OpenFoam UserConference 2014 Berlin

October 17, 2014

www.esi-group.com

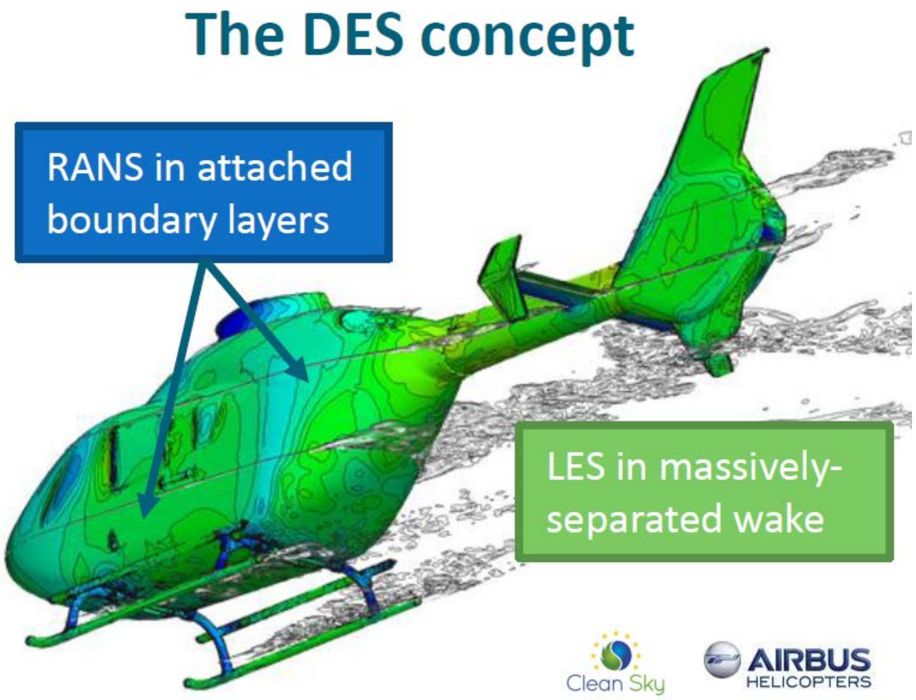
目次

1. DDES enhancements
2. SnappyHexMesh Developments
3. FoamyHexMesh
4. Parallel operation
5. Boundedness and MULES

DDES Enhancements

After RANS, Detached-Eddy Simulation (DES) is becoming the new standard in industry:

- Turbulence modelling is the principal accuracy bottleneck in CFD
- DES addresses this by *resolving more* and *modelling less* of the turbulent motion
- Enabled by increases in computing power



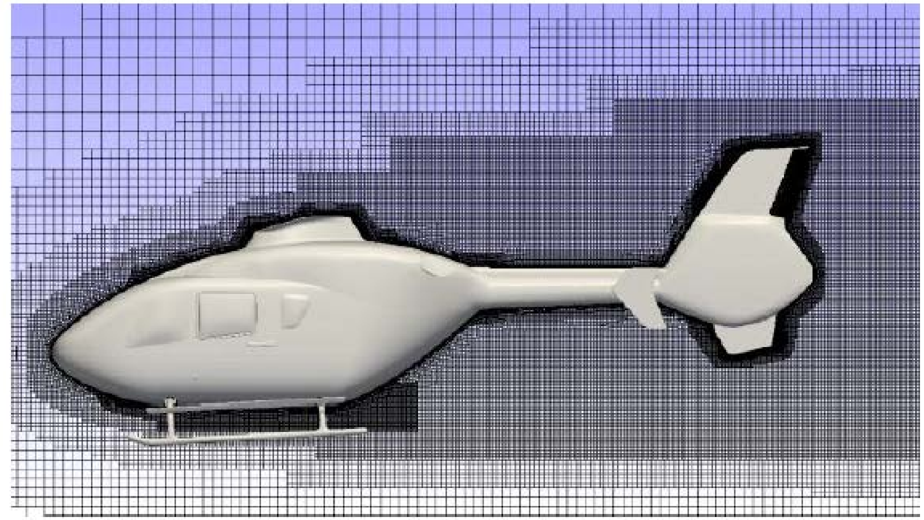
DDES Enhancements

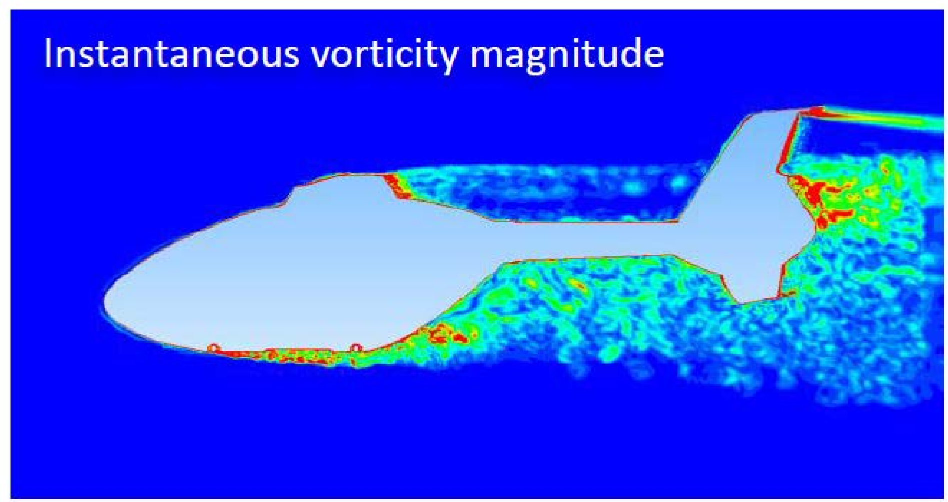
EC135 helicopter fuselage

- EU-funded CleanSky project ‘Helides’
- RANS, URANS and DES compared for complex helicopter fuselage at flight Reynolds number
- Configurations with and without landing skids, three angles of attack
- Blind comparison with experimental measurements

Simulation setup:

- Grids of 32M (with skids) generated using NUMECA HEXPRESS
- Spalding wall function, $y+\approx 40$
- DDES based on Menter SST model
- Hybrid convection scheme

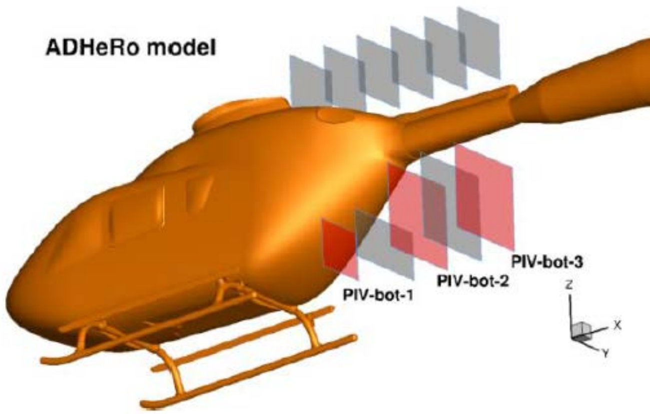


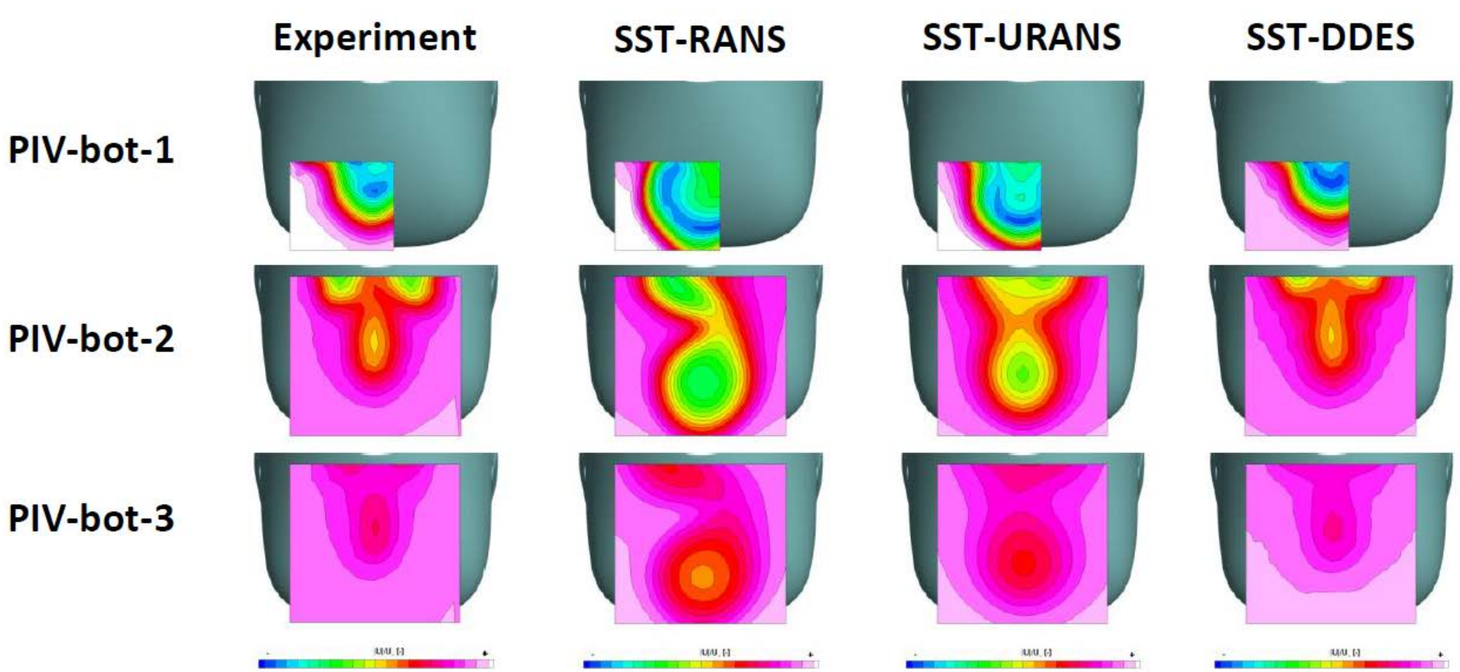


DDES Enhancements

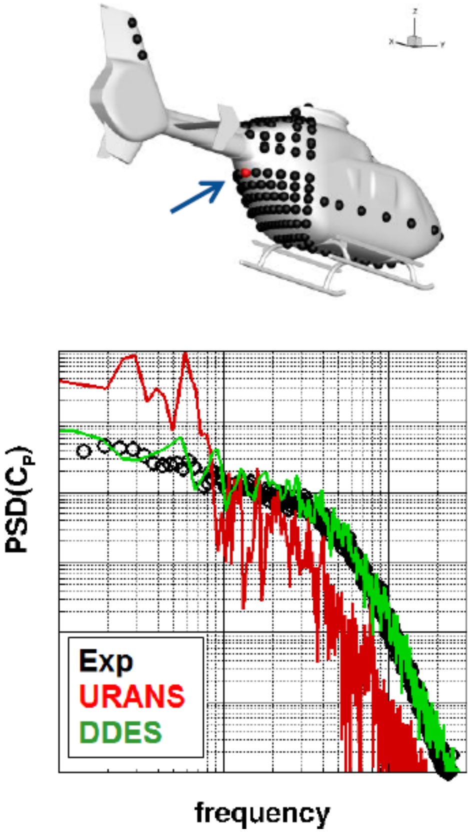
Prediction of wake topology

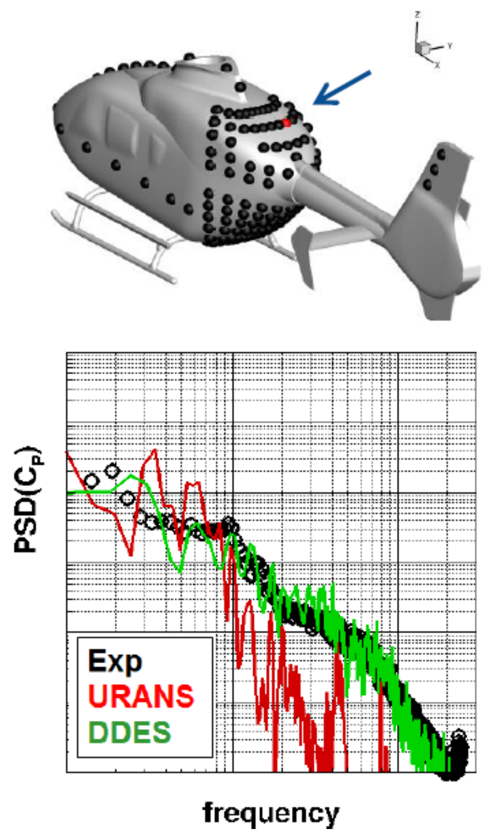
- Strong improvement of DES relative to RANS and URANS
- Improved prediction of surface pressure in wake region

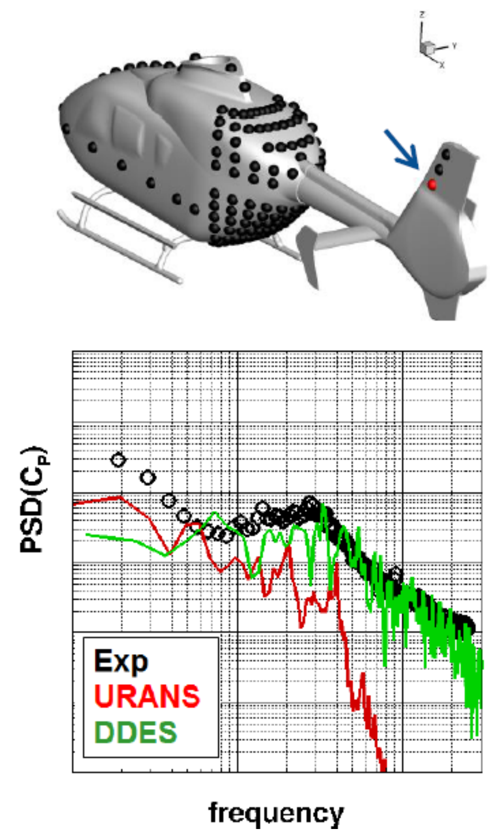




Prediction of wake topology

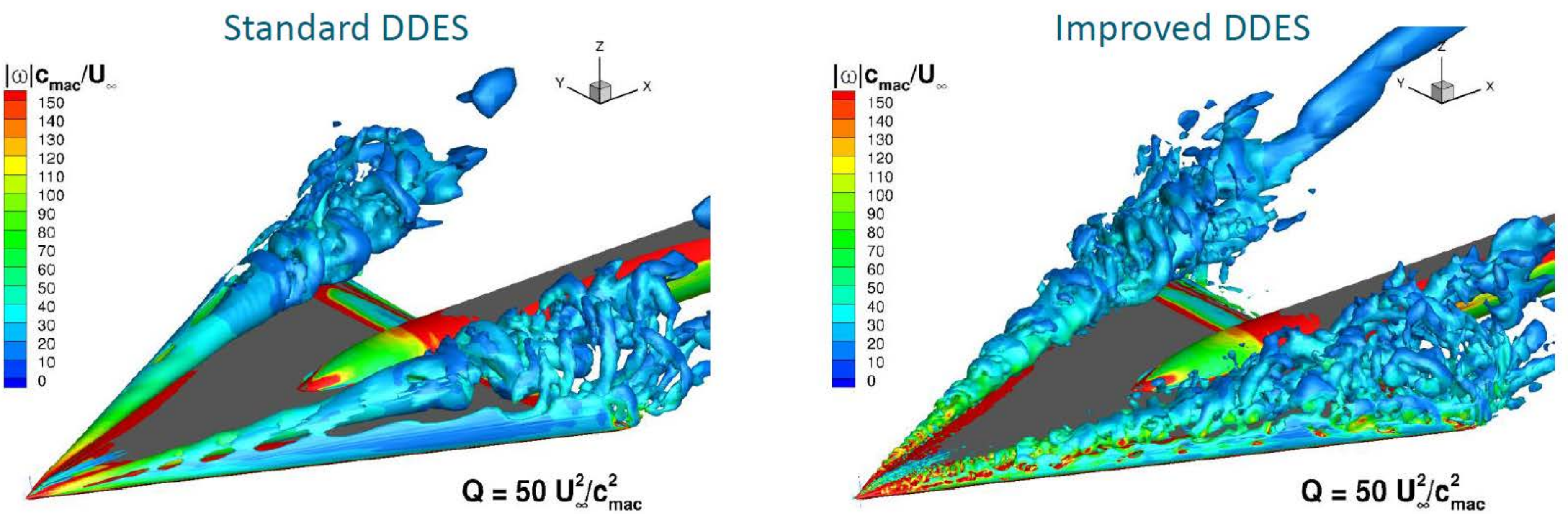






DDES Enhancements

In the next OpenFOAM major release, enhanced DDES methods like SA-WALE-DDES and SA-σ-DDES will be available.

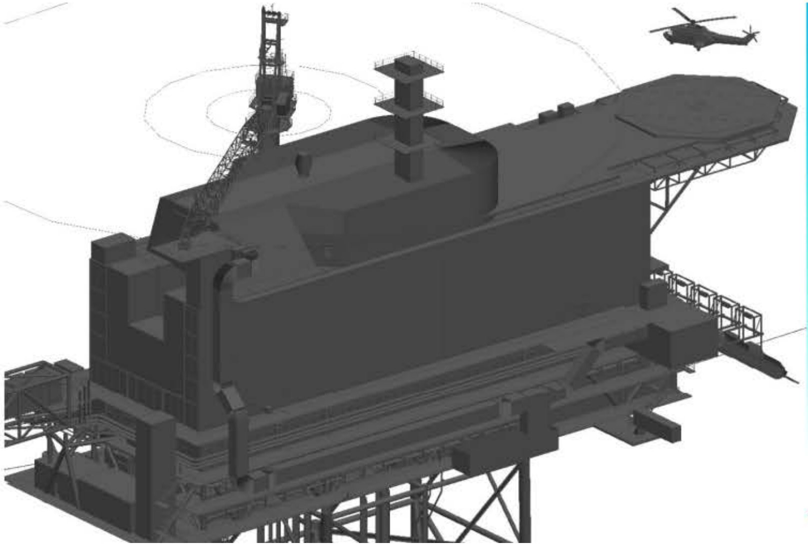


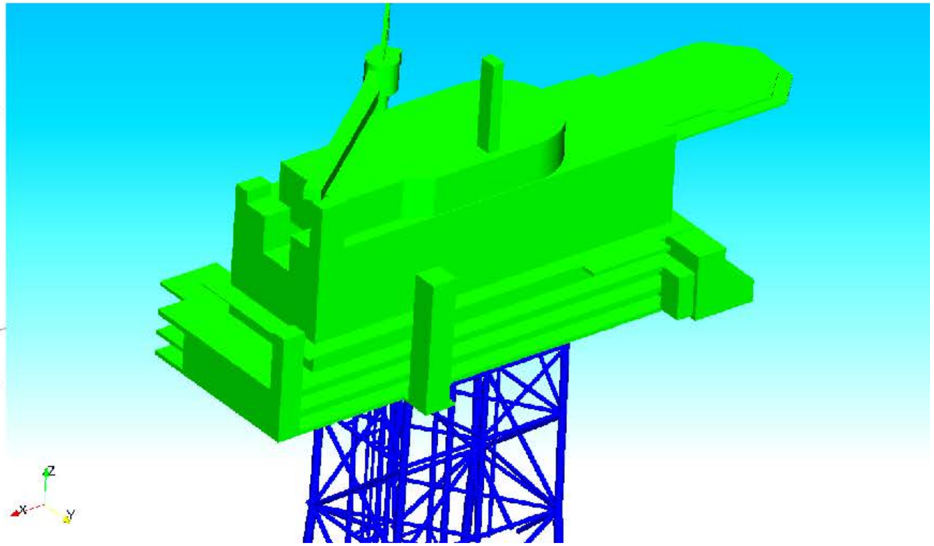
Improved DES methods appear promising:

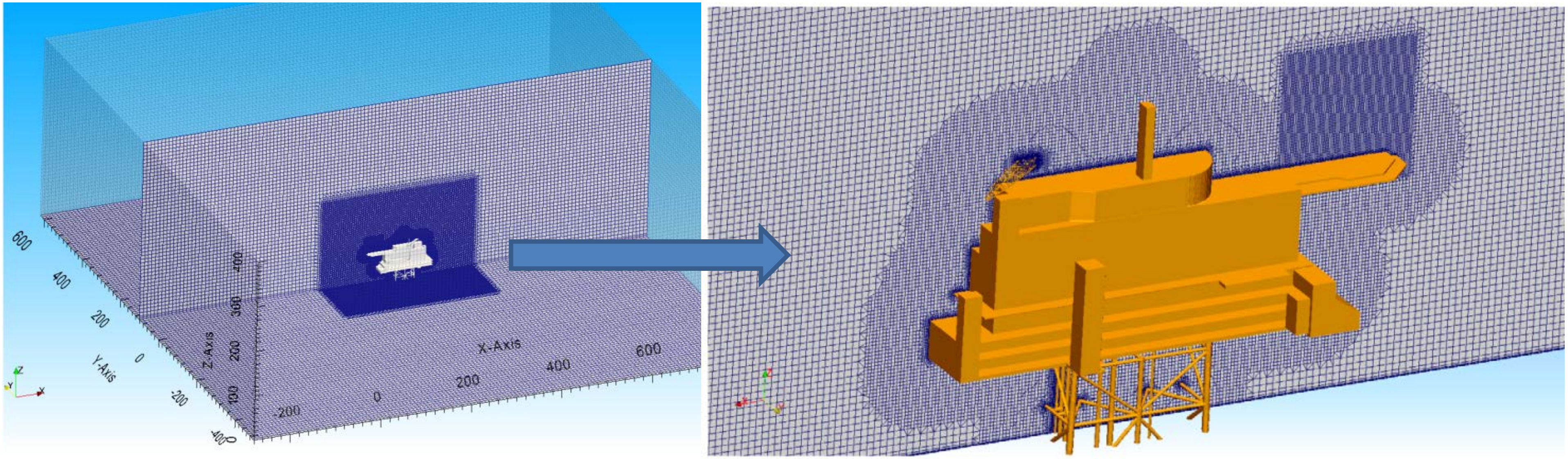
- Improved transition to turbulence in free and separated shear layers
- Retains practical and robust nature of approach

Increasing the level of complexity

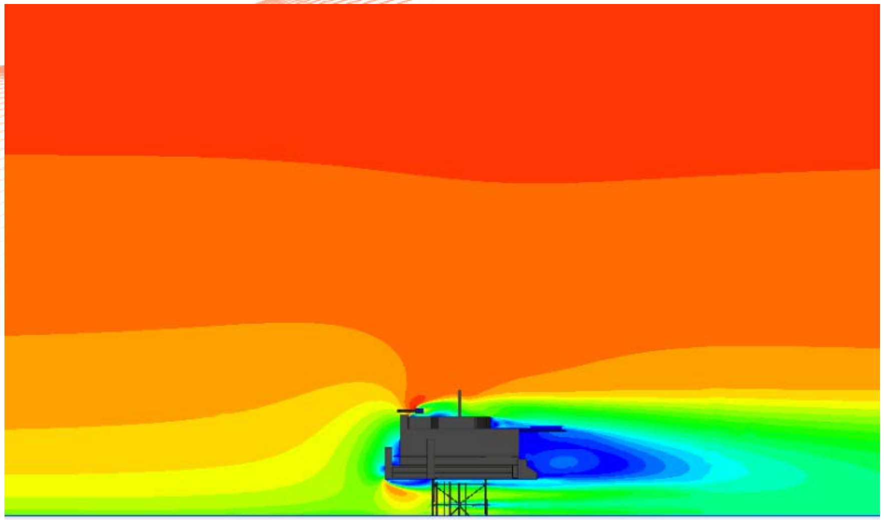
Increasing the level of complexity

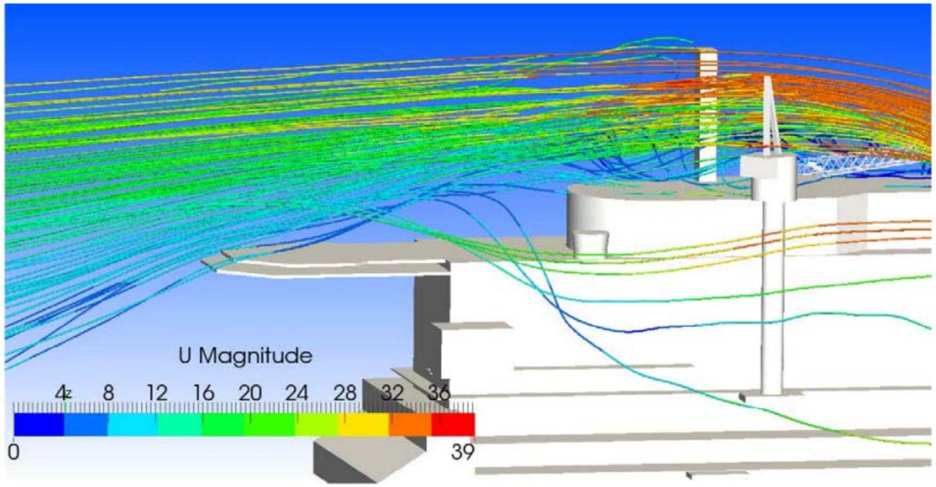


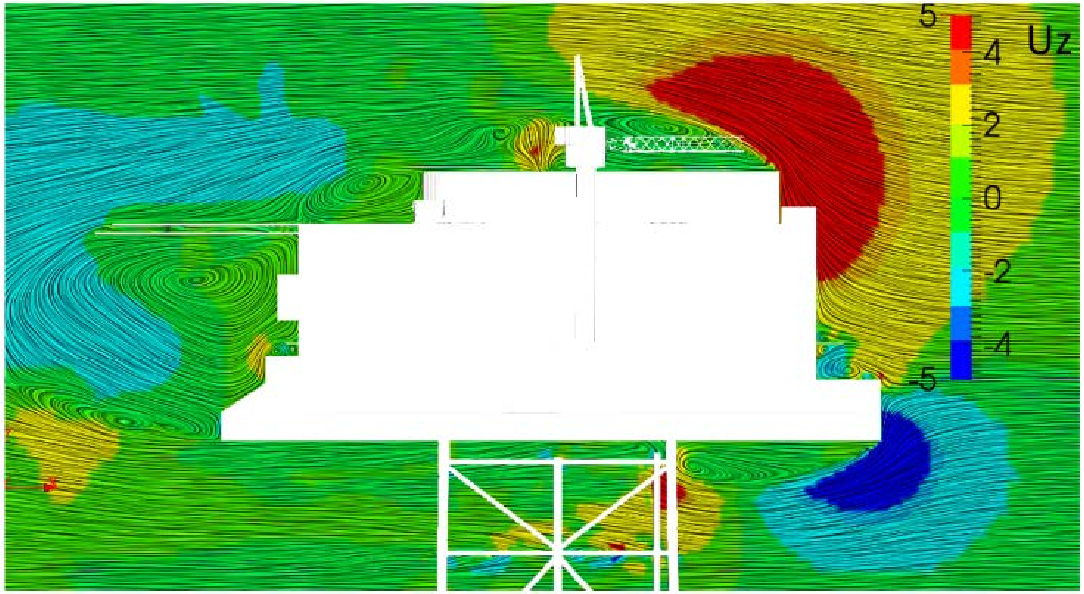


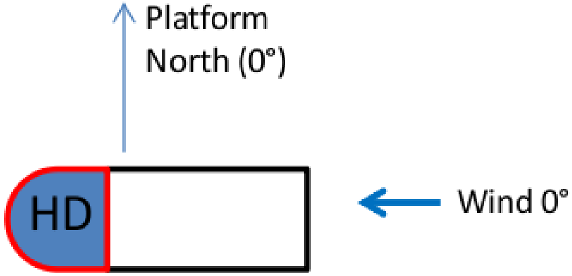


Increasing the level of complexity

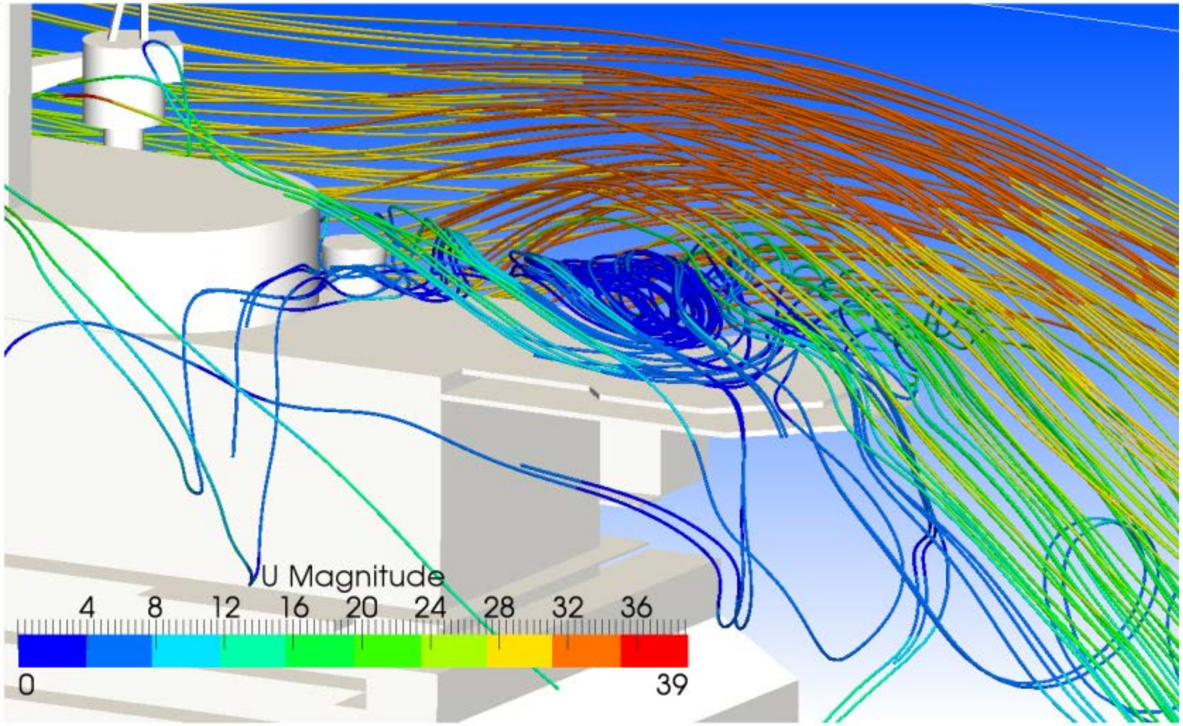


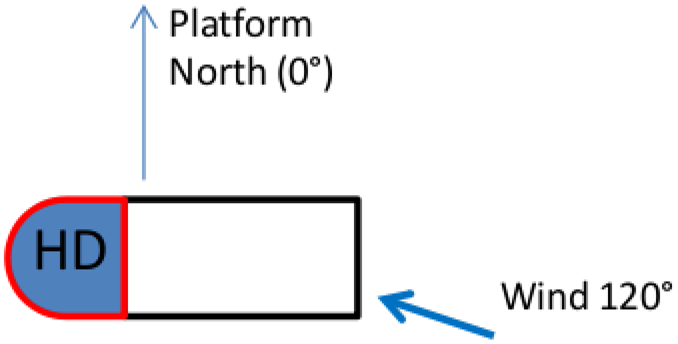






Increasing the level of complexity



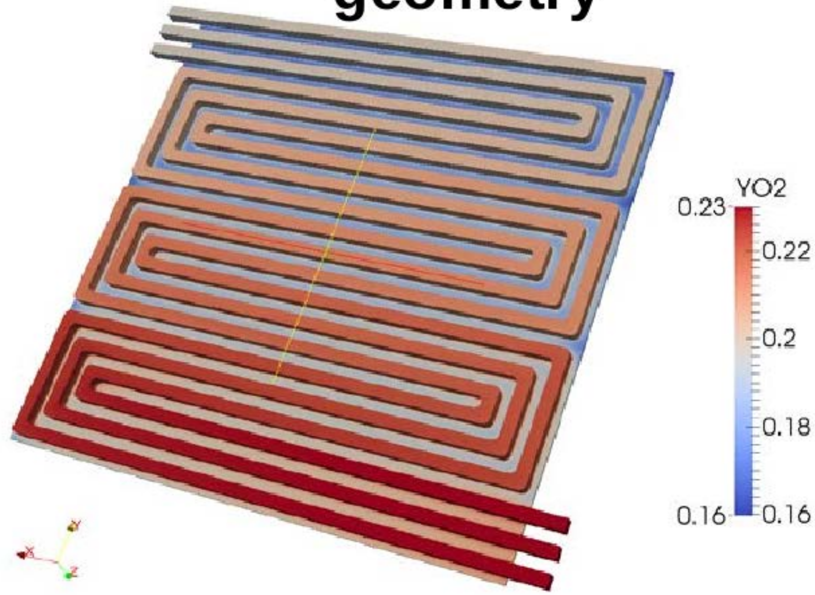


Several wind directions are automatically tested to investigate which are the most dangerous wind directions with strong downdraft on the helideck (90% of accidents on oil-rigs are related to helipcoter accidents during landing and take-off).

Green Energies related Simulations

Liquid metal batteries
for smart grid storage

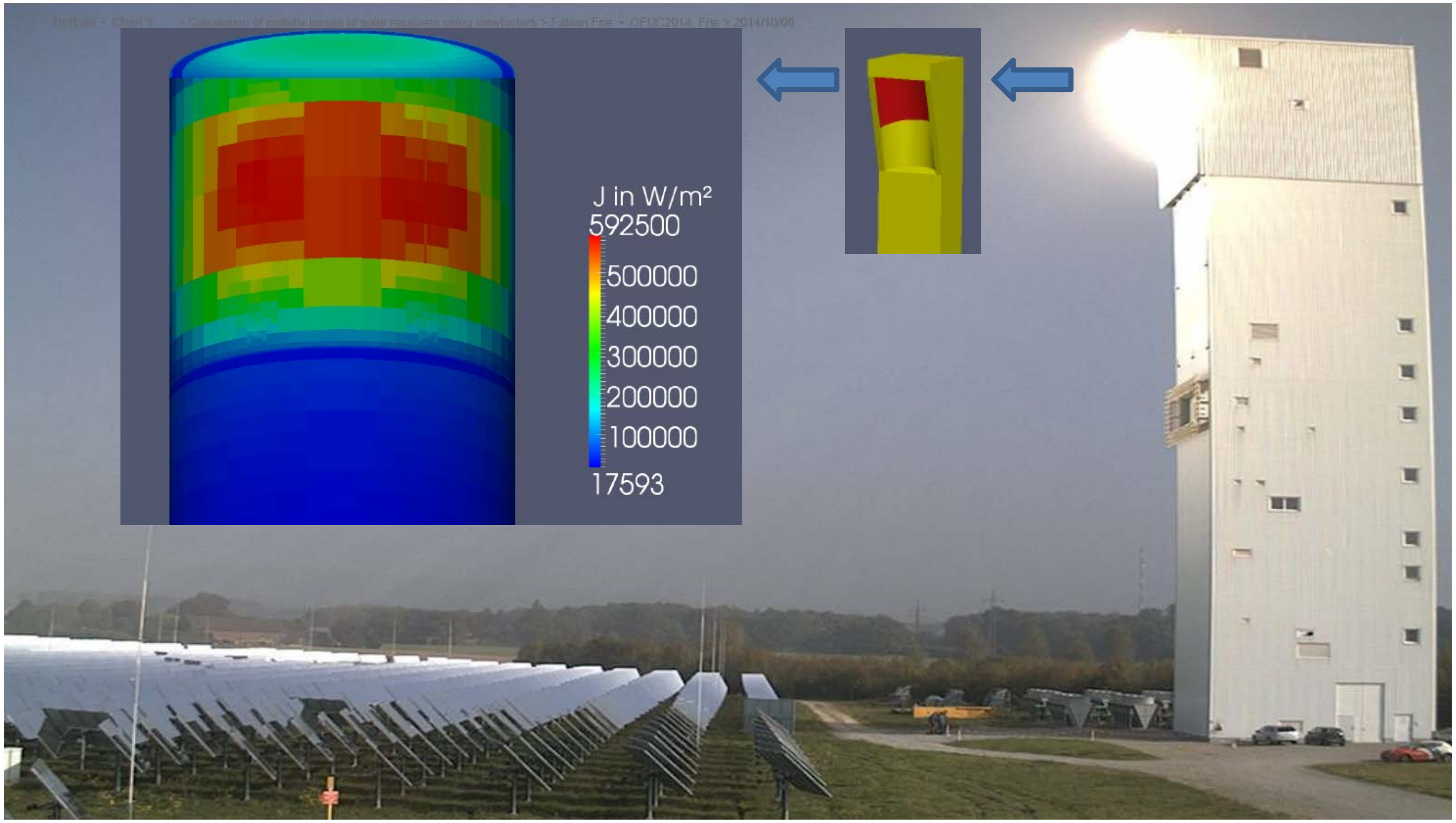
Salome-generated geometry



Fuel cells analysis

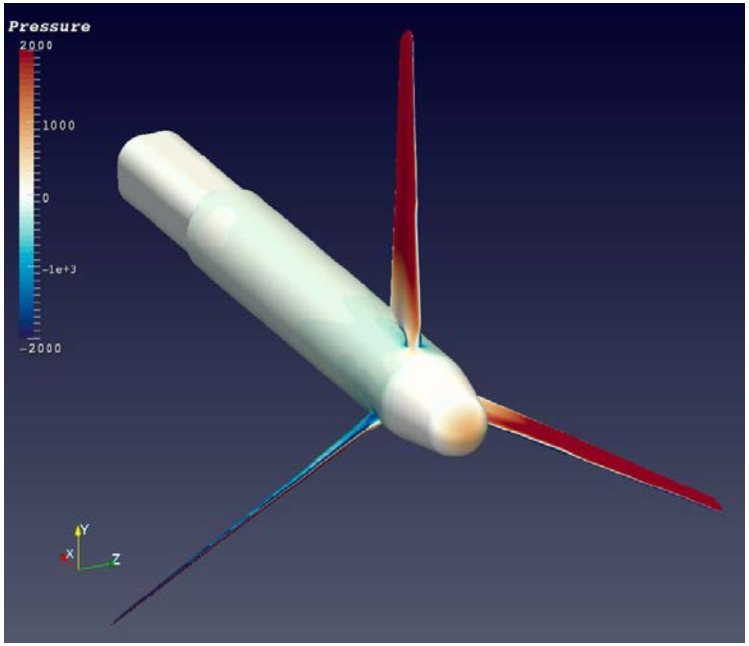
To contribute
and download → http://openfuelcell.sourceforge.net/
http://www.ieafuelcell.com/

Solar Power Tower



Solver : buoyantPimpleFoam, **Radiation**: Viewfactor, **Turbulence model** : k-ω-SST

Green Energies related Simulations



Measurements:
between June-July 2014
Place:
Large Scale Low Speed Facility (LLF) of the German Dutch Wind Tunnels (DNW)
Objective:
improve the quality of the database

MEXICO project (Model Experiments In Controlled Conditions)

SnappyHexMesh OF230 new features

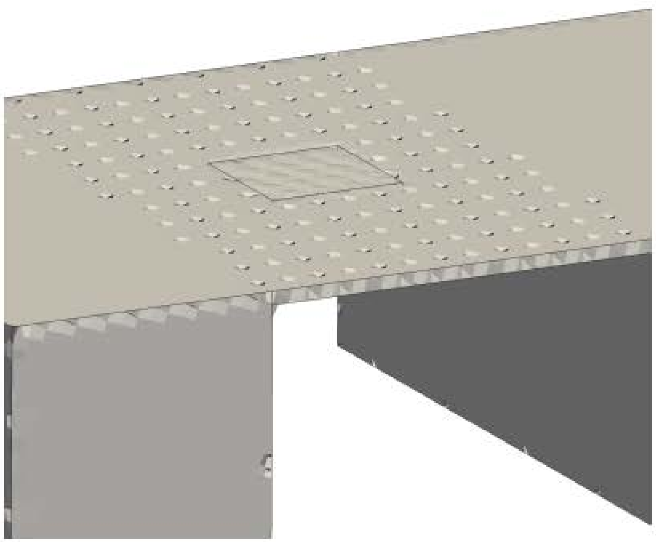
Automatic gap refinement:

- Small gaps will form blockages
- Difficult to refine manually
- Instead: detect and increase surface refinement level

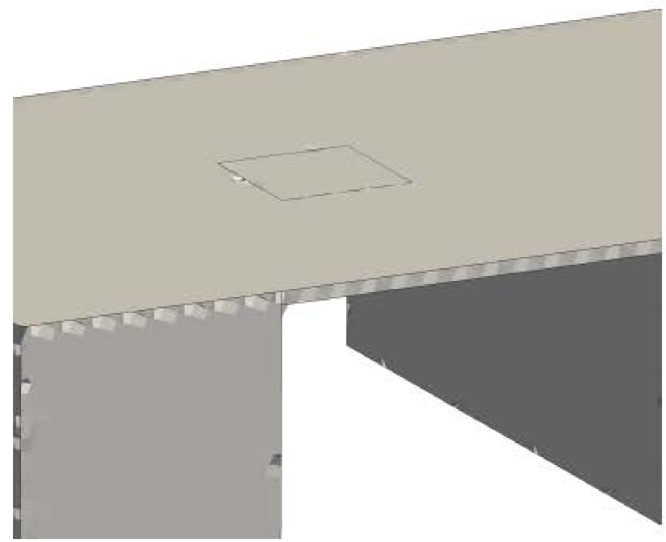
SnappyHexMesh OF230 new features

Enhanced thin surface snapping:

- Snap to nearest surface location
- Robust but can snap to 'wrong' side
- Instead: snap to nearest intersection



Snap to nearest

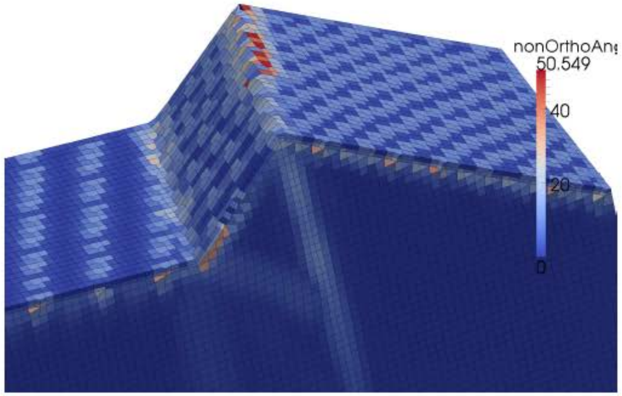


Snap to nearest intersection

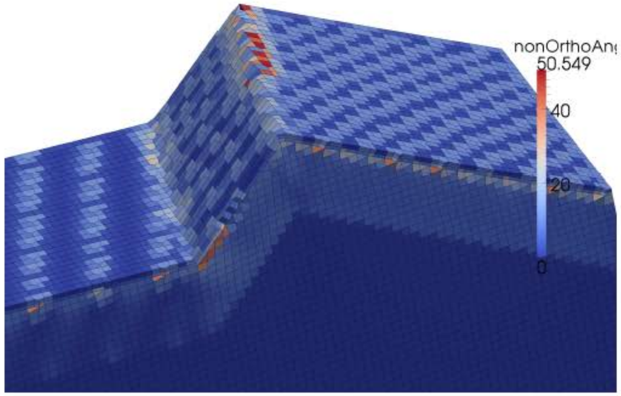
SnappyHexMesh OF230 new features

Enhanced layer shrinking:

- Layers created by shrinking mesh and adding layers in gap
- Shrinking algorithm uses distance between surfaces
- . . . even if it is far away
- Instead: limit distance with `nMedialAxisIter`



Unlimited shrinking steps

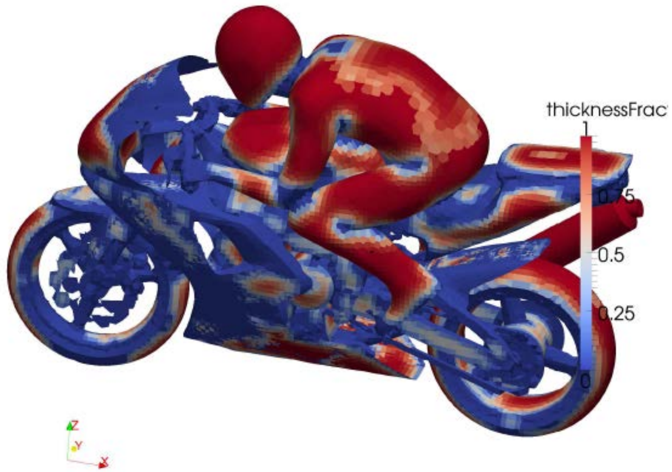


Shrinking steps limited to 10

SnappyHexMesh OF230 new features

Enhanced layer coverage statistics

- fields with boundary values
- `...nSurfaceLayers` number of layers
- `...thickness` overall thickness
- `...thicknessFraction` overall thickness as fraction of wanted thickness



Layer thickness fraction

SnappyHexMesh OF230 new features

Enhanced layer coverage statistics

- table with statistics
- `...thickness` overall thickness
- `...thicknessFraction` overall thickness as fraction of wanted thickness

patch	faces	layers	overall [m]	thickness [%]
-----	-----	------	---	---
lowerWall	5349	0.987	0.0244	92.7
motorBike_frt-fairing:001%1	5310	0.336	0.000847	18.1
motorBike_windshield:002%2	49	0.857	0.00334	71.2
motorBike_rr-wh-rim:005%5	119	0.487	0.000831	18
motorBike_rr-wh-rim:010%10	356	0.185	0.000251	5.57
motorBike_fr-wh-rim:011%11	502	0.241	0.000244	5.21
motorBike_fr-wh-brake-disk:012%12	46	0	0	0

SnappyHexMesh Current Developments

In the next major release you can expect:

- Improved multi-region meshing
- Improved feature snapping by splitting faces
- Ability to add layers to faceZones
- Compatibility with dynamic refinement/unrefinement

SnappyHexMesh Current Developments

Multi-region meshing

- Note: regions are defined using cellZones
- . . .with faceZones on region boundaries

Present behaviour

- cellZones and faceZones specified through closed surfaces
- . . . hard to do nested cellZones
- . . . neighbouring cellZones create inconsistent faceZones

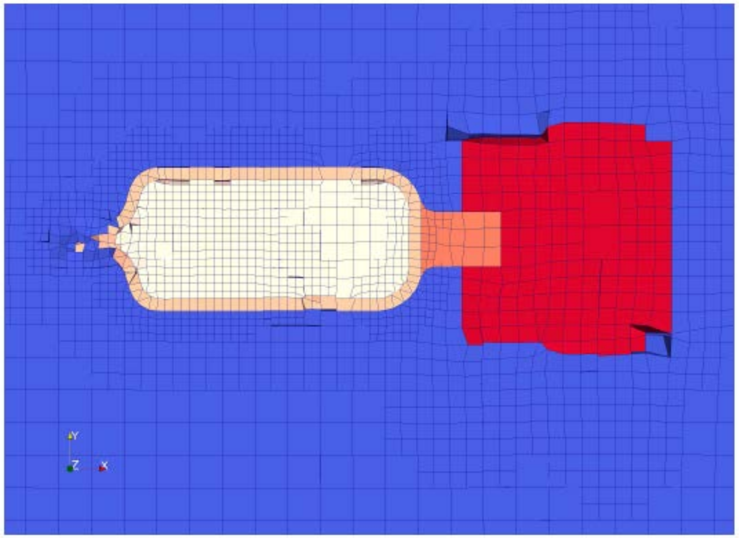
New behaviour

- specify **`locationsInMesh`** (plural)
- . . . each location defines a region (cellZone)
- . . . faceZones between regions automatically synthesised
- . . . optional patchType specification for faceZones
- . . . optional **`locationsOutsideMesh`** to warn for leaks

Multi-region meshing

- In practice: nested regions much easier. . .

```
castellatedMeshControls
{
    ...

    locationsInMesh
    (
        ((-0.09 0.001 -0.049)   leftSolid)
        ((0.01 0.0299 0.01)     none)
    );
}
```

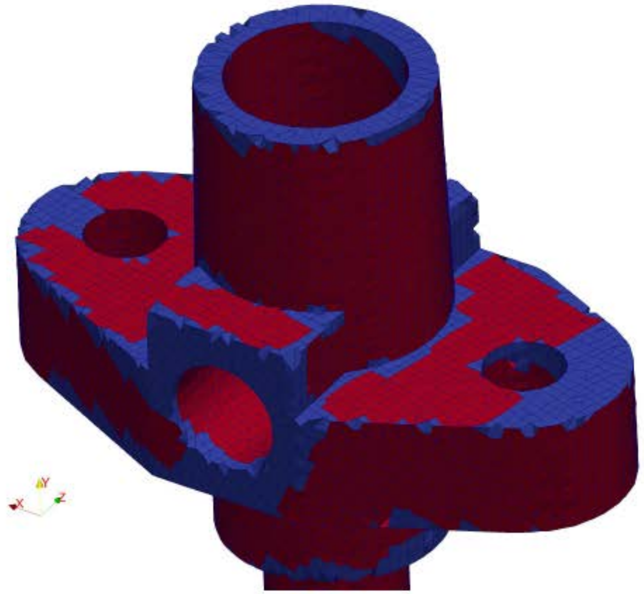


Nested cellZones

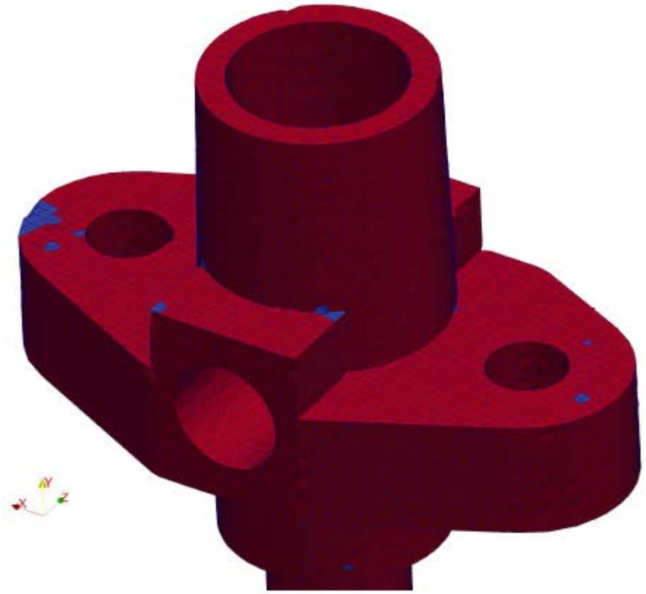
SnappyHexMesh Current Developments

Snapping: splitting boundary faces in feature snapping

- Present behaviour align boundary edges with feature edges
- New behaviour split boundary faces to create new boundary edges



No splitting of boundary faces

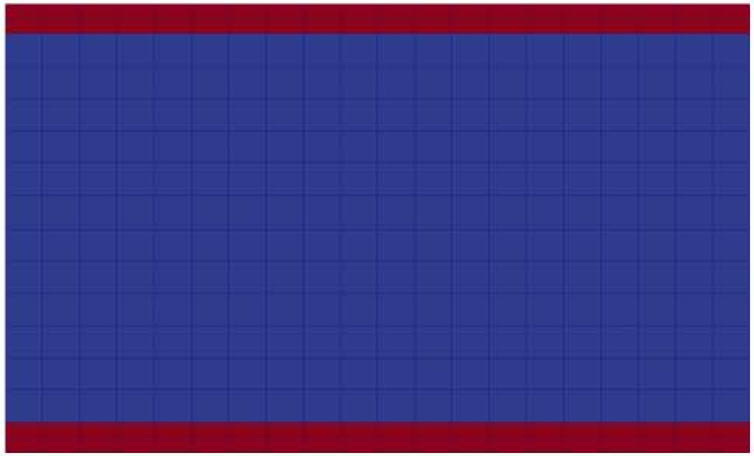


Splitting boundary faces

Layers on faceZones

- Present behaviour layers only on patches
- New behaviour layers on faceZones

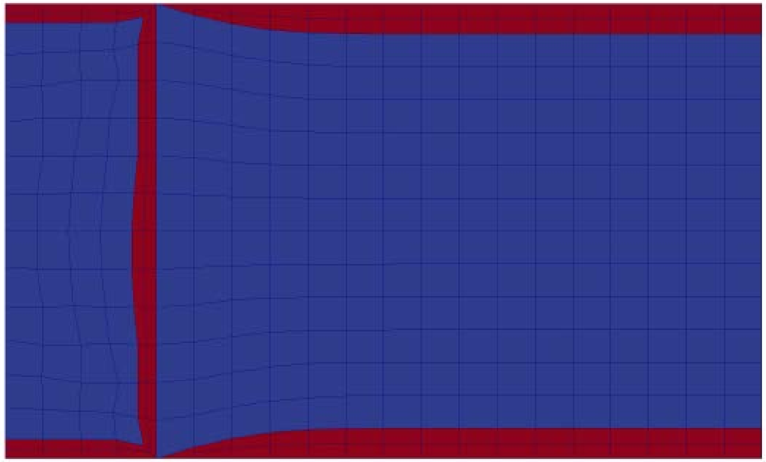
```
addLayersControls
{
    ...

    layers
    {
        wall
        {
            nSurfaceLayers 2;
        }
    }
}
```



Layers on faceZones

- Present behaviour layers only on patches
- New behaviour layers on faceZones

```
addLayersControls
{
    ...

    layers
    {
        wall
        {
            nSurfaceLayers 2;
        }
        faceZoneA
        {
            nSurfaceLayers 2;
        }
    }
}
```

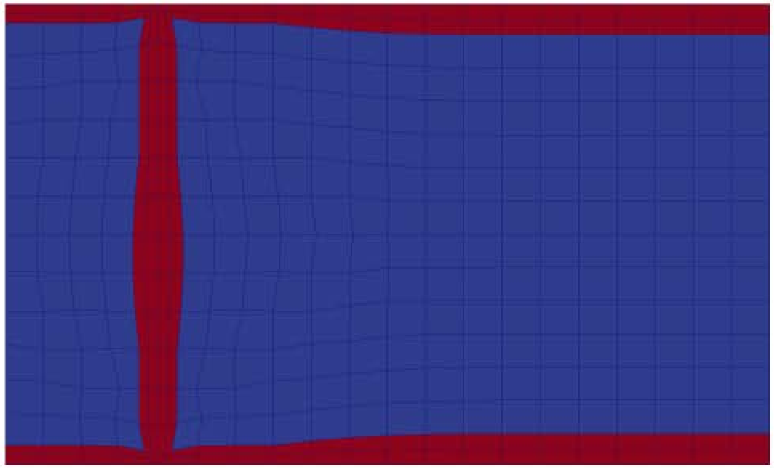


SnappyHexMesh Developments

Layers on faceZones

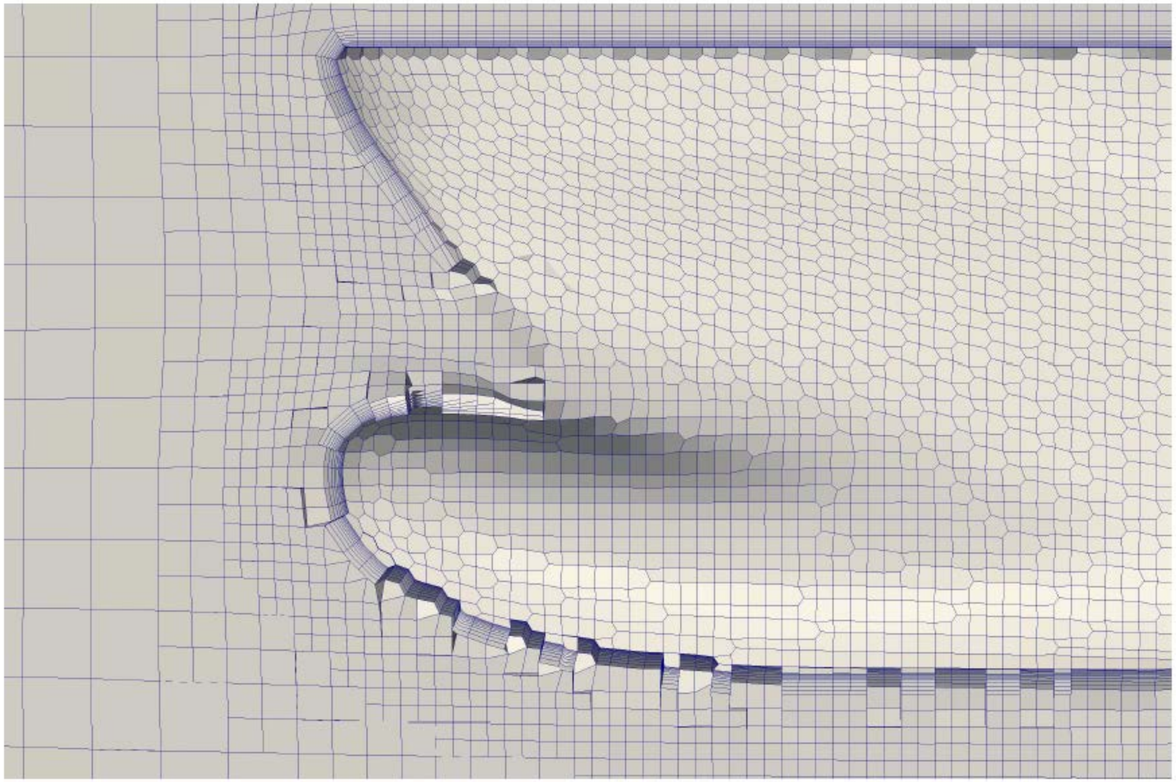
- Present behaviour layers only on patches
- New behaviour layers on faceZones

```
addLayersControls
{
    ...

    layers
    {
        wall
        {
            nSurfaceLayers 2;
        }
        faceZoneA
        {
            nSurfaceLayers 2;
        }
        faceZoneA_slave
        {
            nSurfaceLayers 2;
        }
    }
}
```



Offset surface layer addition:

- Layer addition currently uses mesh shrinking

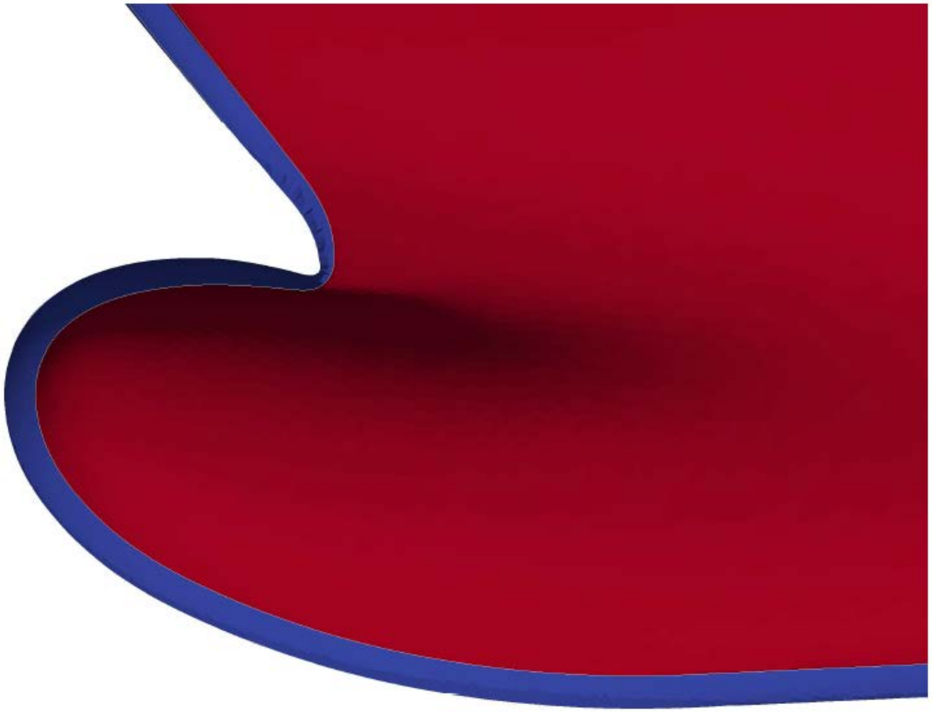


snappyHexMesh layer addition

SnappyHexMesh Developments

Offset surface layer addition

- Layer addition currently uses mesh shrinking
- Currently investigating using offset surfaces

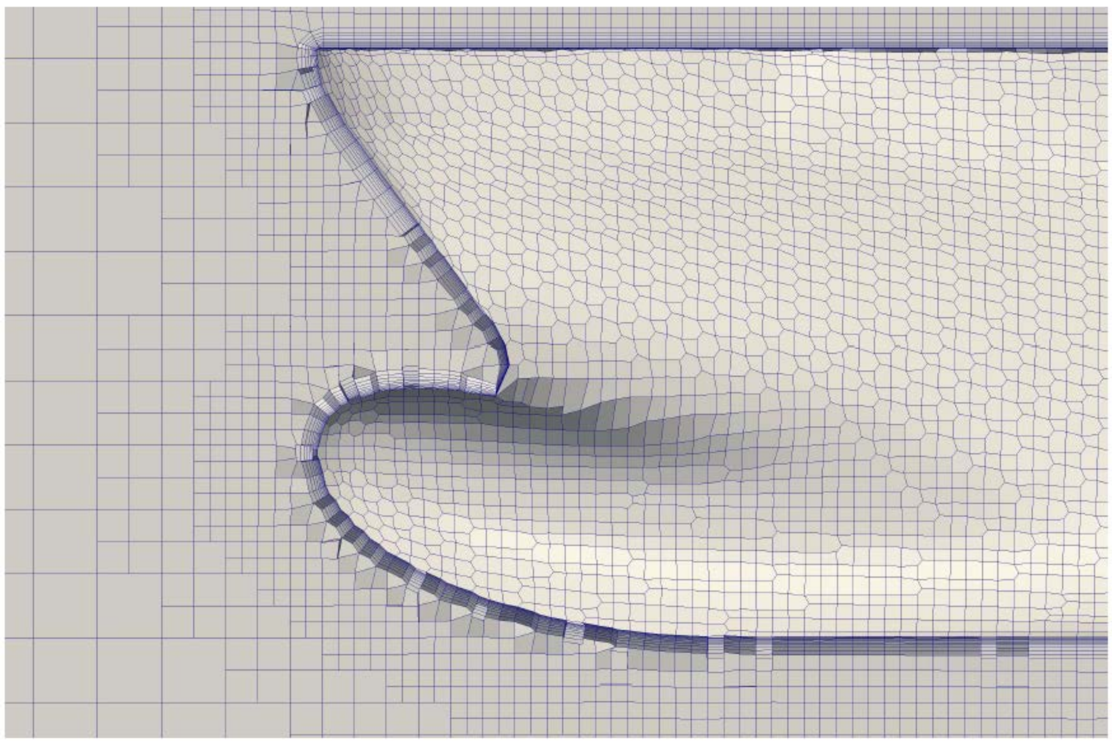


Offset surface generation

SnappyHexMesh Developments

Offset surface layer addition

- Layer addition currently uses mesh shrinking
- Currently investigating using offset surfaces
- . . . uses extrudeMesh from geometry to offset surface

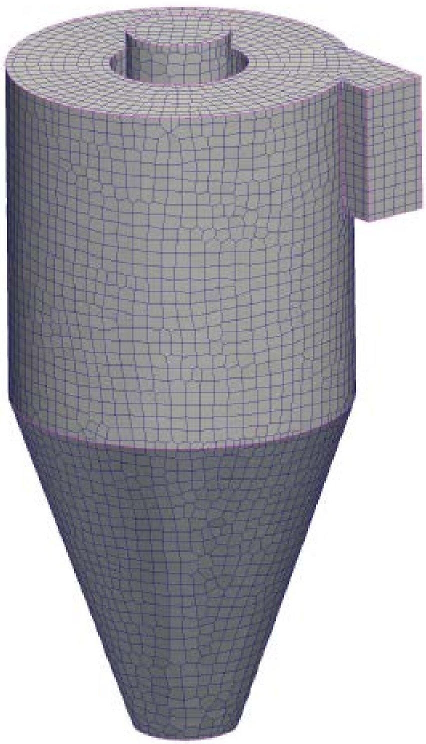


Adding extrusion between original mesh and geometry

foamyHexMesh

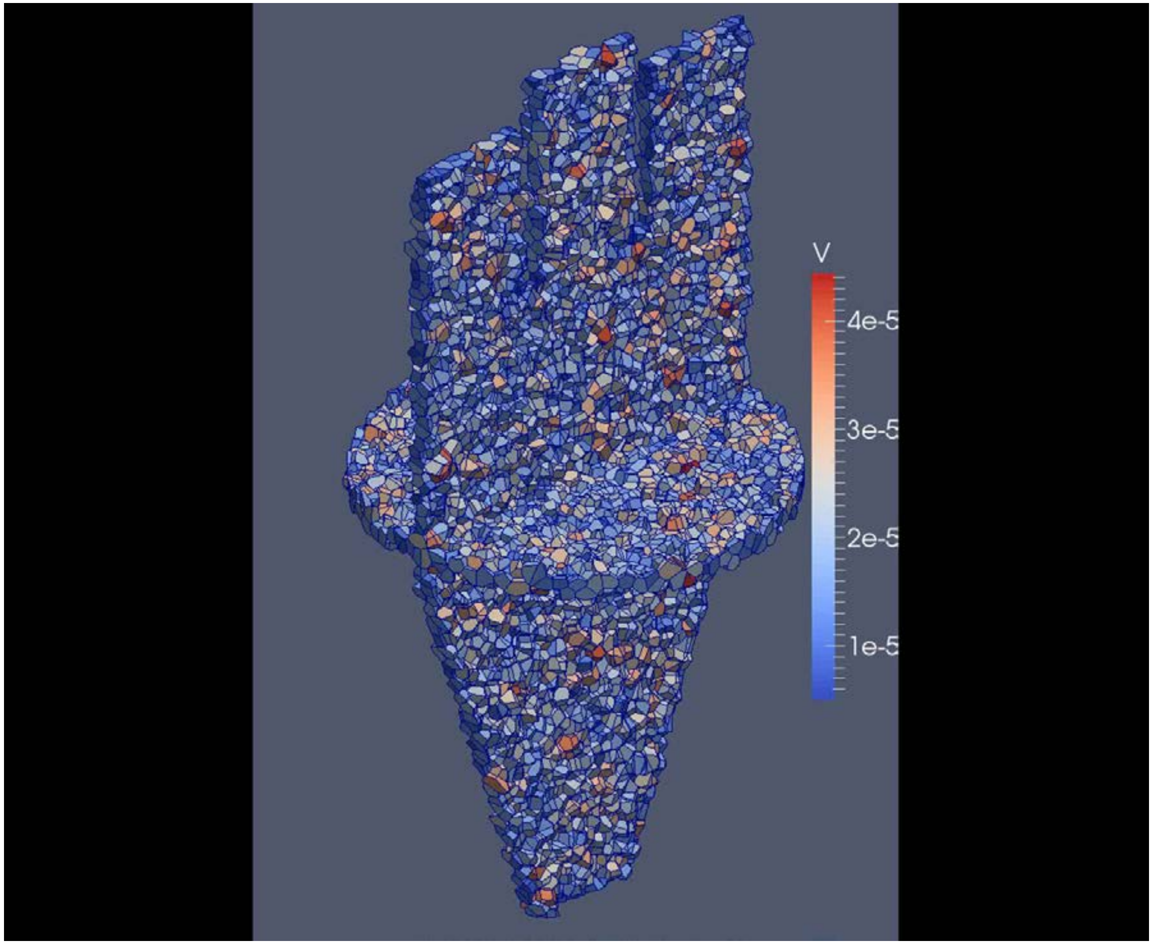
- foamyHexMesh fully functional and can be run in parallel
- ... requires user evaluation!
- Users should find many dictionary inputs familiar *e.g.* geometry

```
geometry
{
    cyclone.stl
    {
        name cyclone;
        type triSurfaceMesh;
    }
}

surfaceConformation
{
    locationInMesh      (0 0 0);

    geometryToConformTo
    {
        cyclone
        {
            featureMethod          extractFeatures;
            includedAngle          165;
        }
    }
}
```



foamyHexMesh

- foamyHexMesh in action. . .



Animation illustrating the point motion algorithm

Parallel operation

Three steps to parallel matrix solvers

1 Processor patch communication

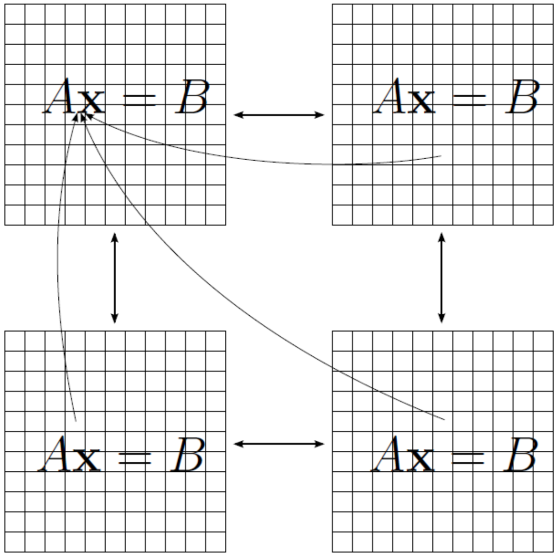
- Swap neighbouring processor cell values

2 Solve each processor domain

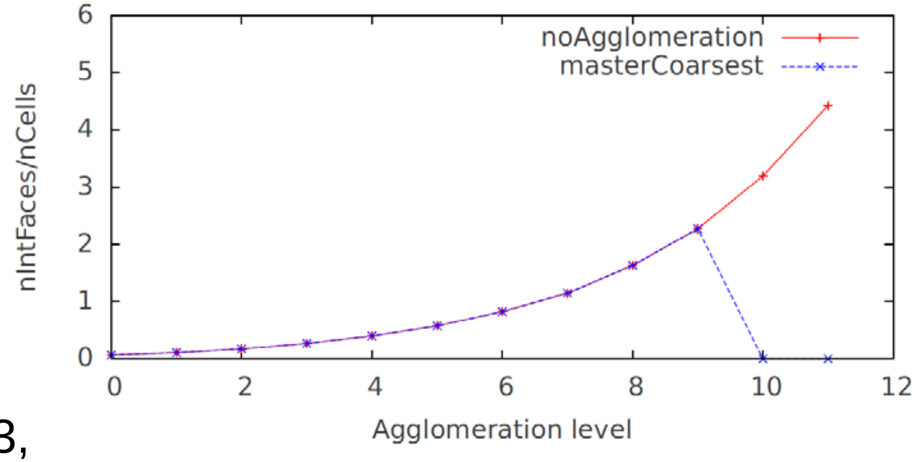
- Solve local cells

3 Calculate statistics (global reductions)

- . . . Same on all processors
- . . . By transferring to single processor and back



Parallel operation



Use more processors?

- Fewer cells per domain, faster to solve?
- . . .More communication in steps 1 and 3,
- . . .More neighbouring processors,
- . . .More processor faces v.s. cells (internal faces),
- . . .More explicit meaning worse convergence

Worst case: agglomeration in GAMG solver

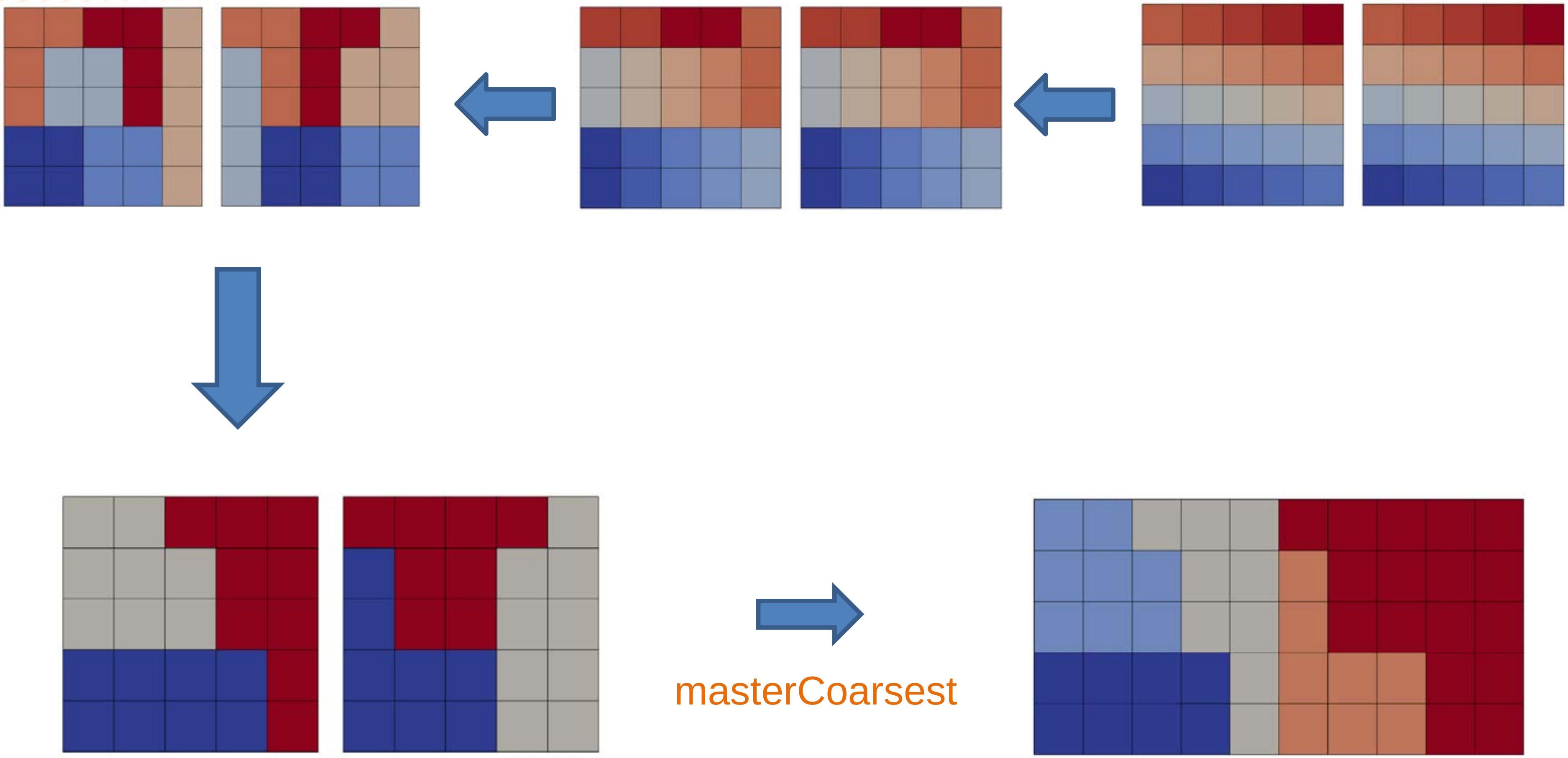
- Combines (clusters of) cells
- Number of cells at coarsest level very low (10?)
- Some agglomeration of processor faces
- At coarsest level CG solver
    - Few cells
    - With lots of processor faces
    - Still same cost of global reductions

Parallel operation

- Few cells (work) with lots of processor faces (communication)
- Agglomerate across processors!
  - Remove communication
  - More implicitness, less iterations in CG solver
  - Idle unused processors

- At what level to agglomerate which processors
- 'Normal', cell agglomeration: combine strongly coupled cells
- Choice of processorAgglomerators
  - none: no agglomeration, display statistics only
  - manual: select at what level which processors to combine
  - masterCoarsest: coarsest level on master processor
  - eager: at every level combine two neighbouring processors (keep number of cells constant)
  - cellFaceRatio: uses faceAreaPair cell-agglomeration method, weighted on number of inter-processor faces

Parallel operation



get it right®

Parallel operation example

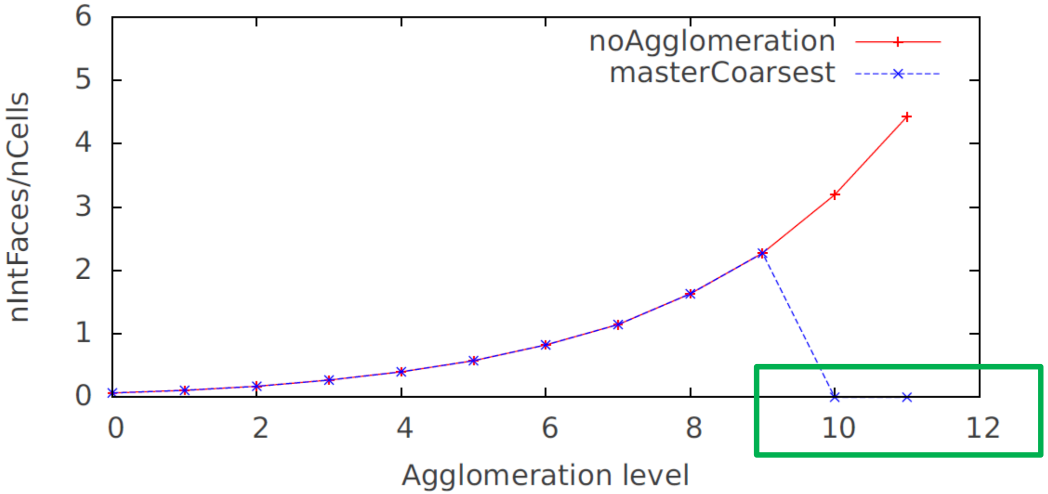
- 3.2M cells, pisoFoam
- 23 nodes, varying from 1 core to 4 cores per node
- 140k down to 35k cells per core
- Cluster: Intel E5-1650, QDR Infiniband and Gigabit Ethernet
- Using masterCoarsest processor agglomeration

```
p
{
    solver           GAMG;
    tolerance        1e-6;
    relTol           0.1;

    smoother         GaussSeidel;

    cacheAgglomeration true;

    nCellsInCoarsestLevel 50;
    agglomerator     faceAreaPair;
    mergeLevels      1;

    processorAgglomerator masterCoarsest;
};
```

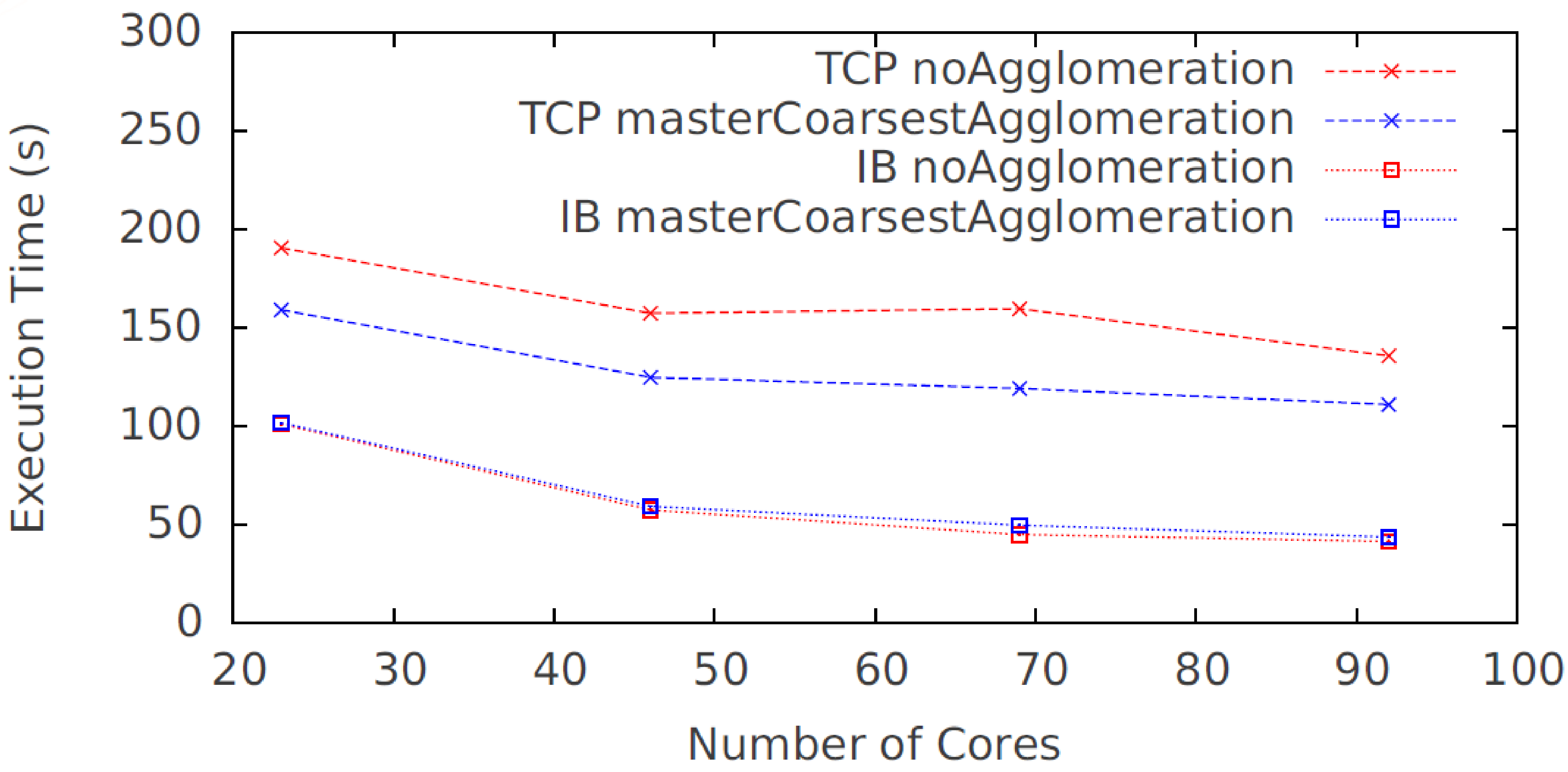
Extract from fvSolution

Parallel operation example

Level	nProcs	nCells avg	nCells max	nFaces/nCells avg	nFaces/nCells max	nInterfaces avg	nInterfaces max	nIntFaces/nCells avg	nIntFaces/nCells max
0	23	139972	141371	2.977	3.016	8.174	16	0.06763	0.1026
1	23	69074	70137	3.604	3.932	8.174	16	0.1096	0.1744
2	23	34266	34937	4.361	4.888	8.174	16	0.1725	0.2835
3	23	16910	17415	5.061	5.626	8.174	16	0.2711	0.4483
4	23	8396	8672	5.58	6.083	8.174	16	0.4011	0.6813
5	23	4136	4322	5.924	6.525	8.174	16	0.5773	1.005
6	23	2051	2149	6.102	7.074	8.174	16	0.8254	1.501
7	23	1009	1071	6.126	7.415	8.174	16	1.147	2.121
8	23	497	532	6.036	7.664	8.174	16	1.633	3.16
9	23	244	262	5.805	7.618	8.174	16	2.273	4.554
10	1	2747	2747	7.083	7.083	0	0	0	0
11	1	1340	1340	7.225	7.225	0	0	0	0



- Effect of processor agglomeration



Boundedness and MULES

MULES = Multi-dimensional Universal Limiter for Explicit Solution

FVM does not guarantee boundedness.

With MULES we can guarantee boundedness:
- MULES explicit with sub-cycling ✓
- MULES implicit with limiter iteration ✓
- MULES predictor-corrector ✓ Recommended and available from OF220
- MULES for 2nd-order transport ✓
- MULES for 2nd order time... UNDER DEVELOPMENT
- MULES for non-orthogonal diffusion correction... UNDER DEVELOPMENT
- MULES for coupled variables... UNDER DEVELOPMENT

The future of finite-volume for bounded properties is MULES.

- MULES introduces significant code complexity
- requires some core reorganization of OpenFOAM to preserve convenient top-level finite-volume language and to ease maintenance.

Final remarks

- All developments will be undertaken on behalf of the OpenFOAM foundation
- Copyright transferred to the OpenFOAM Foundation
- Released publicly under the GPLv3

For those interested in Discrete Particle Method:

Check **MPPICFoam** much faster than DPMFoam

Thank you for

get it right®

www.esi-group.com

Your attention