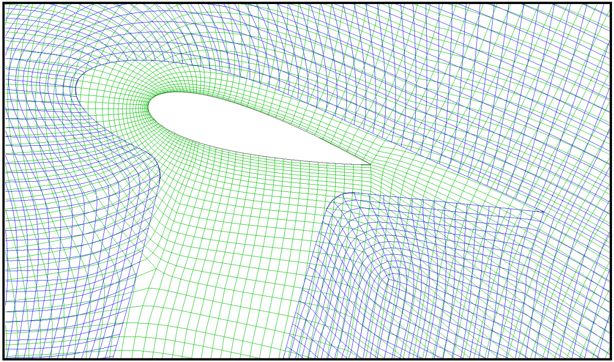
Stitch 2
Detail: Blade



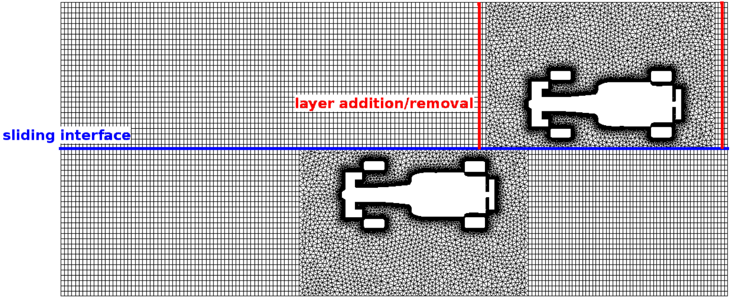








Dynamic mesh for external aerodynamics



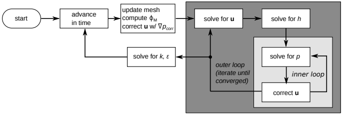
Simulation of overtake maneuver

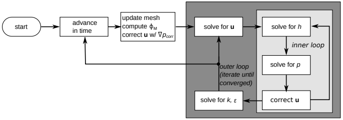
- `layerAdditionRemoval` on car front and back;
- fixed cells around car;
- `slidingInterface` on mesh middle section

The `topoDyMFoam` family

Improved algorithm for the compressible dynamic solver:

- ✓ strict coupling between energy with pressure equation

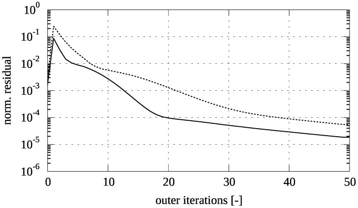




- ✓ enhanced flux correction after topological change (or remapping):

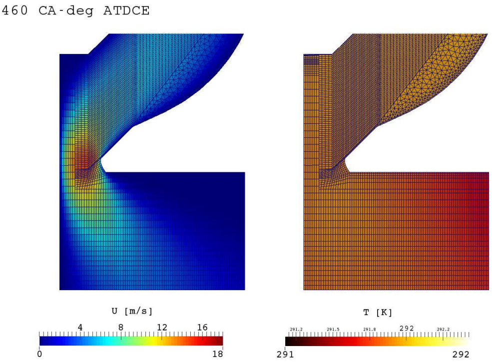
$$\nabla^2 p_{\text{corr}} + \nabla \cdot \left[\rho\left(\mathbf{x}^{n+1}, t^n\right)\mathbf{u}\left(\mathbf{x}^{n+1}, t^n\right)\right] dt = 0$$

- ✓ reference levels dynamically changed for ea ess of flux correction;
- ✓ **significantly faster convergence**



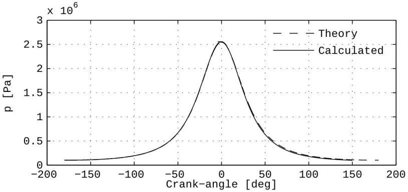
Compressible solver: conservation of variables

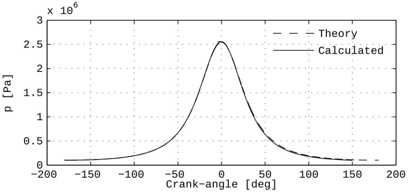
Conservation of momentum and energy through the non-conformal interface with layerAR (fields are NOT interpolated during post processing).



Whatever software you are using, please always check if energy (i.e. T) is conserved over the faces where topological changes (layerAR, slidingInterface, attachDetach) occur!

Compressible solver: validation

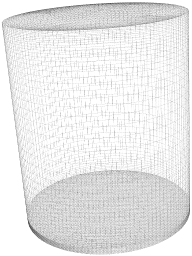




Adiabatic compression/expansion of air by a piston in a constant vessel: comparison between simulations and theory

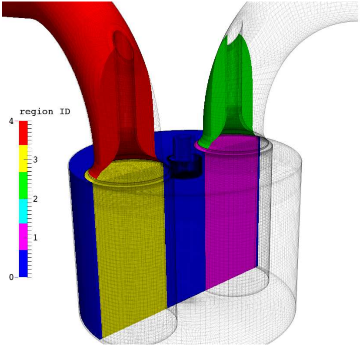
- piston speed: 2000 RPM
- vol. compr. ratio r=10
- initial cond: p=101325 Pa, T=292 K
- theory: $p\ V^k = const$

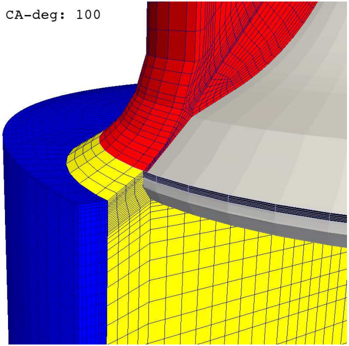
▸ **Mass conserved within 10^{-5} of relative error**

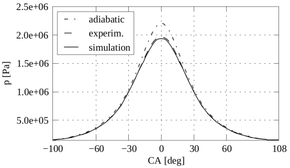


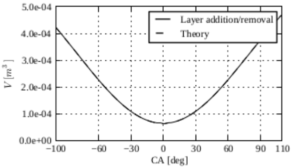
Adiabatic cylinder geometry

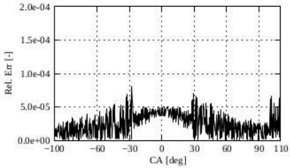
TCC: mesh motion





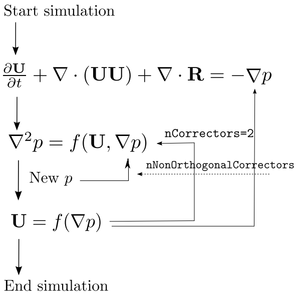






Effect of under-relaxation factors on the segregated solver

From what we have seen, EVERY equation is solved by a linear system AX=B, where the size of vectors and matrices is proportional to the mesh size.

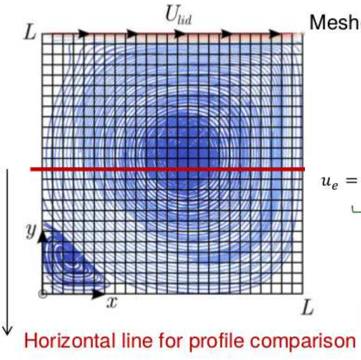


A common way in CFD to solve governing equations (mass+momentum) is by the PISO (Pressure implicit with splitting of operator) algorithm:

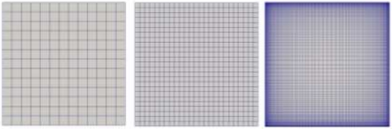
- segregated solver
- momentum and mass solved sequentially
- pressure-velocity coupling is achieved by iterations

Flux interpolation: influence on the solution

Lid-driven cavity case (Re=500



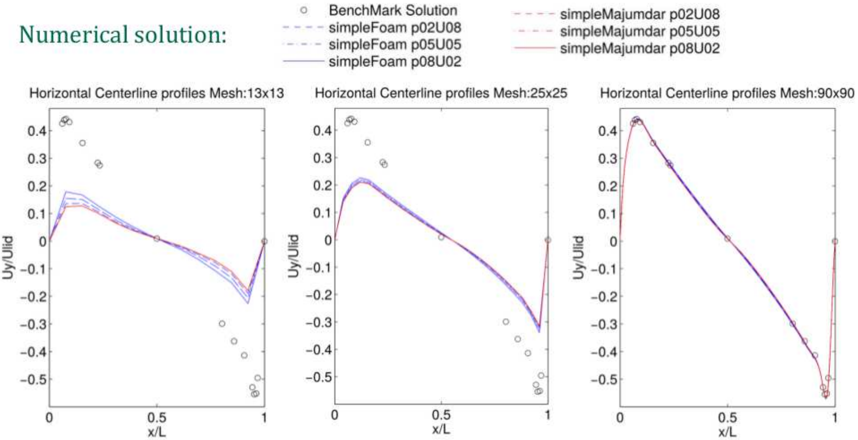
Meshes:
- 13 x 13
- 25 x 25
- 90 x 90



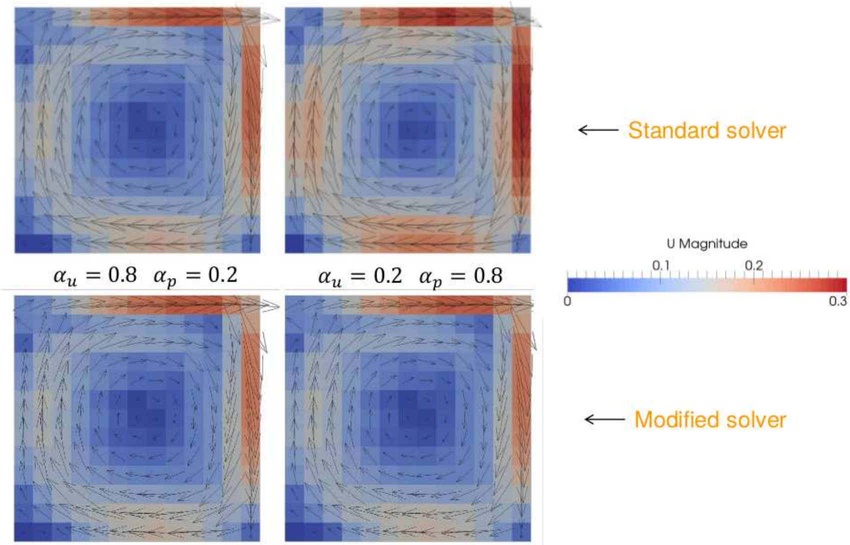
$$u_e = \underbrace{f_e^+ u_E + (1 - f_e^+) u_P}_{\text{linear interpolation}} \underbrace{- \Delta y \frac{P_E - P_P}{(A_P)_e} + f_e^+ \Delta y \frac{(P_e - P_w)_E}{(A_P)_E} + (1 - f_e^+)\, \Delta y \frac{(P_e - P_w)_P}{(A_P)_P}}_{\text{correction}}$$

- steady-state solver
- Perfect orthogonal cartesian mesh
- 2^{nd} order linear schemes for temporal discretization and diffusion terms
- different flux interpolation (2^{nd} order) for **convection**

Numerical solution:



Lid-driven cavity case (Re=5000)



← Standard solver

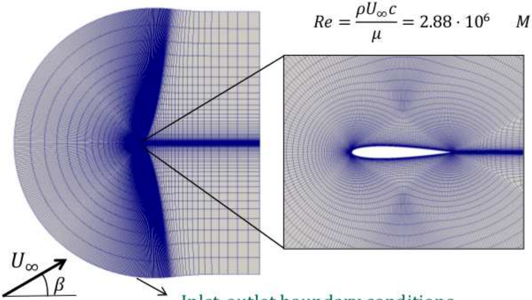
← Modified solver

2D computations of airfoil NACA 0012

Case characteristics

- 14500 cells
- $y^+_{max} < 1$
- LUST blended scheme
- kOmegaSST model
- Several angles of attack

$$Re = \frac{\rho U_\infty c}{\mu} = 2.88 \cdot 10^6 \qquad M = 0.15$$



Inlet-outlet boundary conditions

Comparison of:

Lift Coefficient
Drag Coefficient
Pressure Coefficient in upper wall

Cases studied:

$$\alpha_u = [0.1\ \ 0.2\ \ 0.3\ \ 0.4\ \ 0.5\ \ 0.6\ \ 0.7\ \ 0.8]$$

$$\alpha_P = 1 - \alpha_u$$

(Solution is found again independent of α_P)

$$\beta = [0\ \ 6\ \ 12\ \ 15]\ deg$$

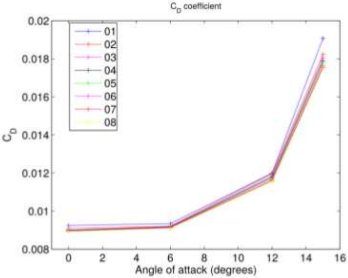
Convergence to residuals lower than $1 \cdot 10^{-6}$

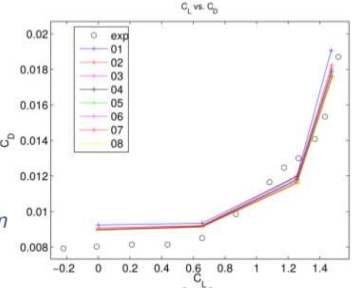
Experimental references:

Gregory, N. and O'Reilly, C. L., *Low-Speed Aerodynamic Characteristics of NACA 0012 Aerofoil Sections, including the Effects of Upper-Surface Roughness Simulation Hoar Frost*, NASA R&M 3726 (1970).

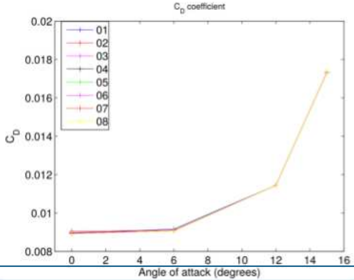
Ladson, C. L., Hill, A. S. and Johnson, Jr., W. G., *Pressure Distributions from High Reynolds Number Transonic Tests of an NACA 0012 Airfoil in the Langley 0.3 Meter Transonic Cryogenic Tunnel*, NASA TM 100526, December 1987.

NACA012 airfoil

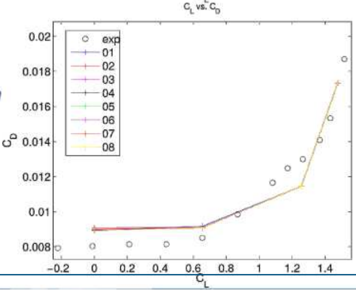


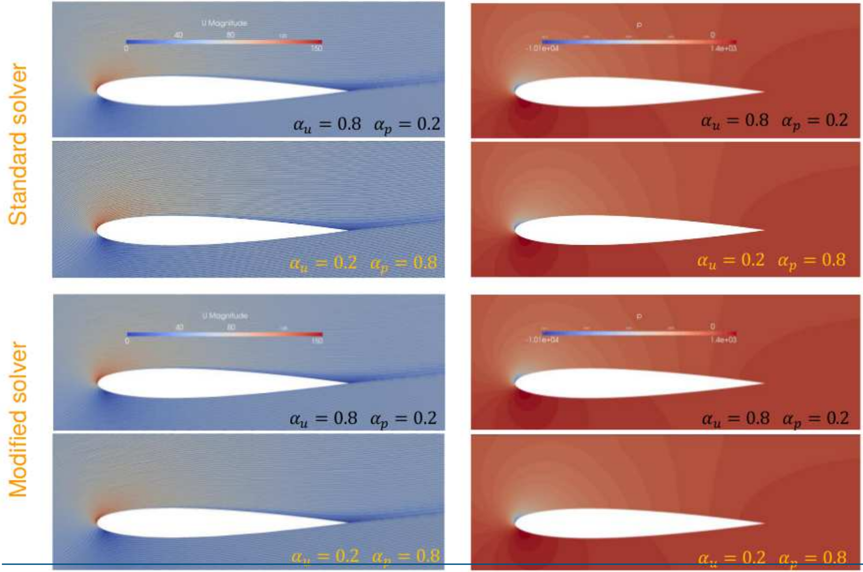


Standard
simpleFoam



Modified
simpleFoam




Standard solver
$\alpha_u = 0.8 \quad \alpha_p = 0.2$
$\alpha_u = 0.2 \quad \alpha_p = 0.8$
$\alpha_u = 0.8 \quad \alpha_p = 0.2$
$\alpha_u = 0.2 \quad \alpha_p = 0.8$
Modified solver
$\alpha_u = 0.8 \quad \alpha_p = 0.2$
$\alpha_u = 0.2 \quad \alpha_p = 0.8$
$\alpha_u = 0.8 \quad \alpha_p = 0.2$
$\alpha_u = 0.2 \quad \alpha_p = 0.8$

Hybrid RANS/LES models

- Model switches between URANS and LES on the basis of a criterion:
  - . . . whether cell lies on a wall (DES)
  - . . . whether distance from wall is below a threshold (hybrid RANS-LES)
  - . . . **local mesh resolution (filtered model)**
- Fine mesh resolution is used only in regions really requiring it: strong unsteadiness, curvature, free shear
- almost-steady regions are solved by RANS (boundary layers, jet cores, channel flow, etc)

A. Montorfano *et al.*, Comparison of Direct and Large Eddy Simulations of the Turbulent Flow in a Valve/Piston assembly, **Flow, Turbulence and Combustion**, 2015. DOI 10.1007/s10494-015-9620-6

Dynamic Length Resolution Model (DLRM)

- Switch between modeling (RANS) and the resolving (LES) the turbulent length scales;

Local resolvable length scale (LES):

$$\Delta_f = \max(\Delta_{eq},\ \alpha|\mathbf{U}|\Delta t)$$

Local integral length scale (RANS):

$$L_t \sim k^{1/2}/\omega$$

- **Locally minimum resolvable scale:**

$$\ell_t = \min\{L_t, \Delta_f\}$$

- Resulting formulation of the eddy viscosity[1]

$$\mu_t = g^2 \rho \frac{k}{\omega}$$

$$g \equiv \left(\frac{\ell_t}{L_t}\right)^{2/3}$$

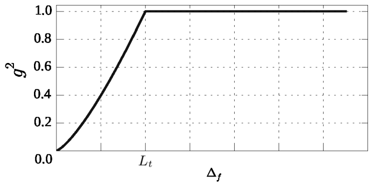
[1] F. Piscaglia, A. Montorfano and A. Onorati. "*A Scale Adaptive Filtering Technique for Turbulence Modeling of Unsteady Flows in IC Engines*". **SAE paper 2015-01-0395**

Dynamic Length Resolution Model (DLRM)

Resulting turbulent viscosity smoothly changes from RANS to LES depending on the local resolvable scales compared to problem's turbulence lengthscale

$$\mu_t = g^2 \rho \frac{k}{\omega}$$

$$g \equiv \left(\frac{\ell_t}{L_t}\right)^{2/3}$$



- $\ell_t = L_T \Rightarrow g^2 = 1$

$$\mu_t = \rho k/\omega$$ (RANS)

- $\ell_t = \Delta_f \Rightarrow g^2 < 1$

$$\mu_t < \mu_{t,\text{RANS}}$$ (LES)

- $\lim_{\Delta_f \to 0} g^2 = 0$

$$\mu_t \to 0$$ (DNS/ILES)

Minimum solvable scale

$$\Delta_f = \max(\Delta_{\mathrm{eq}},\ \alpha|\mathbf{U}|\Delta t)$$

limited by space discretization

- We define the LSR index: [1]

$$\mathrm{LSR} = \frac{\bar{\Delta}}{\ell_{di}}$$

- LES is (almost) complete if

$$\mathrm{LSR} \leq \mathrm{LSR}_{\mathrm{max}} \approx 5 \div 7$$

- Therefore:

$$\Delta_{\mathrm{eq}} = \mathrm{LSR}_{\mathrm{max}} \cdot \ell_{di}$$

limited by time discretization

- Distance covered by fluid particle in a timestep:

$$\Delta x = |\mathbf{U}|\Delta t$$

- ... in relation with cell size:

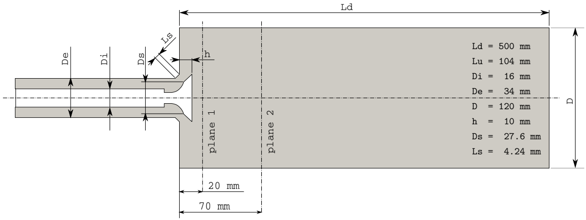
$$\mathrm{CFL} = \frac{|\mathbf{U}|\Delta t}{\Delta x}$$

- too a high CFL limits the turbulence timescale:

$$\alpha = \frac{\mathrm{CFL}_{\mathrm{max}}}{\mathrm{CFL}_{\mathrm{local}}}$$

[1] F. Piscaglia, A. Montorfano, A. Onorati, F. Brusiani. **Oil & Gas Science and Technology, IFPEN, Vol.69, 2014**

Validation: flow around poppet valve

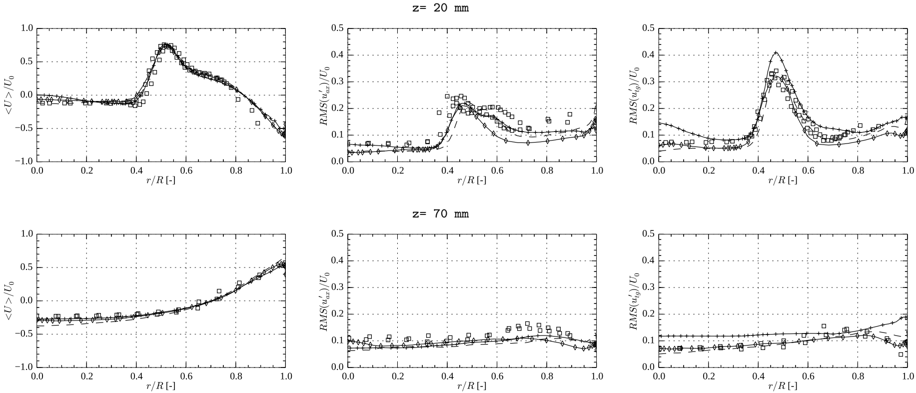


EXPERIMENTS:

- Simple IC engine geometry: one axis-centered valve, expansion ratio=3.5
- Mean velocity at the inlet: 65 m/s ($Ma \approx 0.1$)
- **LDA measurements @** z=**20 mm** and z=**70 mm**
  - axial mean flow velocity
  - velocity fluctuations (**radial** and **tangential** direction)

L. Thobois, G. Rymer, T. Soulères, and T. Poinsot. "Large-eddy simulation in IC engine geometries". SAE Technical Paper 2004-01-1854, 2004.

Validation: flow around a poppet valve

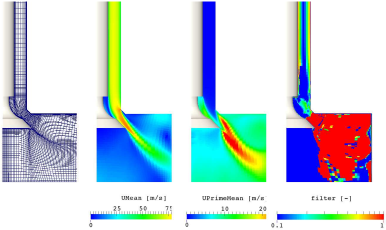


—□— experiments; — ◇ — DLRM (700 k cells); — + — DLRM (1.5 M cells); – – WALE (12 M cells).

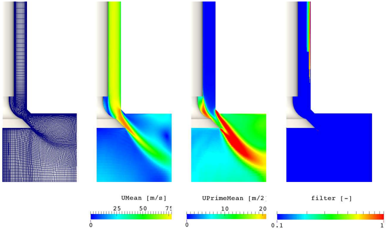
F. Piscaglia, A. Montorfano and A. Onorati. “*A Scale Adaptive Filtering Technique for Turbulence Modeling of Unsteady Flows in IC Engines*”. **SAE paper 2015-01-0395**

DLRM: resolved scales

COARSE MESH (700 k cells)



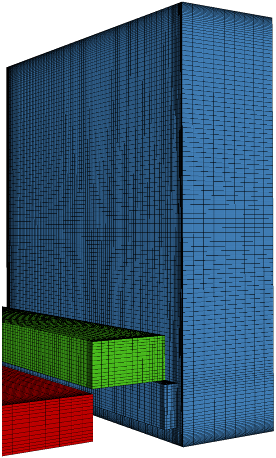
FINE MESH (5.3 M cells)

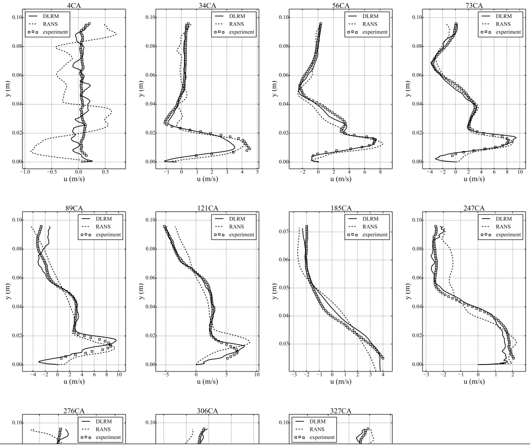


F. Piscaglia, A. Montorfano and A. Onorati, “*A Scale Adaptive Filtering Technique for Turbulence Modeling of Unsteady Flows in IC Engines*”, **SAE paper 2015-01-0395**

Square piston with guillotine valve

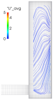
DLRM turbulence model ($\rightarrow$ SAE 2015-01-0395) of topologically changing grids

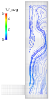


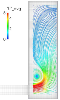


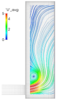
Geometry and measurements were kindly provided by **Prof. Jacques Borée** (*ISAE-ENSMA*, Futuroscope Chasseneuil, France)

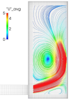
Square piston with guillotine valve

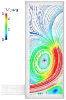

Experiment: 4 (CA)

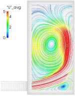

DLRM: 4 (CA)

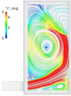

Experiment: 34 (CA)

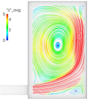

DLRM: 34 (CA)

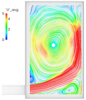

Experiment: 56 (CA)

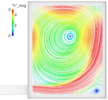

DLRM: 56 (CA)

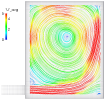

Experiment: 73 (CA)

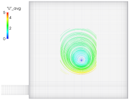

DLRM: 73 (CA)

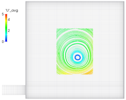

Experiment: 89 (CA)

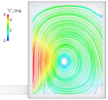

DLRM: 89 (CA)

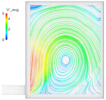

Experiment: 121 (CA)

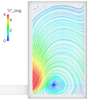

DLRM: 121 (CA)

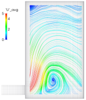

Experiment: 185 (CA)


DLRM: 185 (CA)

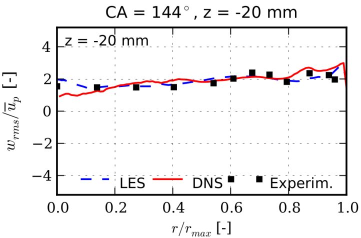

Experiment: 247 (CA)

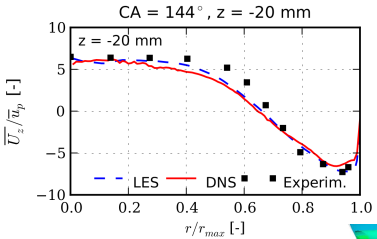

DLRM: 247 (CA)

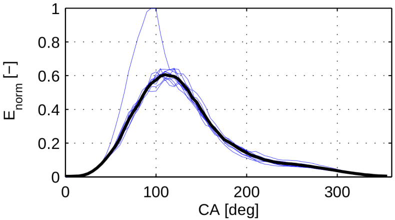

Experiment: 276 (CA)

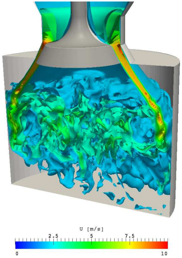

DLRM: 276 (CA)

LES of engine-like geometries in OpenFOAM



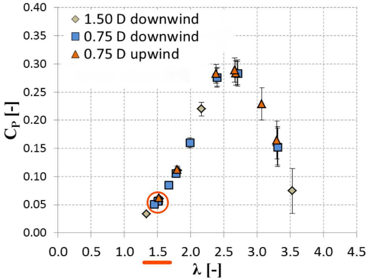


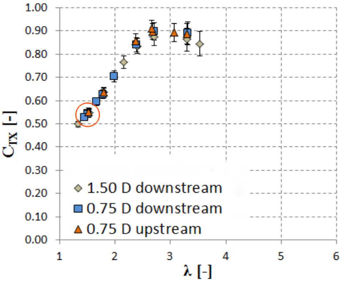




A. Montorfano, F. Piscaglia, M. Schmitt, C.E. Frouzakis, A. G. Tomboulides, K. Boulouchos and A. Onorati. "Comparison of direct and large eddy simulations of the turbulent flow in a valve/piston assembly", Flow, Turbulence and Combustion, Springer. In Press, 2015.

Vertical Axis Wind Turbines (VAWT)

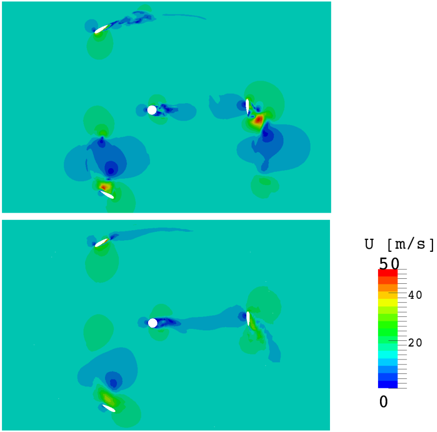




- Wind upstream velocity $V_0 = 14.2$ m/s
- Wind rotational speed $\Omega = 41.44$ m/s
- Tip speed ratio $\lambda = \Omega R/U = 1.5$

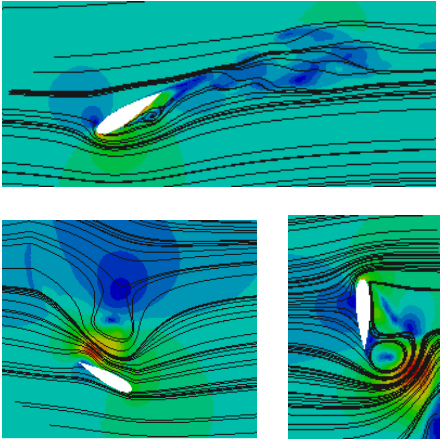
DLRM

$k - \omega$ SST

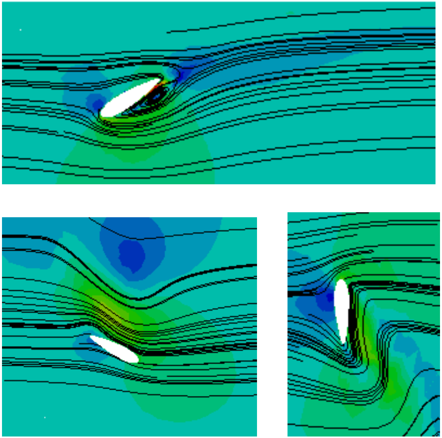


Vertical Axis Wind Turbines (VAWT)

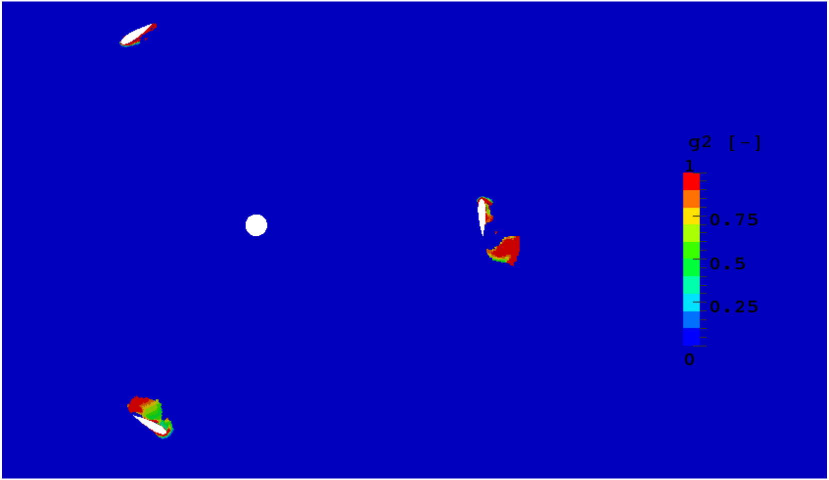
DLRM



$k-\omega$ SST

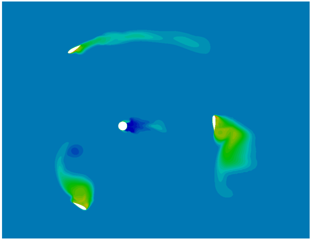


Vertical Axis Wind Turbines (VAWT)

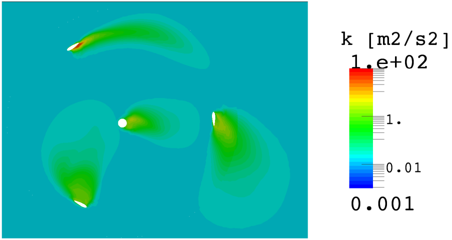


Turbulent Kinetic Energy (TKE)

DLRM

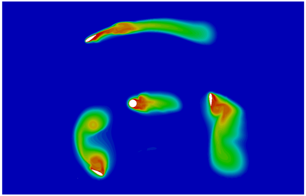


$k - \omega$ SST

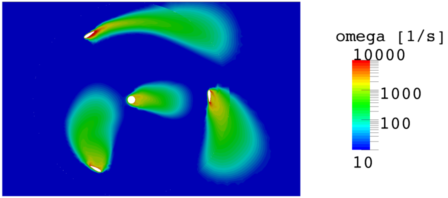


Vertical Axis Wind Turbines

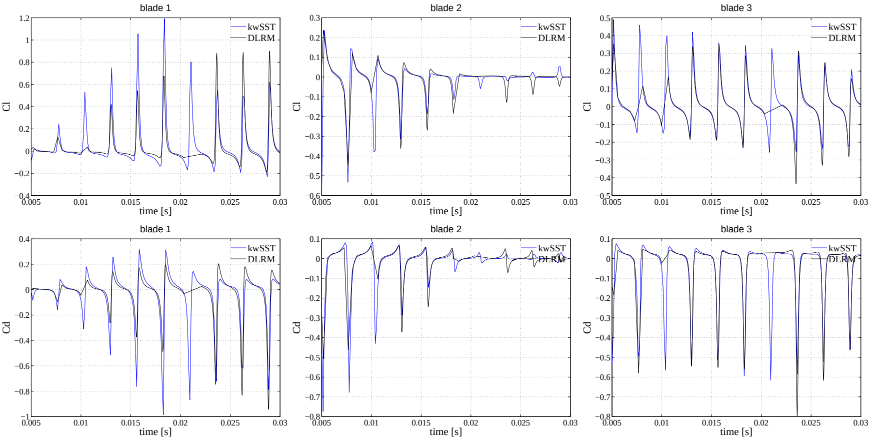
DLRM



$k - \omega$ SST

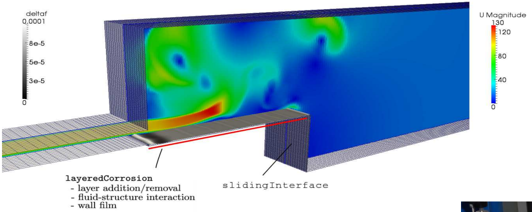


VAWT: aerodynamic coeffs

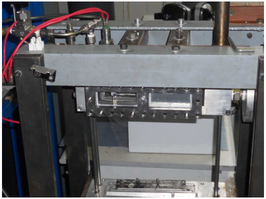


chromoFoam (*Combustion of Hybrid ROcket MOtors*)

`PIMPLE (merged PISO+SIMPLE) solver with dynamic mesh and layering for solid propellant and turbulent diffusion flames with reacting Lagrangian parcels, surface film and pyrolysis modelin`



- **Dynamic mesh modeling** of the propellant corrosion driven by surface regression rate
- **LES turbulence modeling** of the fluid flow
- **Surface-film modeling** (propellant gasification, droplet entrainment)
- **Heat-transfer modeling** at the propellant's interfaces (fluid-liquid, liquid-solid)



Joint project with the *SPLab (Space Propulsion Laboratory)*, Aerospace Science and Technology Department, Politecnico di Milano (**Prof. Luciano Galfetti**)

Some further (and trivial) notes...

The initial mesh (TDC, closed valves) can be done:

- by open-source code (example: `snappyHexMesh`+bash scripting)
- any commercial mesh generators supporting the OpenFOAM® format: you can find an example at:

  `http://www.pointwise.com/theconnector/July-2014/Unsteady-Engine-Analysis.shtml`

After the initial mesh is generated:

- **automatic case setup** (case pre-processing, including constrained domain decomposition, solver settings, etc) is automatically defined by `bash` scripting.
- **post-processing** (averaging for LES post-processing, countour plots, etc) is performed off-line by implemented applications/bash scripting:
  - `python` scripting
  - `VTK` scripting by ParaView®

Conclusions and Future Work

IN THIS PRESENTATION:

- ✓ Development and validation of mesh motion with topological changes (*implementation compatible with software releases by OpenFOAM® Foundation*)
  - enhanced algorithm for non-conformal interfaces coupling to perform mesh stitching over complex interfaces (including sharp corners)
  - variable topology-driven time-stepping for dynamic layer addition/removal
  - full low-level integration to favor extensions of the motion solver
- ✓ combination of `AMI/slidingInterface` for mesh motion strategy[1]
- ✓ novel compressible dynamic solver for consistent handling of flux interpolation through non conformal interfaces (target-mesh and supermesh strategy)
- ✓ novel hybrid RANS/LES turbulence model (**DLRM**)
- ✓ validation

NOTE: the presentation includes all the development performed by the authors from June 2014 up to now.

Computational resources

Simulations were carried out by the computing resources provided on BLUES, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory (IL, USA).

- Compute – 310 nodes, Each with two Sandy Bridge 2.6 GHz Pentium Xeon (hyper threading disabled.) 4960 available compute cores
- Memory – 64GB of memory
- Storage – 110TB of clusterwide space provided by GPFS (shared with Fusion) 15GB on node Ramdisk
- Network – Infiniband Qlogic QDR

Argonne NATIONAL LABORATORY

Acknowledgments

Authors would like to thank:

- the **LCRC** (Laboratory Computing Resource Center), ARGONNE National Lab, for making available the computing resources thorugh the HPC cluster blues within the PETSc-Foam project.
- **Dr. S. M. Aithal** (*Argonne National Laboratory*, Lemont, IL 60439, United States) for helping in scaling and parallelization tests.
- **Proff. V. Sick**, **D. Reuss**, **X. Yang** and **T. Kuo**, who made available detailed data and geometry of the TCC engine; the TCC engine work has been funded by General Motors through the General Motors University of Michigan Automotive Cooperative Research Laboratory, Engine Systems Division.
- **Prof. Jacques Borée** (*ISAE-ENSMA*, Futuroscope Chasseneuil, France) for the very detailed dataset of experimental data provided on the square piston geometry.
- Mr. **Y. Wu**, Mr. J. Martinez (phD student, ICE Group, Politecnico di Milano) and Mr. **Mr. F. Giussani** (MS student, ICE Group, Politecnico di Milano).

Thank you for your attention!

Prof. **Federico Piscaglia**, Ph.D.
Associate Professor of Internal Combustion Engines

CONTACT INFORMATION

Address	Dipartimento di Energia, Politecnico di Milano via Lambruschini 4, 20156 Milano (ITALY)
E-Mail:	federico.piscaglia@polimi.it
Phone:	(+39) 02 2399 8620

References I

F. Piscaglia, A. Montorfano, and A. Onorati.
A Scale Adaptive Filtering Technique for Turbulence Modeling of Unsteady Flows in IC Engines.
SAE Int. J. Engines, Paper n. 2015-01-0395, 2015.

A. Montorfano, F. Piscaglia, and A Onorati.
An Extension of the Dynamic Mesh Handling with Topological Changes for LES of ICE in OpenFOAM.
SAE paper 2015-01-0384, 2015.
SAE World Congress & Exhibition, Detroit, Michigan (USA).

F. Piscaglia, A. Montorfano, and A. Onorati.
Adaptive LES of dynamically changing geometries in OpenFOAM®: an application to the TCC test case.
In *LES4ICE - LES for Internal Combustion Engine Flows@IFPEN, Rueil-Malmaison, 4-5 December*, 2014.

A. Montorfano, F. Piscaglia, M. Schmitt, Y.M. Wright, C.E. Frouzakis, A.G. Tomboulides, K. Boulouchos, and A. Onorati.
Comparison of Direct and Large Eddy Simulations of the Turbulent Flow in a Valve/Piston Assembly.
To appear in Flow, Turbulence and Combustion, 2015.

J. Martínez, F. Piscaglia, A. Montorfano, A. Onorati, and S.M. Aithal.
Influence of spatial discretization schemes on accuracy of explicit LES: Canonical problems to engine-like geometries.
Computers & Fluids, 117(0):62 – 78, 2015.

F. Piscaglia, A. Montorfano, and A. Onorati.
Towards the LES Simulation of IC Engines with Parallel Topologically Changing Meshes.
SAE Int. J. Engines,, 6(2):926–940, 2013.

F. Piscaglia, A. Montorfano, A. Onorati, and F. Brusiani.
Boundary Conditions and SGS Models for LES of Wall-Bounded Separated Flows: An Application to Engine-Like Geometries.
Oil Gas Sci. Technol. - Rev. IFP Energies nouvelles, 69(1):11–27, 2014.

References II

F. Piscaglia, A. Montorfano, A. Onorati, and J. P. Keskinen.
Boundary conditions and subgrid scale models for LES simulation of Internal Combustion Engines.
In International Multidimensional Engine Modeling User's Group Meeting 2012, Detroit, 2012.

F. Piscaglia, A. Montorfano, and A. Onorati.
Development of a Non-Reflecting Boundary Condition for Multidimensional Nonlinear Duct Acoustic Computation.
Journal of Sound and Vibration, 332(4):922–935, 2013.

F. Piscaglia, A. Montorfano, A. Onorati, and G. Ferrari.
Modeling of Pressure Wave Reflection from Open-Ends in ICE Duct Systems.
SAE Technical Paper n. 2010-01-1051, 2010.

F. Piscaglia and G. Ferrari.
Development of an Offline Simulation Tool to Test the On-Board Diagnostic Software for Diesel Aftertreatment Systems.
SAE paper n. 2007-01-0133, 2007.
doi:10.4271/2007-01-1133.

F. Piscaglia and G. Ferrari.
Modeling of the Unsteady Reacting Flows in the Diesel Exhaust Aftertreatment Systems.
ECOS 2007 Conference: Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy . Padova, Italy, 2007.

F. Piscaglia, C. J. Rutland, and D. E. Foster.
Development of a CFD Model To Study The Hydrodynamic Characteristics And The Soot Deposition Mechanism On The Porous Wall Of A Diesel Particulate Filter.
SAE paper n. 2005-01-0963, SAE 2005 Int. Congress & Exp. (Detroit, Michigan), April 11-14, 2005.