

# **Examples of Simulations with OpenFOAM**

## **Complex Physics and Industrial CFD**

**Hrvoje Jasak**

`h.jasak@wikki.co.uk, hrvoje.jasak@fsb.hr`

**Wikki Ltd, United Kingdom**

**Faculty of Mechanical Engineering and Naval Architecture**

**University of Zagreb, Croatia**

**Seoul National University, 14 April 2011**

## Complex Physics Topics and New Capabilities

- Viscoelastic flow modelling
- Free surface flows with compressibility effects: under-water explosions
- Flash boiling simulations
- Block matrix in use: strong coupling in multi-phase VOF
- Some new fluid-structure interaction results
- New mesh handling features: Overset Grid Method, Immersed Boundary Method, remeshing with Tetrahedral Edge Swapping

## Industrial CFD Topics

- External aerodynamics with LES
- Simulations in metallurgical industry
- Radial Basis Function (RBF) and geometric shape optimisation
- OpenFOAM Turbo Tools
- Automotive Simulations
- Complex mesh motion cases

## Example of Capabilities of OpenFOAM in Complex Physics and Industrial CFD

- This is only a part of the OpenFOAM capabilities!
- Chosen for relevance and illustration of the range of capabilities rather than exhaustive illustration of range of capabilities
- In some cases, simplified geometry is used due to confidentiality
- Regularly, the work resulted in a new solver; in many cases, it is developed as an extension or combination of existing capabilities

## Description of Simulation and Setup

- Physics and numerical method setup
- Standard or customised solver; details of mesh resolution and customisation

MSc Thesis: Jovani Favero, Universidade Federal de Rio Grande del Sul, Brazil

- Viscoelastic flow model:

$$\nabla \cdot \mathbf{u} = 0$$

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = -\nabla p + \nabla \cdot \boldsymbol{\tau}_s + \nabla \cdot \boldsymbol{\tau}_p$$

where  $\boldsymbol{\tau}_s = 2\eta_s \mathbf{D}$  is the solvent stress contribution and  $\boldsymbol{\tau}_p$  is the **polymeric part of the stress**, non-Newtonian in nature

- Depending on the model,  $\boldsymbol{\tau}_p$  is solved for: saddle-point problem
- Models introduce “upper”, “lower” or Gordon-Schowalter derivatives, but we shall consider a general form: standard transport equation in relaxation form

$$\frac{\partial \boldsymbol{\tau}_p}{\partial t} + \nabla \cdot (\mathbf{u} \boldsymbol{\tau}_p) = \frac{\boldsymbol{\tau}^* - \boldsymbol{\tau}_p}{\delta}$$

where  $\delta$  is the relaxation time-scale

- Problem:  $\boldsymbol{\tau}_p$  dominates the behaviour and is explicit in the momentum equation

## Model Implementation Recipe

- Recognise  $\boldsymbol{\tau}^*$  as the equilibrium stress value: make it implicit!

$$\nabla \cdot \boldsymbol{\tau}^* = \nabla \cdot \left[ \boldsymbol{\kappa} \cdot \frac{1}{2} \left( \nabla \mathbf{u} + (\nabla \mathbf{u})^T \right) \right]$$

- Calculate implicit viscoelastic viscosity:

$$\boldsymbol{\kappa} = \boldsymbol{\tau}^* \cdot \left[ \frac{1}{2} \left( \nabla \mathbf{u} + (\nabla \mathbf{u})^T \right) \right]^{-1}$$

- Split complete stress into implicit and explicit component

$$\begin{aligned} \nabla \cdot \boldsymbol{\tau}_p &= \nabla \cdot \boldsymbol{\tau}^* + \nabla \cdot \boldsymbol{\tau}_{\text{corr}} \\ &= \nabla \cdot (\boldsymbol{\kappa} \cdot \nabla \mathbf{u}) + \nabla \cdot \boldsymbol{\tau}_{\text{corr}} \end{aligned}$$

## Implemented Viscoelastic Models

- **Kinetic Theory Models:** Maxwell linear; UCM and Oldroyd-B; White-Metzner; Larson; Cross; Carreau-Yasuda; Giesekus; FENE-P; FENE-CR
- **Network Theory of Concentrated Solutions and Melts Models:** Phan-Thien-Tanner linear (LPTT); Phan-Thien-Tanner exponential (EPTT); Feta-PTT
- **Reptation Theory / Tube Models:** Pom-Pom model; Double-equation eXtended Pom-Pom (DXPP); Single-equation eXtended Pom-Pom (SXPP); Double Convected Pom-Pom (DCPP)
- **Multi-Mode Form:** The value of  $\tau_p$  is obtained by the sum of the K modes

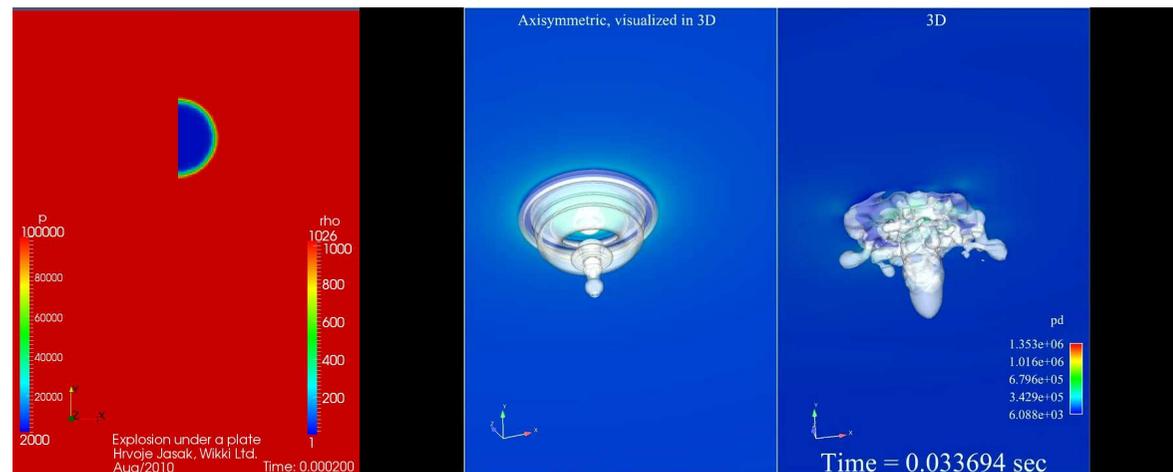
$$\tau_p = \sum_{K=1}^n \tau_{pK}$$

## Flow Solvers Implemented by Jovani Favero: **Example Simulation**

- Single-phase non-Newtonian solver based on transient SIMPLE
- Multi-phase free surface VOF solver: viscoelastic in each phase
- Support for topological changes: syringe ejection

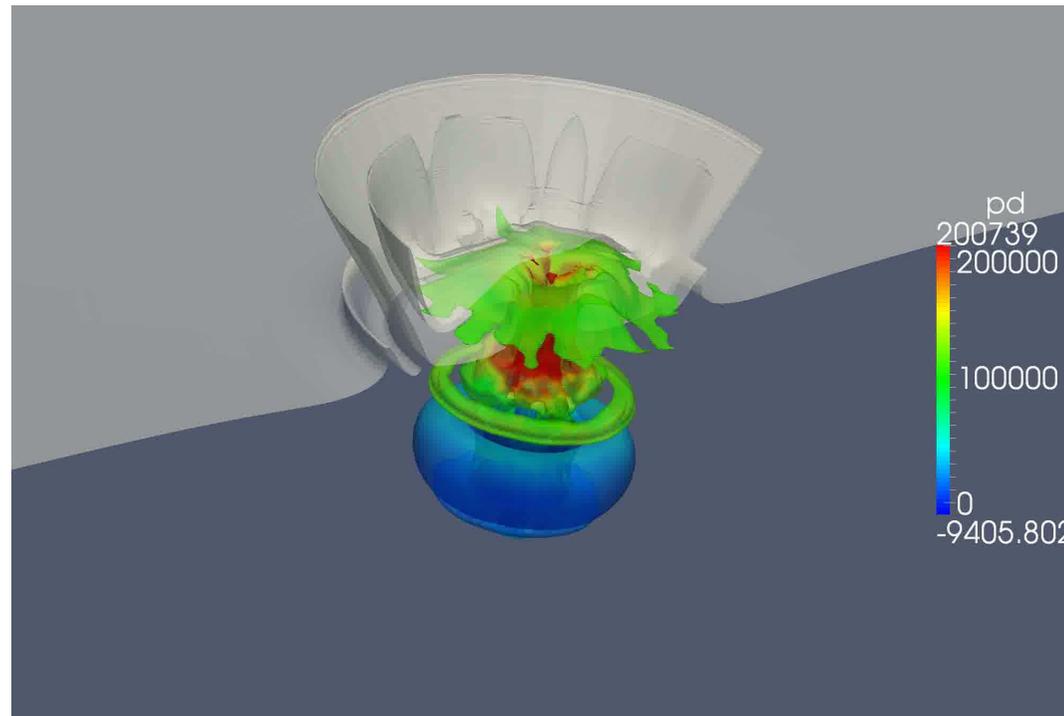
## Simulation of Under-Water Explosions

- This is ongoing collaboration with Johns Hopkins APL, Penn State University and Wikki: working hard for almost a year
- Dominating effects of compressibility in air and water: massive change in density, with propagating pressure waves
- Pressure ranges from 500 bar to 20 Pa
- Stiff numerics: collapse of over-expanded bubble due to combined compressibility of both phases are the basis of the phenomenon
- Test cases: Rayleigh-Plesset oscillation, undex under a plate, explosion
- Eric Paterson, Scott Miller, David Boger, Penn State; Ashish Nedungadi, JHU-APL



First Simulations of Under-Water Explosions: Eric Paterson, Penn State

- Bubble of high initial pressure expands after explosion
- Initial pressure pulse is very fast - with little effect
- Bubble collapse creates re-entrant jet which pierces the free surface
- Stability problems resolved in segregated solver
- Next phase: **block-coupled  $p - \alpha$  solution algorithm**



Flash-Boiling Flows: **Shiva Gopalakrishnan, David P. Schmidt, UMass Amherst**

- The fundamental difference between flash boiling and cavitation is that the process has a higher saturation pressure and temperature: higher density
- Enthalpy required for phase change is provided by inter–phase heat transfer
- **Jakob number**: ratio of sensible heat available to amount of energy required for phase change

$$Ja = \frac{\rho_l c_p \Delta T}{\rho_v h_{fg}}$$

- Equilibrium models are successful for cavitation since Ja is large and timescale of heat transfer is small. Flash boiling represents a finite rate heat transfer process:  
**Homogeneous Relaxation Model (HRM)**

$$\frac{Dx}{Dt} = \frac{\bar{x} - x}{\Theta}; \quad \Theta = \Theta_0 \epsilon^{-0.54} \phi^{1.76}$$

$x$  is the quality (mass fraction), relaxing to the equilibrium  $\bar{x}$  over a time scale  $\Theta$

- The timescale  $\Theta$  is obtained from empirical relationship: Downar–Zapolski [1996].  
 $\epsilon$  is the void fraction and  $\phi$  is the non–dimensional pressure.

## Flash-Boiling Flows: Numerical Method

- **Conservation of Mass**

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\phi_v \rho) = 0$$

- **Conservation of Momentum**

$$\frac{(\partial \rho U^0)}{\partial t} + \nabla \cdot (\phi U^0) = -\nabla p^n + \nabla \cdot (\mu \nabla U^0)$$

- **Pressure Equation**

$$\begin{aligned} \frac{1}{\rho} \frac{\partial \rho}{\partial p} \Big|_{x,h} \left( \frac{\partial(\rho p^{k+1})}{\partial t} + \nabla \cdot (\rho U p^{k+1}) \right) + \rho \nabla \cdot \phi^* - \rho \nabla \frac{1}{a_p} \nabla p^{k+1} \\ + M(p^k) + \frac{\partial M}{\partial p} (p^{k+1} - p^k) = 0 \end{aligned}$$

The HRM model term is denoted as  $M(= \frac{Dx}{Dt})$ . The superscripts  $k$  and  $k + 1$  are the corrector steps for the pressure equation.

## Conservation of Mass

```
solve
(
    fvm::ddt(rho) + fvm::div(phiV, rho)
);
```

## Conservation of Momentum

```
fvVectorMatrix UEqn
(
    fvm::ddt(rho, U) + fvm::div(phi, U) - fvm::laplacian(mu, U)
);
solve(UEqn == -fvc::grad(p));
```

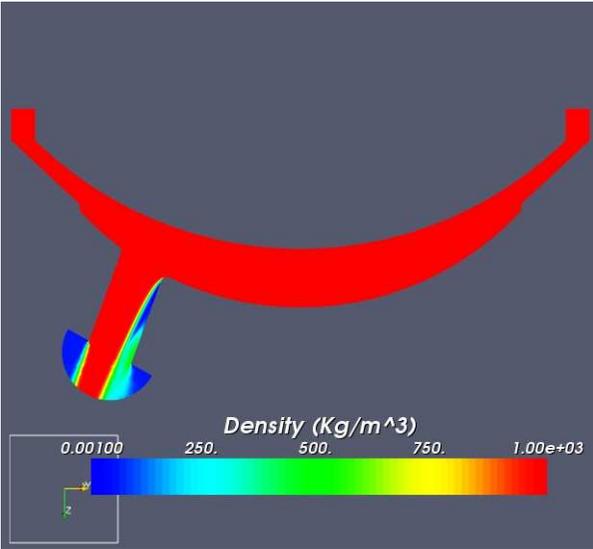
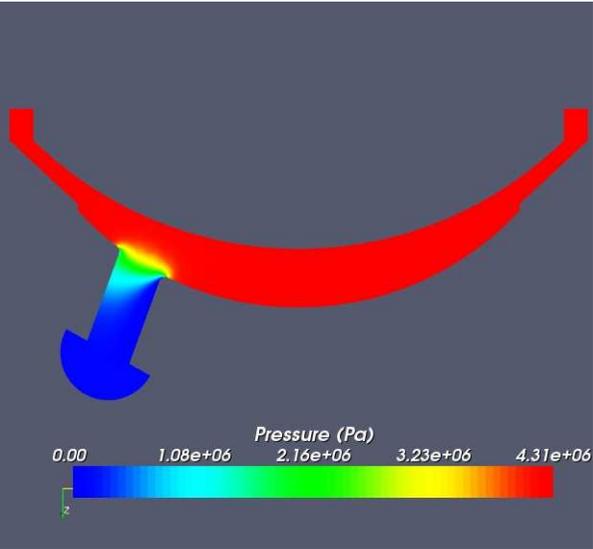
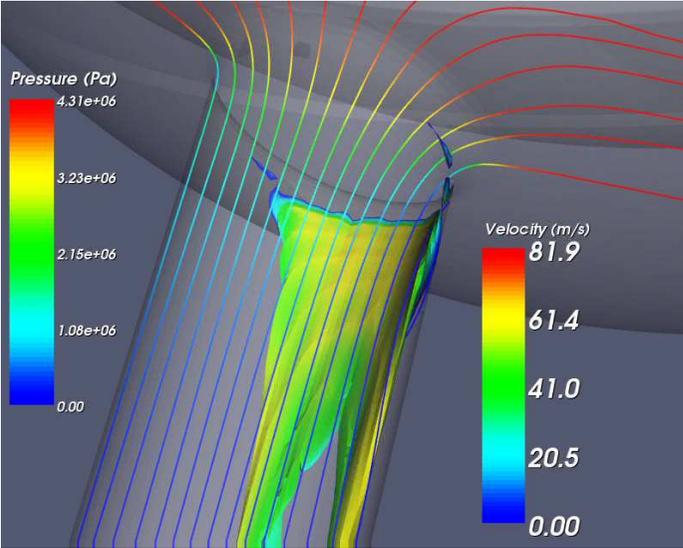
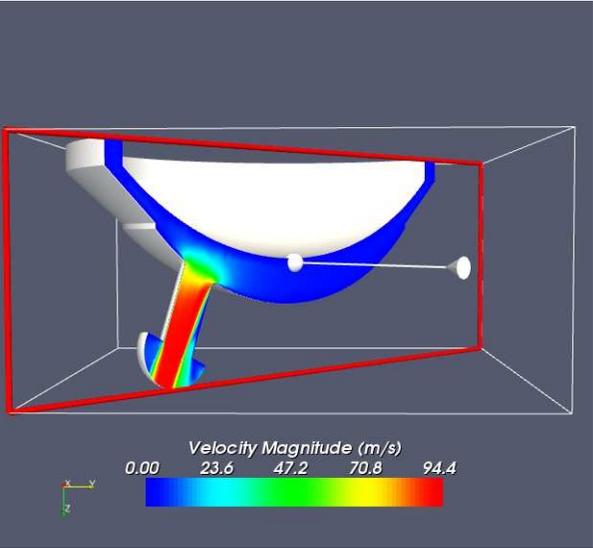
## Pressure Equation

```
fvScalarMatrix pEqn(fvm::laplacian(rUA, p));

solve
(
    psi/sqr(rho)*(fvm::ddt(rho, p) + fvm::div(phi, p))
    + fvc::div(phiVStar) - pEqn
    + MSave + fvm::SuSp(dMdp, p) - dMdp*pSave
);
```

# Flash-Boiling Simulations

Asymmetric Fuel Injector Nozzle-Design from Bosch GmbH.



## Multi-Phase Volume-of-Fluid Solver

- System of equations contains multiple VOF equations and global continuity handled by a pressure equation in standard form
- The phase for which VOF is solved first dominates the other phases: this is not acceptable; flipping the order of solution moves the problem around
- Aim: **Coupled pressure based approach to achieve physical behaviour**
- Solution strategy: solve  $\alpha_i$  transport equations, volumetric continuity and closure equation in a strongly coupled manner, making the coupling terms implicit!
- To make this run, we shall use the block matrix and block solver (Jasak and Clifford, 2009)
- Credit goes to **Kathrin Kissling and Julia Springer, NUMAP-FOAM 2009**

## Multi-Phase Volume-of-Fluid Solver: Equation Set and Coupling

- Volume fraction transport equation, with separated pressure-driven flux terms

$$\left[ \frac{\partial [\alpha_i]}{\partial t} \right] + \left[ \nabla \cdot \left( \left( \left( \frac{A_H}{A_D} \right)_f \bullet \mathbf{s}_f + \frac{1}{A_D} (\sigma \kappa)_f \nabla_f^\perp \alpha + \frac{1}{A_D} (\mathbf{g} \cdot \mathbf{x})_f |\mathbf{s}_f| \nabla_f^\perp \rho \right) [\alpha_i] \right) \right] - \left[ \nabla \cdot \left( \left( \frac{1}{A_D} \right) \nabla p^* [\alpha_i] \right) \right] + \left[ \nabla \cdot \left( [\alpha_i] \sum_{k=1, k \neq i}^N \alpha_k \phi_{r,ij} \right) \right] = 0$$

- Pressure equation, with separated phase fluxes (coupling terms)

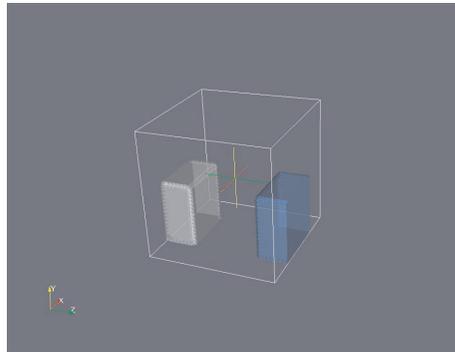
$$\left[ \nabla \cdot \left[ \left( \frac{1}{A_D} \right)_f \nabla [p^*] \right] \right] = \nabla \cdot \left( \sum_{i=1}^N \alpha_i \phi^* \right)$$

- Closure equation and definition of fluxes

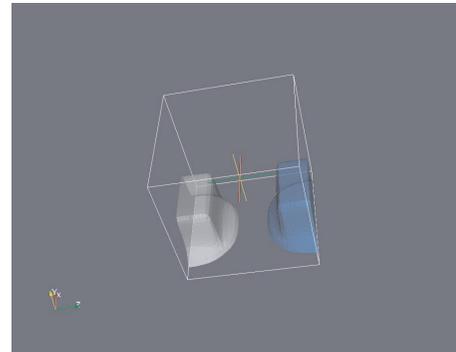
$$\sum_i \alpha_i = 1 \quad \phi^* = \sum_i \alpha_i^N \phi^*$$

## Multi-Phase Volume-of-Fluid Solver: Solution Strategy

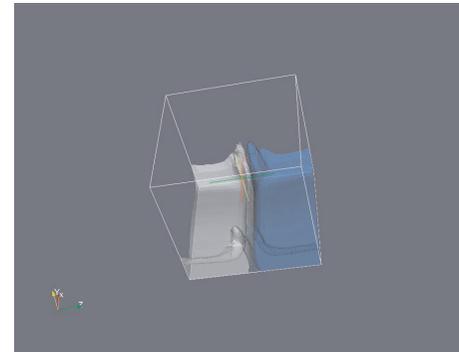
- Above equations are dumped into a block matrix format, with coefficient size  $N + 1$ :  $(p^*, \alpha_i)$  and solved in a block-coupled manner
- Result: strong coupling between  $\alpha$ s and  $p$ : no dominant phase



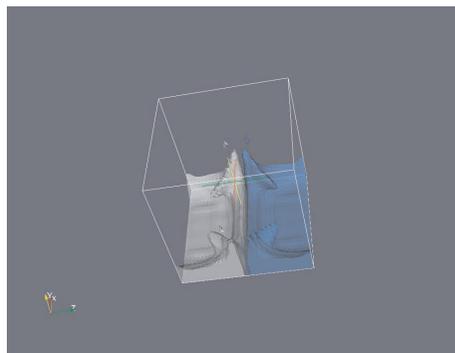
$t = 0$



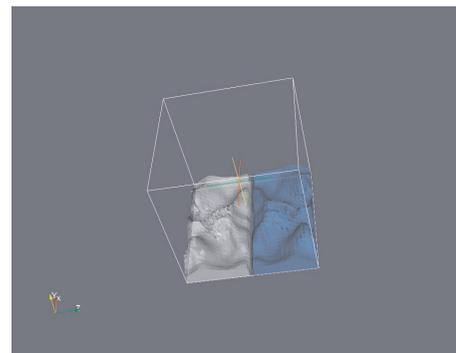
$t = 1$



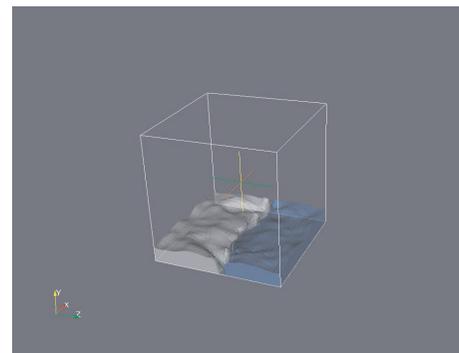
$t = 2$



$t = 3$



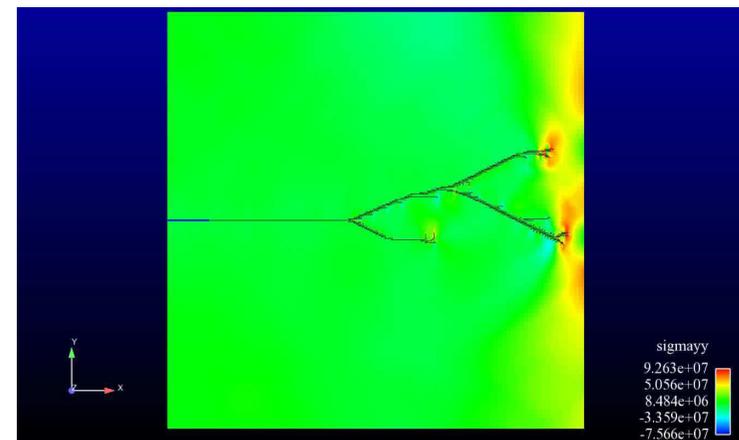
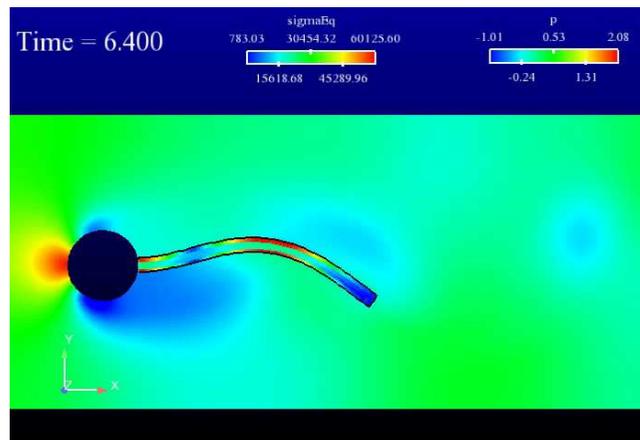
$t = 4$



$t = 5$

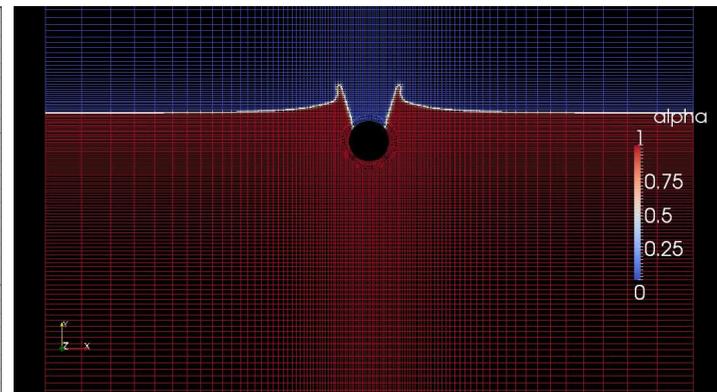
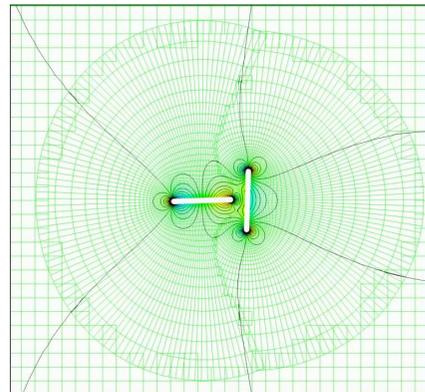
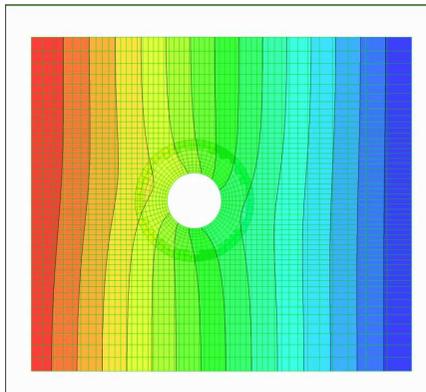
## Fluid-Structure Coupling Capabilities in OpenFOAM

- As a Continuum Mechanics solver, OpenFOAM can deal with both fluid and structure components: easier setup of coupling
- (Parallelised) surface coupling tools implemented in library form: facilitate coupling to external solvers without “coupling libraries” using proxy surface mesh
- Structural mechanics in OpenFOAM targeted to non-linear phenomena: consider best combination of tools
  - Large deformation formulation in absolute Lagrangian formulation
  - Independent parallelisation in the fluid and solid domain
  - Parallelised data transfer in FSI coupling
- Dynamic mesh tools and boundary handling used to manipulate the fluid mesh



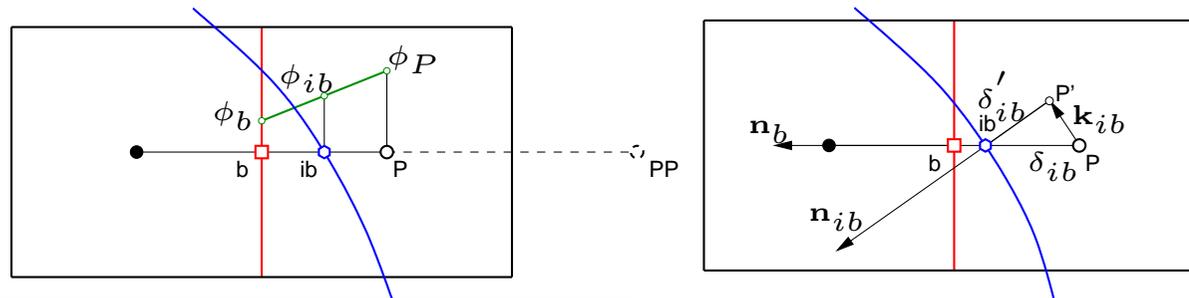
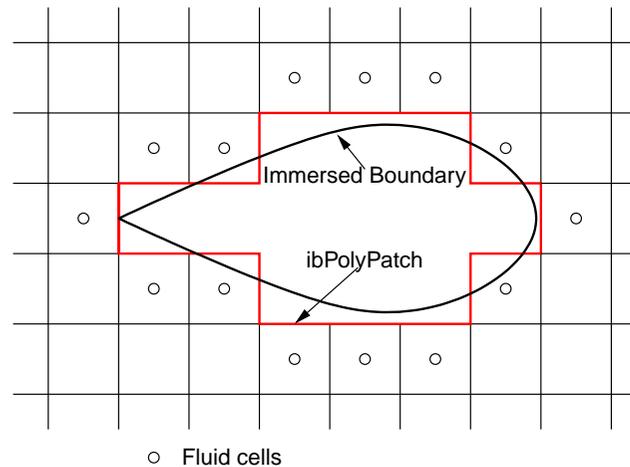
## `foamedOver`: Overset Grid Technology in OpenFOAM

- Work by David Boger, Penn State University using SUGGAR and DirtLib libraries developed by Ralph Noack, Penn State (must mention Eric Paterson!)
- Overset Grid Technology
  - Multiple components meshed individually, with overlap
  - Hole cutting algorithm to remove excess overlap cells
  - Mesh-to-mesh interpolation with implicit updates built into patch field updates and linear solver out-of-core operations
- Body-fitted component meshes: preserving quality and near-wall resolution
- Simple mesh motion and geometrical studies (replacing individual components)
- Overset grid is physics-neutral! Currently testing for free surface flows



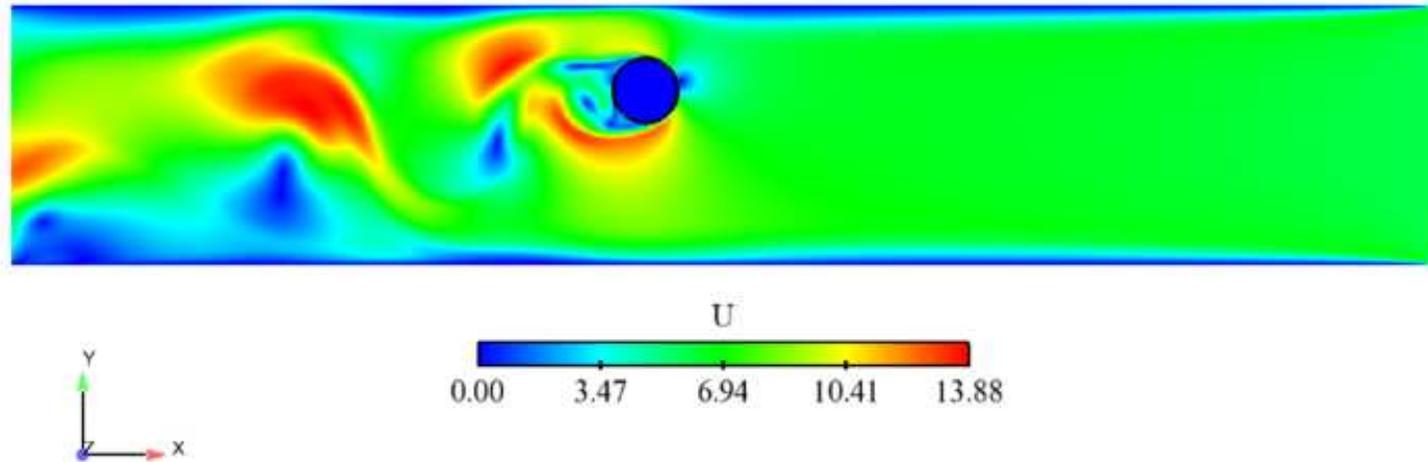
## Immersed Boundary Method

- Handling of moving obstacles in the flow domain whose size is larger than mesh resolution: covering multiple cells
- Mesh topology and connectivity does not change: **immersed STL surface**
- Presence of boundary implicitly accounted for in discretisation, with appropriate handling of boundary conditions

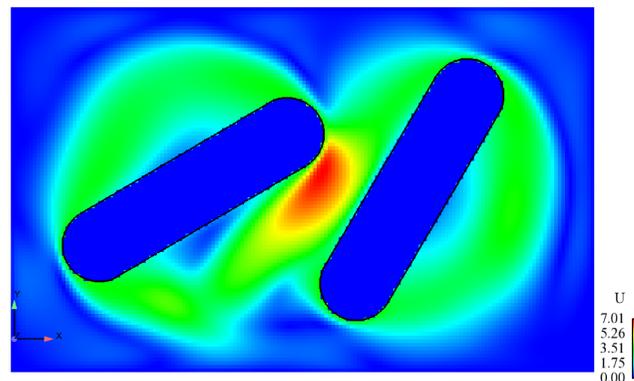


## Immersed Boundary Method: Examples

- Laminar flow around a 2-D moving circular cylinder in a channel

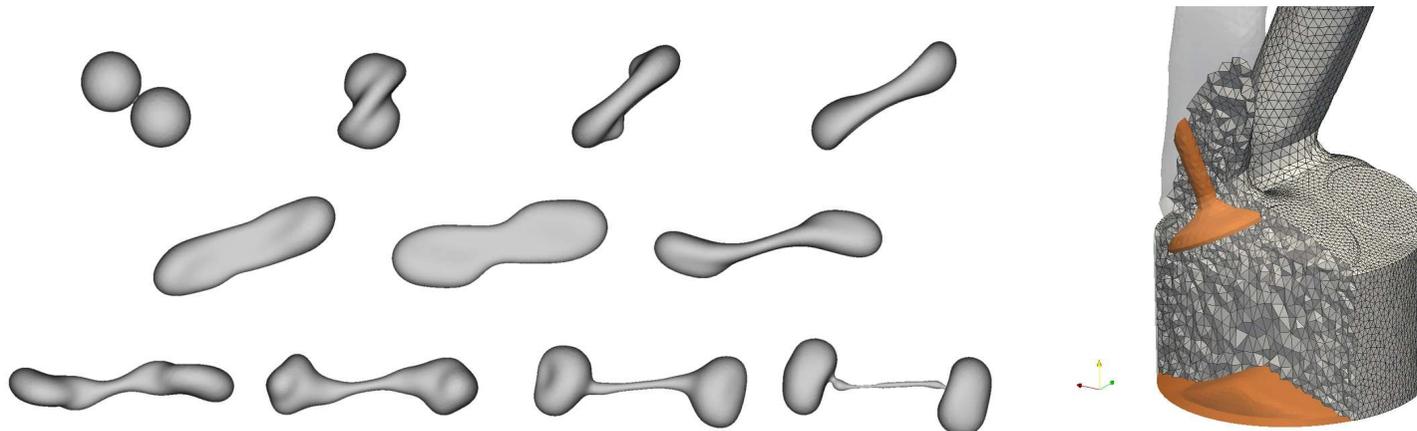


- Laminar flow around two counter-rotating elements in a cavity



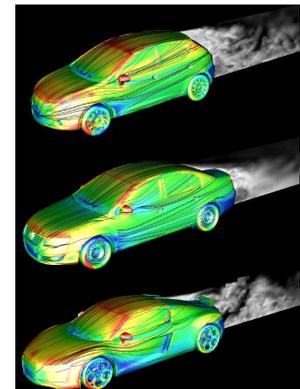
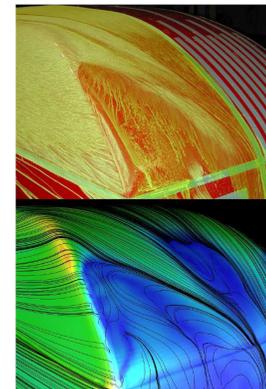
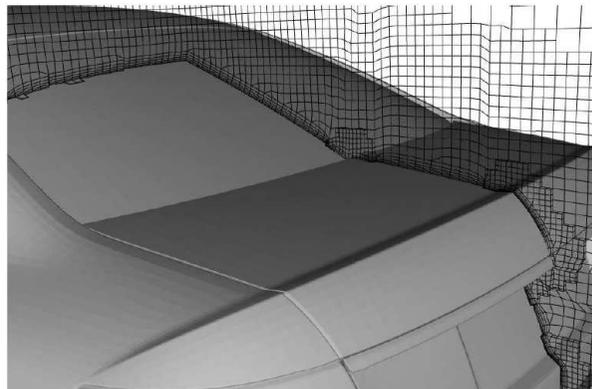
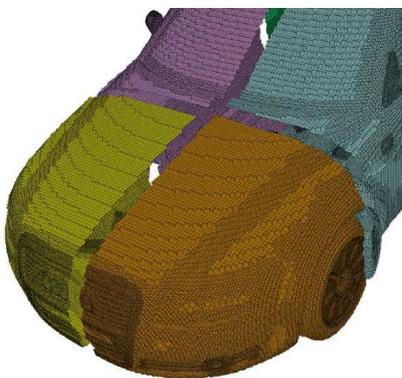
## Re-Meshing with Tetrahedral Edge Swapping

- In cases where mesh motion involves topological change at the boundary or unpredictable mesh deformation, topological change machinery is impractical: cannot decide a-priori where to place topology modifiers
- **Dynamic remeshing using tetrahedral edge swapping**
  - Motion is prescribed on external boundaries
  - Tetrahedral cell quality examined continuously: bad cells trigger automatic remeshing without user interaction: answers to `dynamicMesh` interface
  - Implemented by **Sandeep Menon, UMass Amherst** as a ready-to-use library
- Example: viscoelastic droplet collision using free surface tracking
- Can be used for all dynamic mesh cases: ultimate ease of mesh setup!



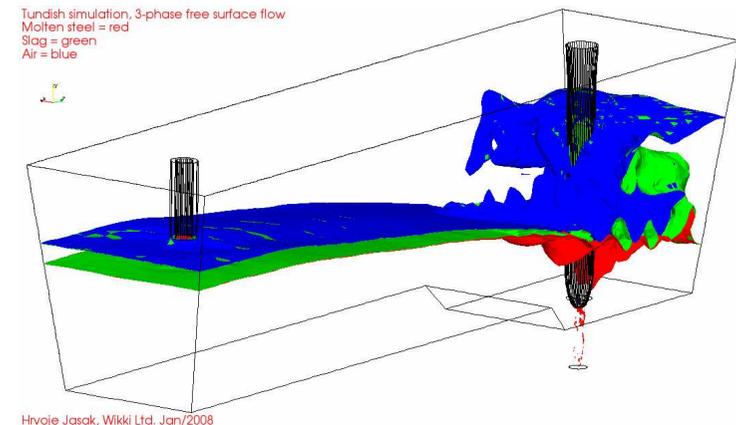
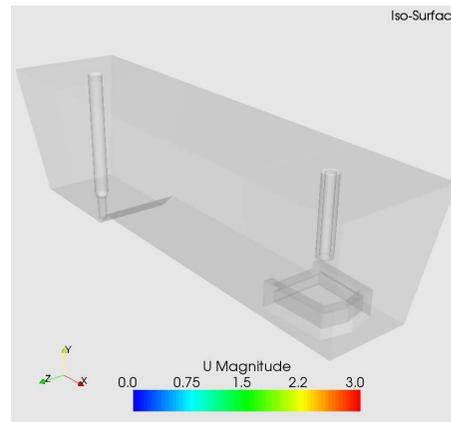
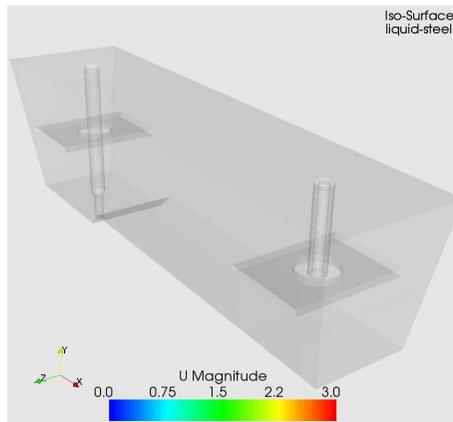
## Detached Eddy Simulation for External Aerodynamics

- Pushing state-of-the-art by applying Detached Eddy Simulation (DES) to full car body external aerodynamics simulations: **native solver and mesher, no change**
- Increase in simulation cost over transient RANS is over 1 order of magnitude!
- **Controlling the Cost of Full Car DES:**
  - Automated meshing and simulation environment, from STL surface of the car body to averaged DES results and forces
  - Hex-core mesher with near-wall layers and local refinement: mesh is designed to make it good for second-order LES numerics with minimal cost
  - No parallel license cost of CFD solver: simulations run on approx. 200 CPUs
- Improvement in  $C_D$ ,  $C_L$  and force-per-component predictions due to better capturing of turbulence and transient flow features



## Three-Phase Free Surface Flow in a Tundish

- 3 phases with extreme density ratio: liquid steel, liquid slag, air (7000:2500:1)
- Note the presence of multiple phase-to-phase interfaces: using consistent discretisation for multiple phase  $\alpha$  equations
- Temperature-dependent properties of slag and steel
- Simultaneous filling and pouring with large outlet velocity
- Examples
  - Simulation of a Turbo-Stop device for initial fill
  - Simulated break-down of a slag layer: **detail of flow at exit nozzle**



## Assembling a Matrix for Conjugate Heat Transfer Problems

- OpenFOAM supports multi-region simulations, with possibility of separate addressing and physics for each mesh: multiple meshes, with local fields
- Some equations present only locally, while others span multiple meshes

```
coupledFvScalarMatrix TEqns(2);
```

```
TEqns.hook
```

```
(  
    fvm::ddt(T) + fvm::div(phi, T)  
    - fvm::laplacian(DT, T)  
);
```

```
TEqns.hook
```

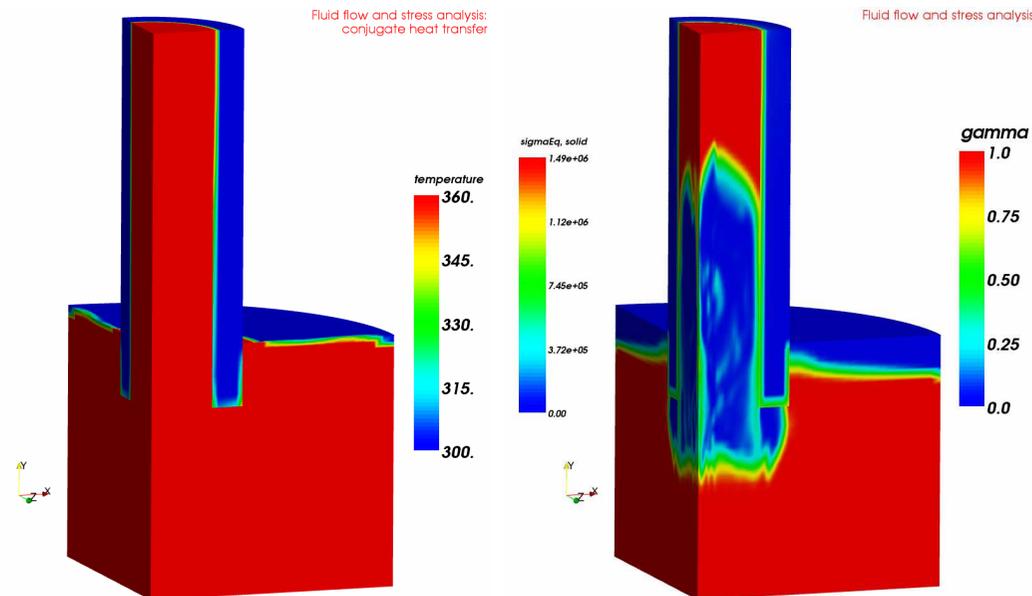
```
(  
    fvm::ddt(Tsolid) - fvm::laplacian(DTsolid, Tsolid)  
);
```

```
TEqns.solve();
```

- Coupled solver handles multiple matrices together in internal solver sweeps: arbitrary matrix-to-matrix and domain-to-domain coupling

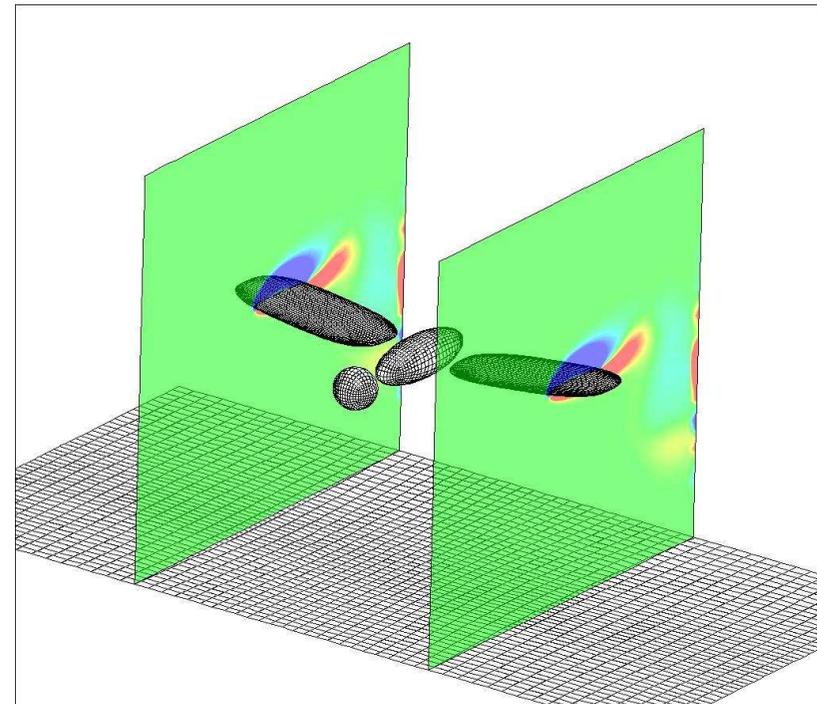
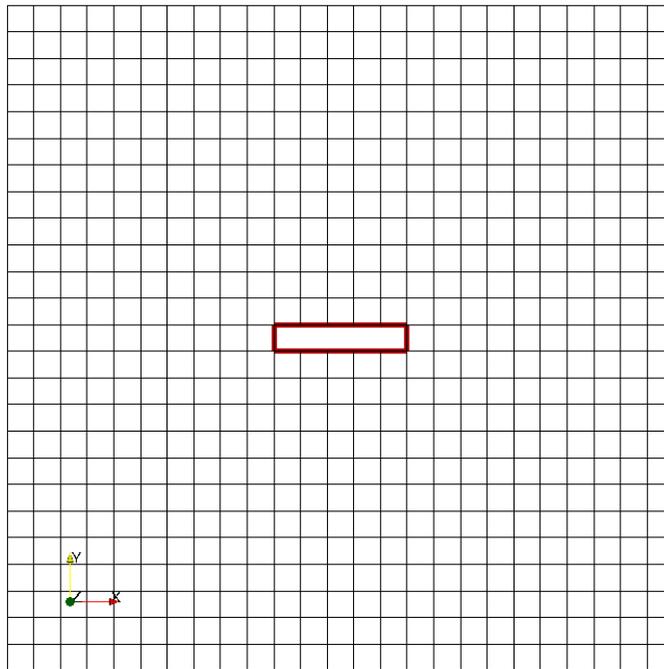
## Conjugate Heat Transfer and Thermal Shock

- Coupling may be established geometrically: adjacent surface pairs
- Each variable is stored only on a mesh where it is active: (U, p, T)
- Choice of conjugate variables is completely arbitrary: e.g. catalytic reactions
- Coupling is established only per-variable: handling a general coupled complex physics problem rather than conjugate heat transfer problem specifically
- Allows additional models to be solved on each region without overhead: structural stress analysis, turbulence or LES



## Radial Basis Function Interpolation

- General interpolation for clouds of points
- Mathematical tool which allows data interpolation from a small set of control points to space **with smoothness criteria** built into the derivation
- Used for mesh motion in cases of large deformation: no inverted faces or cells
- Implemented by Frank Bos, TU Delft and Dubravko Matijašević, FSB Zagreb



## RBF Mesh Morphing Object

- RBF morphing object defines the parametrisation of geometry (space):
  1. Control points in space, where the parametrised control motion is defined
  2. Static points in space, whose motion is blocked
  3. Range of motion at each control point:  $(\mathbf{d}_0, \mathbf{d}_1)$
  4. Set of scalar parameters  $\delta$  for control points, defining current motion as

$$\mathbf{d}(\delta) = \mathbf{d}_0 + \delta(\mathbf{d}_1 - \mathbf{d}_0), \quad \text{where } 0 \leq \delta \leq 1$$

- For each set of  $\delta$  parameters, mesh deformation is achieved by interpolating motion of control points  $\mathbf{d}$  over all vertices of the mesh: new deformed state of the geometry
- Mesh in motion remains valid since RBF satisfies smoothness criteria

## Using RBF in Optimisation

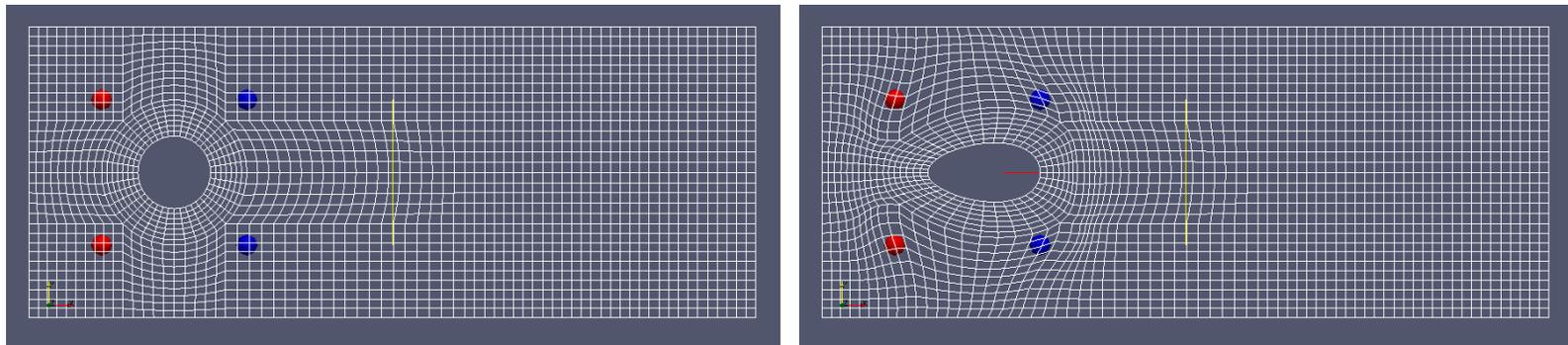
- Control points may be moved individually or share  $\delta$  values: further reduction in dimension of parametrisation of space
- Mesh morphing state is defined in terms of  $\delta$  parameters: to be controlled by the optimisation algorithm

## RBF Morphing Example: Morphing of a Cylinder

- Control points in motion:

Point	Motion range
$(-0.02 \ -0.02 \ 0)$	$((0 \ 0 \ 0) \ (-0.01 \ 0.01 \ 0))$
$( \ 0.02 \ -0.02 \ 0)$	$((0 \ 0 \ 0) \ (0.2 \ 0.0 \ 0))$
$(-0.02 \ 0.02 \ 0)$	$((0 \ 0 \ 0) \ (-0.01 \ -0.01 \ 0))$
$( \ 0.02 \ 0.02 \ 0)$	$((0 \ 0 \ 0) \ (0.02 \ 0 \ 0))$

- Note: parametrisation uses a single parameter  $\delta$  for this motion

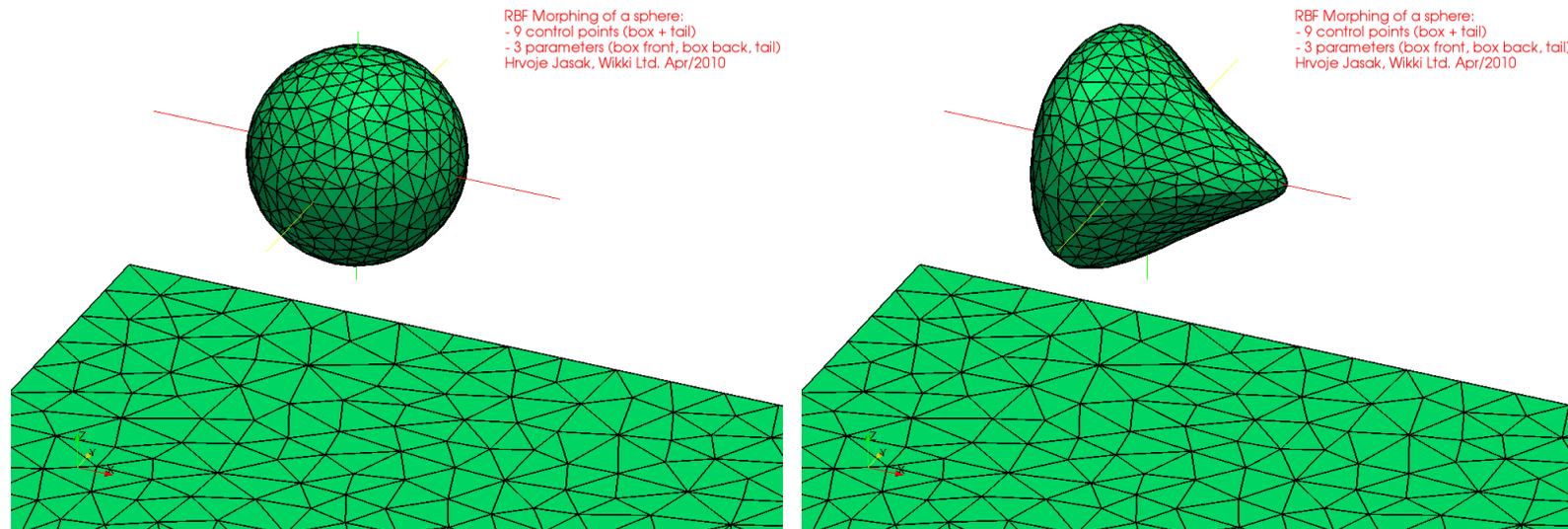


## RBF Morphing Example: Morphing of a Sphere

- Control points in motion:

Point	Motion range
$(-0.3 \pm 0.3 \pm 0.3)$	$((0 \ 0 \ 0) (\pm 0.1 \ \pm 0.1 \ \pm 0.1))$
$(0.3 \pm 0.3 \pm 0.3)$	$((0 \ 0 \ 0) (\pm 0.1 \ \pm 0.1 \ \pm 0.1))$
$(0.7 \ 0 \ 0)$	$((0 \ 0 \ 0) (0.3 \ 0 \ 0))$

- Note: parametrisation uses a single parameter  $\delta$  for this motion
- Optimisation shall be performed with 3 parameters: front, back, tail



## Geometric Shape Optimisation with Parametrised Geometry

- Specify a desired object of optimisation and use the parametrisation of geometry to explore the allowed solution space in order to find the **minimum of the optimisation objective**

$$objective = f(\text{shape})$$

### 1. Parametrisation of Geometry

- Computational geometry is complex and usually available as the computational mesh: a large amount of data
- Parametrisation tool: **RBF mesh morphing**, defining deformation at a small number of mesh-independent points in space

2. **CFD Flow Solver** is used to provide the flow solution on the current geometry, in preparation for objective evaluation
3. **Evaluation of Objective**: usually a derived property of the flow solution
4. **Optimiser Algorithm**: explores the solution space by providing sets of **shape** coordinates and receiving the value of *objective*. The search algorithm iteratively limits the space of solutions in search of a minimum value of *objective*

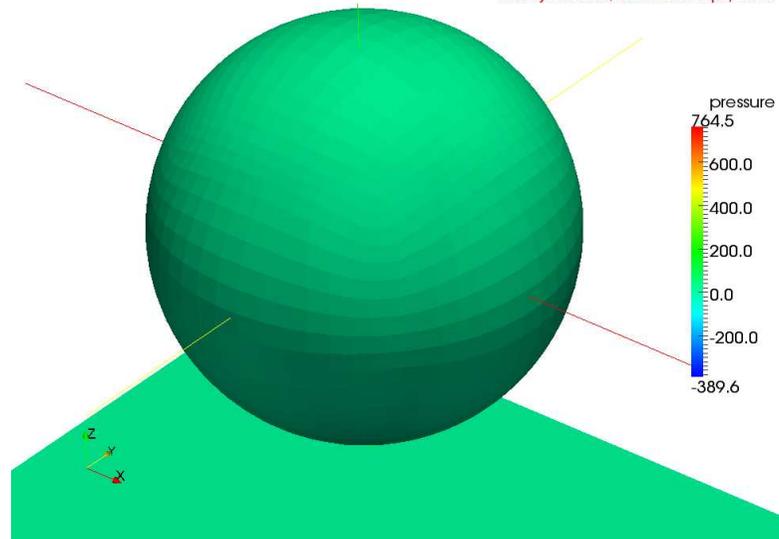
## 3-D Sphere: Minimising Drag Force

- Using 9 control points in motion, with symmetry constraints: 4 points in front square, radial motion; 4 points in back square, radial and axial motion; 1 tail point, axial motion only
- Optimisation is performed with 3 parameters:

```
iter = 1 pos = (0.2 0.7 0.2) v = 147.96 size = 0.2997  
iter = 5 pos = (0.06111 0.7092 0.7092) v = 106.26 size = 0.2153  
iter = 12 pos = (0.03727 0.9354 0.3830) v = 77.934 size = 0.0793  
iter = 22 pos = (0.04095 0.9458 0.3413) v = 75.821 size = 0.006610
```

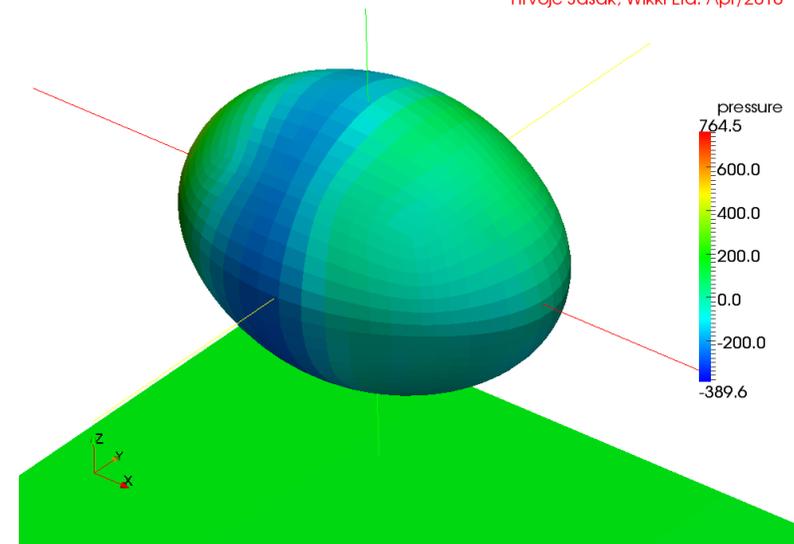
Variant: 0

Geometric shape optimisation:  
Sphere, minimizing drag force  
Hrvoje Jasak, Wikki Ltd. Apr/2010



Variant: 10043

Geometric shape optimisation:  
Sphere, minimizing drag force  
Hrvoje Jasak, Wikki Ltd. Apr/2010



## HVAC 90 deg Bend: Flow Uniformity at Outlet

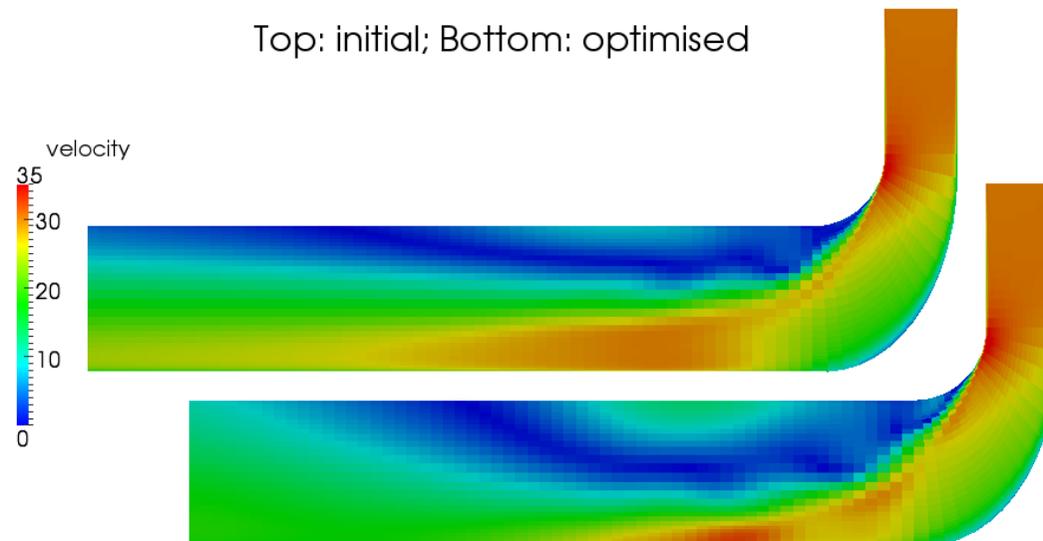
- Flow solver: incompressible steady-turbulent flow, RANS  $k - \epsilon$  model; coarse mesh: 40 000 cells; 87 evaluations of objective with CFD restart
- RBF morphing: 3 control points in motion, symmetry constraints; 34 in total
- Objective: flow uniformity at outlet plane

iter = 0 pos = (0.9 0.1 0.1) v = 22.914 size = 0.69282

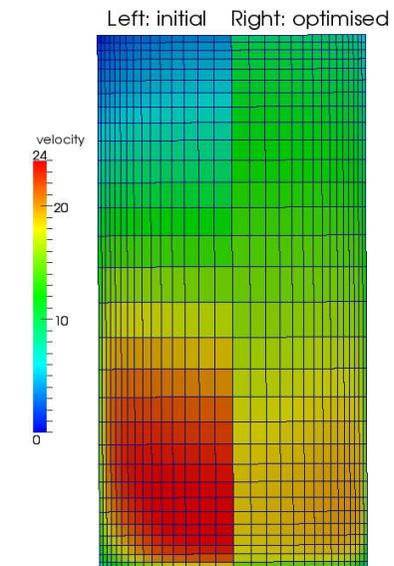
iter = 5 pos = (0.1 0.1 0.1) v = 23.0088 size = 0.584096

iter = 61 pos = ((0.990164 0.992598 0.996147) v = 13.5433 size = 0.00095

Top: initial; Bottom: optimised



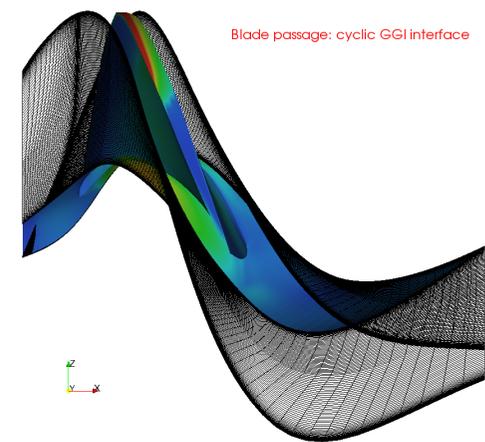
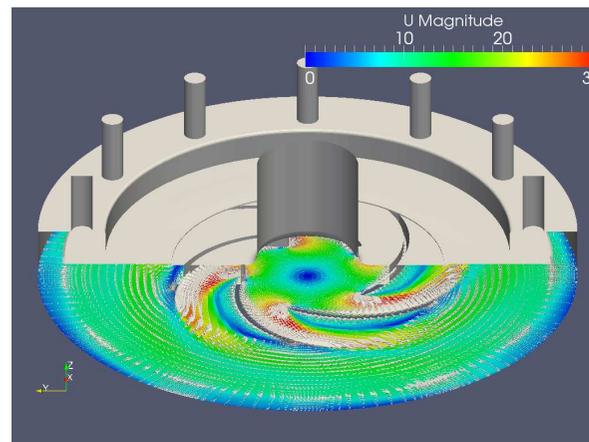
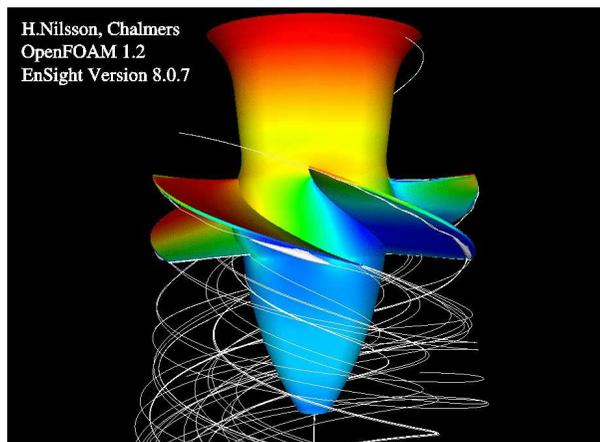
Geometric shape optimisation: flow uniformity at outlet  
RBF mesh morphing, Simplex Nelder-Mead, 3 DoF  
H. Jasak, Wikki Ltd. Oct/2010



Geometric shape optimisation: flow uniformity at outlet  
RBF mesh morphing, Simplex Nelder-Mead, 3 DoF  
H. Jasak, Wikki Ltd. Oct/2010

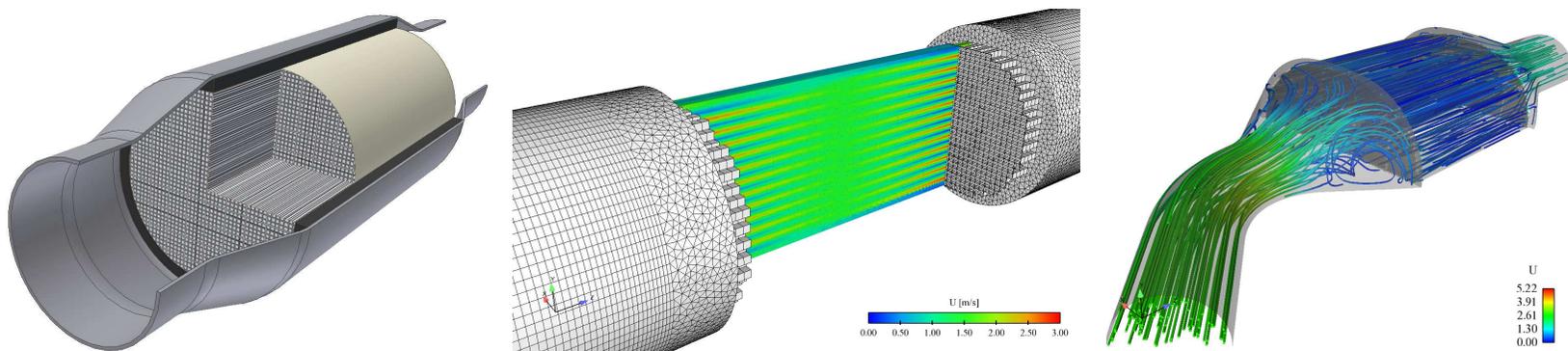
## General Grid Interface

- Turbomachinery CFD requires additional features: implemented in library form
- **General Grid Interface (GGI)** and its derived forms
  - Cyclic GGI
  - Partial overlap GGI
  - Mixing plane interface (under testing)
- Implementation and parallelisation is complete: currently running validation cases in collaboration with commercial clients and Turbomachinery Working Group
- Other turbo-related components in pipeline: harmonic balance solver solver
- Library-level implementation allows re-use of GGI beyond turbomachinery



## Modelling Diesel Particular Filters: **Federico Piscaglia, Politecnico di Milano**

- Steady-state compressible flow through thin porous layers
- Detailed 3-D meshing of channels expensive: mesh resolution requirements
- ... but due to flow non-uniformity, channel-scale simulations cannot provide the answer: not all channels are equally loaded and energy equation is solved globally
- Solution: **multi-scale filter model**
  - Each channel is one cell thick: (1-D) simulation
  - Porosity is a face property; flow friction is a volumetric sink
  - Porous faces are assigned time-dependent filtration efficiency due to soot deposition, affecting the flow and species distribution
- Automatic meshing tool for the monolith, unstructured mesh for inlet and outlet



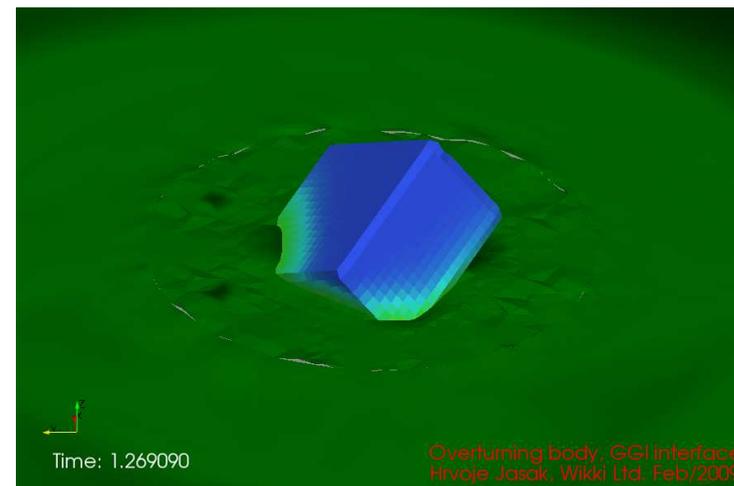
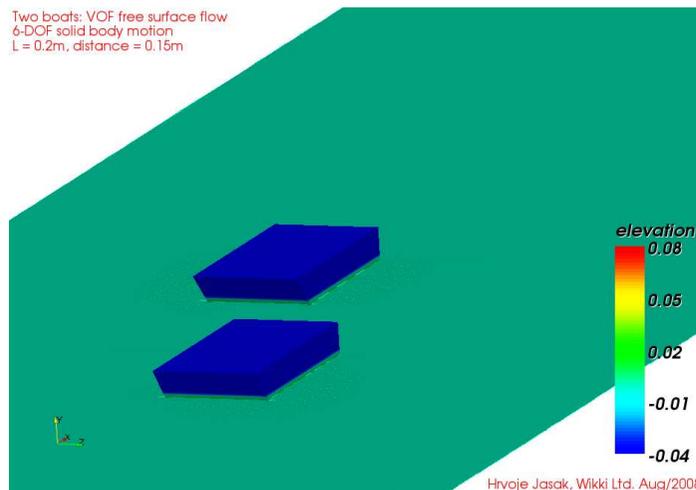
## Spray, Wall Film and Combustion Simulations in Internal Combustion Engines

- Complete simulation of spray injection, evaporation, wall film and combustion in a GDI engine. Mesh motion and topological changes as shown before
- Basic flow solver, **automatic mesh motion**, topological changes used in standard form. Simulation includes intake stroke (moving piston + valves): **reverse tumble**
- Full suite of Diesel spray modelling using Lagrangian modelling framework
- Implementation of wall film and spray-film interaction: Željko Tuković, FSB
- Mesh sensitivity of **spray penetration**: solved with adaptive refinement!
- Authors of engine simulations: Tommaso Lucchini, Gianluca D'Errico, Daniele Ettore, Politecnico di Milano and Dr. Federico Brusiani, University of Bologna



## 6-DOF Floating Body in Free Surface Flow

- **Flow solver:** turbulent VOF free surface, with moving mesh support
- **Mesh motion** depends on the forces on the hull: 6-DOF solver
- **6-DOF solver:** ODE + ODESolver energy-conserving numerics implemented using quaternions, with optional elastic/damped support
- Variable diffusivity Laplacian motion solver with 6-DOF boundary motion as the boundary condition for the mesh motion equation
- Topological changes to preserve mesh quality on capsizes
- Coupled transient solution of flow equations and 6-DOF motion, force calculation and automatic mesh motion: custom solver is built from library components



## Project Status Summary

- OpenFOAM is a free software, available to all at no charge: GNU Public License
- Object-oriented approach facilitates model implementation
- Equation mimicking opens new grounds in Computational Continuum Mechanics
- Extensive capabilities already implemented; open design for easy customisation
- Solvers are validated in detail and match the efficiency of commercial codes
- **Open Source model dramatically drops the cost of industrial CFD**

## OpenFOAM in Research and Industry

- Technical development driven by **Special Interest Groups & Birds-of-a-Feather**

Turbomachinery	Ship Hydrodynamics	Simulation of Engines
Turbulence	Fluid-Structure Interaction	Multiphase Flows
Aeroacoustics	Combustion and explosions	Solid Mechanics
Documentation	High Performance Computing	OpenFOAM in Teaching

- **Sixth OpenFOAM Workshop:** Penn State University 13-16 June 2011  
<http://www.openfoamworkshop.org>