Object-Oriented Software in Computational Continuum Mechanics

Hrvoje Jasak
h.jarak@wikki.co.uk

Wikki Ltd, United Kingdom
Outline

Objective

- Present a novel way of handling software implementation in computational continuum mechanics
- Demonstrate OpenFOAM in Large Eddy Simulation, free surface flows and fluid-structure interaction

Topics

- Design and limitations of modern CFD software
- A new approach to model representation
- OpenFOAM: Object-oriented software for computational continuum mechanics
- Example simulations: LES, free surface and fluid-structure interaction
- Review of the open source effort
- OpenFOAM library capabilities and pre-implemented solvers
Open Source CFD Platform

OpenFOAM: Open Source Computational Continuum Mechanics

- Commercial CFD vendors do not provide flexibility for customisation and add-on developments to answer the needs of commercial and research users
- Proprietary approach to solution methodology is the limiting factor: closed software architecture; difficulty in adding models and customising capabilities
- Complete CFD methodology is already in the public domain (research papers, model formulation, numerical schemes, linear equation solvers etc.)
- The way forward involves educated users and customised simulations
- Objective: open source implementation of existing knowledge and an object-oriented platform for easy 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 handling
  5. On-order targeted and customer-driven model development
- …but the mode of operation changes considerably
Implementing Continuum Models

How to Handle Complex Continuum Models in Software?

- Natural language of continuum mechanics: partial differential equations
- Example: turbulence kinetic energy equation

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

- Objective: represent differential equations in their natural language

```cpp
solve
{
    fvm::ddt(k) + fvm::div(phi, k)
    + fvm::laplacian(nu() + nut, k)
    - 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

- 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
  - Sparse addressing pattern (CR format, arrow format)
  - Diagonal coefficients, off-diagonal coefficients

- Operations on matrices or data members
  - 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 \( Ax = b \) 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
Basic Components

- Scalars, vectors and tensors with algebra
- Computational mesh; mesh motion, adaptive refinement, topological changes
- Fields (scalar, vector, tensor) and boundary conditions: Dirichlet, Neumann etc.
- Sparse matrix support with linear solver technology

Discretisation Classes

- Implemented as interpolation, differentiation and discretisation operators
- All discretisation methods use identical basic components, e.g. common mesh and matrix support. Better testing and more compact software implementation

Physical Modelling Libraries and Top-Level Solvers

- Libraries encapsulate interchangeable models answering to a common interfaces
- Models implement the interface functions, isolated with run-time selection
- Custom-written and optimised top-level solvers for class of physics

Utilities

- Common functionality is needed for most simulation methods
- Example: problem setup, mesh manipulation, data analysis, parallel execution
Geometry Handling

Complex Geometry Handling

- Complex geometry is a rule, not exception: need efficient support
- Polyhedral cell handling: first in class
  - A cell is a polyhedron bounded by polygons
  - Consistent handling of all cell types
  - More freedom in mesh generation
- Interfaces to all major mesh generators: polyhedral mesh encompasses all

Automatic Mesh Motion and Topological Changes

- Solution-dependent automatic mesh motion from prescribed boundary motion
- Encapsulated on-demand topological mesh changes: e.g. mesh layering
Large Eddy Simulation

LES and Aeroacoustics

- 3-D and transient simulation, incompressible Navier-Stokes model, segregated pressure-velocity solver
- Sufficient mesh resolution to capture energy-containing scales of turbulence. Efficient linear equation solvers and massive parallelism
- Second-order energy-conserving numerics in space and time
- Sub-grid scale model library: 15 models on run-time selection
- Field mapping and data analysis tools, e.g. inlet data mapping for realistic inlet turbulence; on-the-fly sampling and averaging
Aeroacoustics Post-Processing

- LES simulation captures a part of pressure fluctuations: source of aeroacoustic noise. Frequency range depends on mesh resolution.
- In order to analyse noise spectrum time-trace of pressure is stored and analysed in spectral form for selected boundaries.
- Aero-acoustic post-processing on sources of noise allow comparison with experimental data.
- Acoustic wave propagation is not modelled: only acoustic sources.
Example: Free Surface Flow

Hydrofoil Under A Free Surface
- Flow solver gives surface displacement
- Mesh adjusted to free surface position

Free-Rising Air Bubble with Surfactants
- Two meshes coupled on free surface

Single Solver, Complex Coupling
- FVM on moving meshes
- Automatic mesh motion
- FAM: Surface physics

\[ \mathbf{v}_b = -\mathbf{v}_F \]
Example: Multi-Phase Flow

Free Surface Capturing Solver

- Position of free surface determined from a phase indicator variable. Efficient handling of interface breakup and interaction
- Special algorithm based on Volume of Fluid (VOF) approach with physical compression of the interface
- Accurate handling of dominant surface tension: no parasitic velocity

Breaking Surface Simulations

Ship tank sloshing free surface + k

Splash, $u = 50m/s$, $d = 0.3mm$
Example: Flow-Induced Vibration

Fluid-Structure Interaction: **Flow-Induced Deformation**

- Traditional explicit coupling: Picard iterations. Pressure transferred from fluid to structure and displacement from structure to fluid: profile and force conservation
- Fluid: incompressible flow model, with run-time model selection
- Stress analysis: linear response with large deformation
- Automatic mesh motion solver used to deform the fluid mesh
- On solid side, mesh deformation is a part of the solution
Example: POD on LES Data

- Proper Orthogonal Decomposition (POD): analysis of flow data in terms of energy content. Based on a number of flow field snapshots, result is decomposed into dominant modes.
- In LES, this corresponds to energy analysis in different scales of turbulence: separation into large and small scales.
- Decomposition rapidly converges in energy: first 3-4 modes contain 99.9 % E.
- Extension to rapid CFD simulation: converting governing PDE’s into ordinary differential equations by projection to basis.
Example: POD on LES Data

- First POD mode captures main characteristics of the flow. Note the different nature of $u_z$ component: dominant turbulence interaction
- Higher modes capture the location and nature of turbulent interaction: concentrated close behind the step
OpenFOAM in Research and Industry

OpenFOAM Project: Leading Open Source Numerics Library

- Rapidly expanding scope of OpenFOAM makes it difficult to list its current use and capabilities: user-side development and research

OpenFOAM in Industry

- An open platform for in-house or specialist software development
- Interest greatly increased in the last year, sometimes following PhD projects, study visits, funded projects or joint development
- Active use in Audi, ABB Corporate Research, BAE Systems, Calderys SA, Esteco, Mitsubishi, Shell Oil, Toyota, Volkswagen and a number of consultancy companies

OpenFOAM in Research

- Open architecture and extensive capabilities make a good research platform
- Currently, downloads on over 200 Universities worldwide
- First OpenFOAM Workshop: Zagreb Croatia, Jan/2006: 80 attendees
- Leading research/development centres: Chalmers University, Sweden; Politecnico di Milano, Imperial College, University College Dublin, TU Freiberg, Germany
- Major development through Research Labs: National Energy Technology Lab (NETL), US Dept. of Energy (MFIX-NG), NRC Canada
Summary

OpenFOAM Consultancy and Wikki Ltd.

- Original developer of OpenFOAM with full control of its features
- Extensive CFD experience and knowledge of commercial CFD (PhD Imperial College, STAR-CD and Fluent developer – next generation solver effort)
- Providing targeted development, support and consultancy: the customer only pays for what is needed, both in support and development effort

Project Status Summary

- OpenFOAM is a free software, available to all at no charge: GNU Public License
- Wikki Ltd. provides enhanced version with additional capabilities; support and consultancy
- Object-oriented approach facilitates model implementation
- Equation mimicking opens new grounds in CCM
- Extensive capabilities already implemented
- Open design for easy user customisation
- Solvers are validated in detail and match efficiency of commercial codes
- Number of contributors is rapidly increasing
OpenFOAM: Implemented Capabilities

Discretisation Methods and Support

- Second and fourth-order Finite Volume with mesh motion and topological changes
- Polyhedral Finite Element solver (mesh motion)
- **Lagrangian particle tracking** (discrete particle model); Diesel spray model
- Finite Area Method: FVM on a curved surface in 3-D
- A-posteriori error estimation
- Dynamic mesh handling and topology changes; automatic mesh motion

Standard Top-Level Solvers

- Basic: Laplace, potential flow, transport
- Incompressible flow, compressible flow
- Heat transfer: buoyancy-driven flows
- Multiphase: Euler-Euler, surface capturing and surface tracking
- DNS and LES turbulent flows, aero-acoustics
- Pre-mixed and Diesel combustion, spray and in-cylinder flows
- Stress analysis, fluid-structure interaction, electromagnetics, MHD, etc.

This is just a “standard set”: users write their own applications using the library
OpenFOAM: Implemented Capabilities

Physical Modelling Libraries
- Thermo-physical models (liquids and gasses)
- Chemical reaction library interface (Chemkin)
- Non-Newtonian viscosity models
- Turbulence models (RANS and LES, compressible and incompressible); DNS
- Diesel spray (atomisation, dispersion, heat transfer, evaporation, spray-wall etc.)

Utilities
- Pre-processing, data manipulation
- Mesh import and export, mesh generation and manipulation
- Parallel processing tools: decomposition and reconstruction
- Post processor hook-up (reader module) and data export

High Performance Computing Support
- Massively parallel computing: domain decomposition approach
- Next-generation of linear equation solver technology: up to 3 times faster!