Open Source CFD in Research and Industry

OpenFOAM with Examples

Hrvoje Jasak
h.jasak@wikki.co.uk, hrvoje.jasak@fsb.hr

Wikki Ltd, United Kingdom
FSB, University of Zagreb, Croatia

Asian Symposium on Computational Heat Transfer and Fluid Flow, 2009
Background

Objective

- Present an open source CFD simulation platform based on fundamental ideas of object orientation, equation mimicking and layered software design
- Illustrate the use of library tools on complex physics cases

Topics

1. Background: Writing an open source CFD code
2. Implementing physical models through equation mimicking
3. OpenFOAM: Object-oriented software for Computational Continuum Mechanics
4. Physical modelling capabilities
5. Some illustrative examples
   - DNS of rising bubbles with surfactant effects
   - Naval hydrodynamics examples: floating bodies in free surface flows
   - Flash-boiling flows in fuel injector nozzles
   - Soiling simulation: Eulerian-Lagrangian-liquid film coupling
   - Combustion and spray in internal combustion engines
6. Project status summary
Open Source CFD Platform

Open Source Computational Continuum Mechanics

- Commercial CFD dominates the landscape: a complete code with sufficient efficiency, parallelism, mesh handling and pre- and post- utilities is a large project
- Targeted at industrial user and established physics. Can we extend the scope?
- Complete CFD methodology is already in the public domain (research papers, model formulation, numerical schemes, linear equation solvers etc.)

**Objective:** open source implementation of existing knowledge and an object-oriented platform for easy (and collaborative) future development
  1. Completely open software platform using object-oriented design
  2. Extensive modelling capabilities in library form: component re-use
  3. Fast, robust and accurate numerical solver
  4. State-of-the-art complex geometry and dynamic mesh handling
  5. Collaborative and project-driven model development: managing, merging and distributing open source contributions
  6. Community-based and professional software support and training

- This furthers the research and collaboration by removing proprietary software issues: complete source code and algorithmic details available to all
- ...but the source code needs to be easy to understand: equation mimicking
Object-Oriented Numerics for CCM

Flexible Handling of Arbitrary Equations Sets
- Natural language of continuum mechanics: partial differential equations
- Example: turbulence kinetic energy equation

\[
\frac{\partial k}{\partial t} + \nabla \cdot (u k) - \nabla \cdot [(\nu + \nu_t) \nabla k] = \nu_t \left[ \frac{1}{2} (\nabla u + \nabla u^T) \right]^2 - \frac{\epsilon_o}{k_o} k
\]

- Objective: Represent differential equations in their natural language

```cpp
solve
(
    fvm::ddt(k)
    + fvm::div(phi, k)
    - fvm::laplacian(nu() + nut(), k)
    == nut*magSqr(symm(fvc::grad(U)))
    - fvm::Sp(epsilon/k, k)
);
```

- Correspondence between the implementation and the original equation is clear
Object Orientation

Object-Oriented Software: Create a Language Suitable for the Problem

- Analysis of numerical simulation software through object orientation:
  “Recognise main objects from the numerical modelling viewpoint”
- Objects consist of **data** they encapsulate and **functions** which operate on the data

Example: Sparse Matrix Class

- **Data members**: protected and managed
  - Sparse addressing pattern (CR format, arrow format)
  - Diagonal coefficients, off-diagonal coefficients
- Operations on matrices or data members: **Public interface**
  - Matrix algebra operations: +, −, ∗, /,
  - Matrix-vector product, transpose, triple product, under-relaxation
- Actual data layout and functionality is important only internally: efficiency

Example: Linear Equation Solver

- Operate on a system of linear equations \([A][x] = [b]\) to obtain \([x]\)
- It is irrelevant how the matrix was assembled or what shall be done with solution
- Ultimately, even the solver algorithm is not of interest: all we want is new \(x\)!
- Gauss-Seidel, AMG, direct solver: all answer to the same interface
OpenFOAM: Executive Overview

What is OpenFOAM?

- **OpenFOAM** is a free-to-use Open Source numerical simulation software with extensive CFD and multi-physics capabilities
- Free-to-use means using the software without paying for license and support, including **massively parallel computers**: free 1000-CPU CFD license!
- Software under active development, capabilities mirror those of commercial CFD
- Substantial installed user base in industry, academia and research labs
- Possibility of extension to non-traditional, complex or coupled physics: fluid-structure interaction, complex heat/mass transfer, complex chemistry, internal combustion engines, nuclear engineering, acoustics etc.

Main Components

- Discretisation: Polyhedral Finite Volume Method, second order in space and time
- Lagrangian particle tracking (discrete element model)
- Finite Area Method: 2-D FVM on curved surface in 3-D
- Automatic mesh motion (FEM), support for topological changes
- Massive parallelism in domain decomposition mode
- Physics model implementation through equation mimicking
Physical Modelling Capabilities

Physical Modelling Capability Highlights

- Basic: Laplace, potential flow, passive scalar/vector/tensor transport
- Incompressible and compressible flow: segregated pressure-based algorithms
- Heat transfer: buoyancy-driven flows, conjugate heat transfer
- Multiphase: Euler-Euler, VOF free surface capturing and surface tracking
- RANS for turbulent flows: 2-equation, RSTM; full LES capability
- Pre-mixed and Diesel combustion, spray and in-cylinder flows
- Stress analysis, fluid-structure interaction, electromagnetics, MHD, etc.
Top-Level Solver Code

Top-Level Solvers
- Libraries encapsulate interchangeable models with run-time selection
- New models provide functionality by adhering to a common interface
- Custom-written and optimised top-level solvers written for a class of physics, eg. compressible combusting LES or VOF free-surface flow
- Code clarity is paramount: existing solvers act as examples for further development or customisation

Utilities
- Pre-processing, data manipulation, mesh-to-mesh mapping etc.
- Mesh import and export, mesh generation and manipulation
- Parallel processing tools: decomposition and reconstruction
- Post processor hook-up (reader module) and data export
- A-posteriori error estimation and solution analysis

Customised Data Extraction and Analysis
- User-defined on-the-fly data extraction: function objects

This is just a “standard set”: Users write their own applications using the library
Examples of Use

OpenFOAM in Use

- If the design is well executed, code capabilities must be impressive
- A guided tour:
  1. Demonstrate implementation of a complex and non-linear physical model
  2. Illustrate solution-dependent automatic mesh motion and surface physics
  3. Show handling of highly non-linear multi-phase flow problems
  4. Illustrate coupling between multiple modelling frameworks
  5. Summarise with an engineering application on a complex geometry

Example Simulations Using OpenFOAM

- DNS of rising bubbles with surfactant effects
- Naval hydrodynamics: floating bodies in free surface flow
- Flash-boiling flows in fuel injector nozzles
- Thin liquid film model
- Soiling simulation: Eulerian-Lagrangian-liquid film coupling
- Combustion and spray in internal combustion engines
DNS of Rising Bubbles

Free Surface Tracking Solver

- Solving incompressible Navier-Stokes equations for free surface flows, deforming the mesh to accommodate surface motion
- Double boundary condition on the surface: provides surface motion
  - Fixed (atmospheric) pressure at the surface
  - No mass flux through the surface
- Remainder of the mesh moved using **automatic mesh motion solver**
  - Boundary motion acts as a boundary condition
  - Motion of internal points obtained by solving a mesh motion equation
  - FEM-based (mini-element) mesh motion solver available
  - Choice of equation: variable diffusivity Laplacian or pseudo-solid

Example: Hydrofoil Under a Free Surface
DNS of Rising Bubbles

Multi-Phase Free Surface Tracking

- Two meshes coupled on free surface: perfect capturing of the interface and curvature evaluation
- Coupling conditions on the interface include stress continuity and surface tension pressure jump

Free Rising Air Bubbles

- Simulation particularly sensitive on accurate handling of surface curvature and surface tension
- Full density and viscosity ratio
- Locally varying surface tension coefficient as a function of surfactant concentration
- Coupling to volumetric surfactant transport: boundary conditions
Complex Coupling in a Single Solver: 3-D Rising Bubble: Željko Tuković, FSB Zagreb

- FVM flow solver: incompressible $p - u$ coupling
- FEM automatic mesh motion: variable diffusivity Laplacian
- FAM for surfactant transport: convection-diffusion on surface, coupled to 3-D
- Non-inertial frame of reference, attached to bubble centroid
Free Surface Flow: Examples

Naval Hydrodynamics Examples

- Ship resistance simulation in calm sea conditions requires a “steady-state” formulation for the free surface flow
  - “Traditional” steady-state VOF solver with under-relaxation
  - Large Co-number tolerant transient solver

- Steady trim: 6-DOF force balance with mesh motion with steady-state formulation

- **Level set method**
  - Alternative to VOF surface capturing
  - Resolves problem of VOF re-sharpening
  - Necessary for **overlapping grid solver**: with SUGGAR and DirtLib libraries (Eric Paterson, Ralph Noack, Penn State)
Free Surface Flow: Examples

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 capsize
- Coupled transient solution of flow equations and 6-DOF motion, force calculation and automatic mesh motion: custom solver is built from library components
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

\[
J_a = \frac{\rho_l c_p \Delta T}{\rho_v h_{fg}}
\]

- 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 Simulations

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**
  \[
  \frac{1}{\rho} \frac{\partial \rho}{\partial p} \bigg|_{x,h} \left( \frac{\partial (\rho p^{k+1})}{\partial t} + \nabla \cdot (\rho U p^{k+1}) \right) + \rho \nabla \cdot \phi^* - \rho \nabla \frac{1}{a_p} \nabla p^{k+1}
  \]
  \[
  + M \left( p^k \right) + \frac{\partial M}{\partial p} \left( p^{k+1} - p^k \right) = 0
  \]

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.
Flash-Boiling Simulations

Conservation of Mass

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

Conservation of Momentum

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

Pressure Equation

```cpp
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.
Thin Liquid Film Model

Modelling Assumptions: 2-D Approximation of a Free Surface Flow

- Isothermal incompressible laminar flow
- Boundary layer approximation:
  - Tangential derivatives are negligible compared to normal
  - Normal velocity component is negligible compared to tangential
  - Pressure is constant across the film depth
- Similitude of the flow variables in the direction normal to the substrate
  - Prescribed cubic velocity profile

Dependent variables: $h$ and $\bar{v}$

$$\bar{v} = \frac{1}{h} \int_{0}^{h} \mathbf{v} \, dh$$

$$\mathbf{v}(\eta) = \mathbf{v}_{fs} \cdot \text{diag} (a \eta + b \eta^2 + c \eta^3)$$

$$\eta = \frac{n}{h}, \, 0 \leq n \leq h$$
Thin Liquid Film Model

Liquid Film Continuity Equation

\[
\int_{S_w} \frac{\partial h}{\partial t} \, dS + \int \mathbf{m} \cdot \mathbf{\bar{v}} \, h \, dL = \int_{S_w} \frac{\dot{m}_S}{\rho_L} \, dS
\]
Thin Liquid Film Model

Liquid Film Momentum Equation

\[
\int_{S_w} \frac{\partial h \vec{v}}{\partial t} \, dS + \oint_{\partial S_w} m \cdot (h \vec{v} \vec{v} + C) \, dL
\]

\[
= \frac{1}{\rho_L} \int_{S_w} (\tau_{fs} - \tau_w) \, dS + \int_{S_w} h g_t \, dS - \frac{1}{\rho_L} \int_{S_w} h \nabla_s p_L \, dS + \frac{1}{\rho_L} \int_{S_w} \tilde{S}_v \, dS
\]

\[
p_L = p_g + p_d + p_\sigma + p_h
\]

\[
p_{d,i} = \frac{\rho (v_{d,i})_n^2}{2}
\]

\[
p_\sigma = -\sigma \nabla_s \cdot (\nabla_s h)
\]

\[
p_h = -\rho_L n \cdot g h
\]

\[
\tilde{S}_v = \sum_i m_{d,i} (v_{d,i})_t \frac{\partial t}{dS}
\]
Thin Liquid Film Model

Validation: Droplet Spreading Under Surface Tension

- Liquid film equations governing the flow; self-similar velocity profile
- Droplet spread driven by gravity and counteracted by surface tension
- Equation set possesses an (axi-symmetric) analytical solution
Soiling Simulation

Volume-Surface-Lagrangian Coupling

- Coupling a volumetric flow model with Lagrangian particles for a dispersed phase to a thin liquid film model on the solid wall
- In terms of numerics, coupling of volumetric, surface and Lagrangian models is easy to handle: different modelling paradigms
- Main coupling challenge is to implement all components side-by-side and control their interaction: volumetric-to-particle-to-surface data exchange
- Close coupling is achieved by sub-cycling or iterations over the system for each time-step
IC Engine Modelling

Spray, Wall Film and Combustion Simulations in Internal Combustion Engines

- Complete simulation of turbulent reacting flow, spray injection, evaporation, wall film and combustion. Mesh motion and topological changes: dynamic mesh class

- Example: GDI engine with **automatic mesh motion**, topological changes used in standard form. Simulation includes intake stroke (moving piston + valves): capturing **reverse tumble** flow in the cylinder

- Full suite of Diesel spray modelling using Lagrangian modelling framework

- New: wall film model and spray-film interaction, with 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
Point of Interest

Why Consider OpenFOAM

- **Open architecture, under active development**
  - Access to complete source: no secret modelling tricks, no cutting corners
  - Both community-based and professional support available
  - Common platform for new R&D projects: shipping results of research into the hands of a customer without delay

- **Low-cost CFD solver**
  - No license or maintenance cost, including high-performance computing
  - Easily scriptable and embeddable: “automated CFD” and optimisation
  - Efficient on massively parallel computers, portable to new platforms

- **Problem-independent numerics and discretisation**
  - Tackle non-standard continuum mechanics problem, looking beyond the capabilities of commercial CFD software: customised solutions

- **Efficient environment for complex physics problems**
  - Tackling difficult physics is made easier through equation mimicking
  - Utility-level tools readily available: parallelism, dynamic mesh
  - Track record in non-linear and strongly coupled problems
  - Excellent piece of C++ and software engineering! Decent piece of CFD
Summary

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

OpenFOAM in Research and Industry

- Technical development driven by **Special Interest Groups** and **Birds-of-a-feather**

<table>
<thead>
<tr>
<th>Turbomachinery</th>
<th>Ship Hydrodynamics</th>
<th>Simulation of Engines</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turbulence</td>
<td>Fluid-Structure Interaction</td>
<td>Multiphase Flows</td>
</tr>
<tr>
<td>Aeroacoustics</td>
<td>Combustion and explosions</td>
<td>Solid Mechanics</td>
</tr>
<tr>
<td>Documentation</td>
<td>High Performance Computing</td>
<td>OpenFOAM in Teaching</td>
</tr>
</tbody>
</table>

- **Fifth OpenFOAM Workshop**: join us in Gothenburg, 21-24 June 2010:
  [http://www.openfoamworkshop.org](http://www.openfoamworkshop.org)

- Leading research/development centres worldwide: FSB Zagreb, Chalmers University, Sweden; Politecnico di Milano, University College Dublin, UMass Amherst, Penn State University, TU Munich, TU Darmstadt and others
Viscoelastic Flow Model

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

- Viscoelastic flow model:
  \[ \nabla \cdot \mathbf{u} = 0 \]

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

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

- Depending on the model, transport of \( \mathbf{\tau}_p \) is solved for: saddle-point system

- Models introduce “upper”, “lower” or Gordon-Schowalter derivatives, but we shall consider a general form: standard transport equation in relaxation form

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

  where \( \delta \) is the relaxation time-scale

- Problem: \( \mathbf{\tau}_p \) dominates the behaviour and is explicit in the momentum equation
Viscoelastic Flow Model

Model Implementation Recipe

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

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

- Calculate implied viscoelastic (tensorial) viscosity:

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

- Split complete stress into implicit and explicit component

$$\nabla \cdot \tau_p = \nabla \cdot \tau^* + \nabla \cdot \tau_{corr}$$

$$= \nabla \left( \kappa \cdot \nabla \mathbf{u} \right) + \nabla \cdot \tau_{corr}$$

- Implicit component contains (large) tensorial viscosity, based on equilibrium stress and stabilises the momentum equation and coupling

- Solution algorithm: SIMPLE p-U coupling, with FEM-like handling of stress terms
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 behaviours in each phase
- Support for topological changes: syringe ejection