varRhoTurbVOF: A new set of volume of fluid solvers for turbulent isothermal multiphase flows in OpenFOAM

Wenyuan Fan, Henryk Anglart
Nuclear Engineering Division
KTH Royal Institute of Technology

Background - VOF

The volume of fluid (VOF) method.

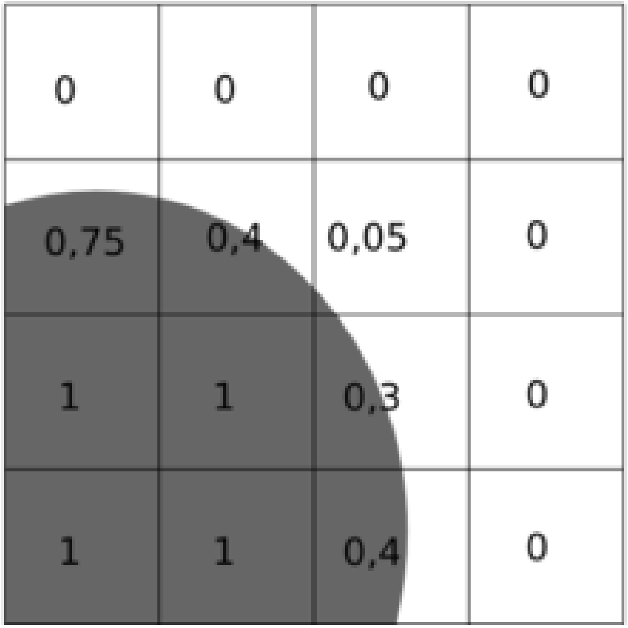
- mass conservation

$$\frac{\partial \alpha}{\partial t} + \vec{u} \cdot \nabla \alpha = 0$$

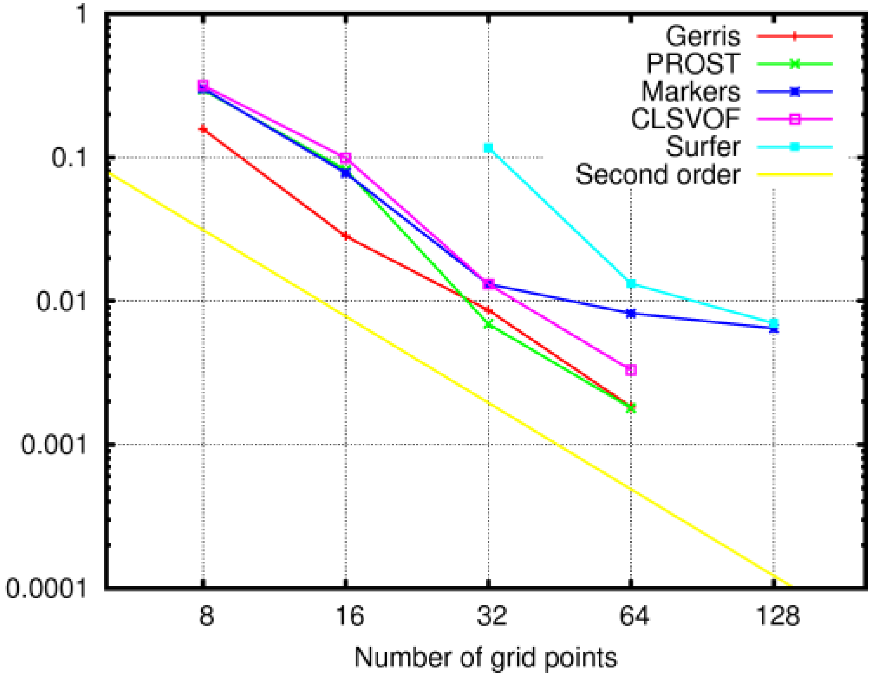
- mixture properties

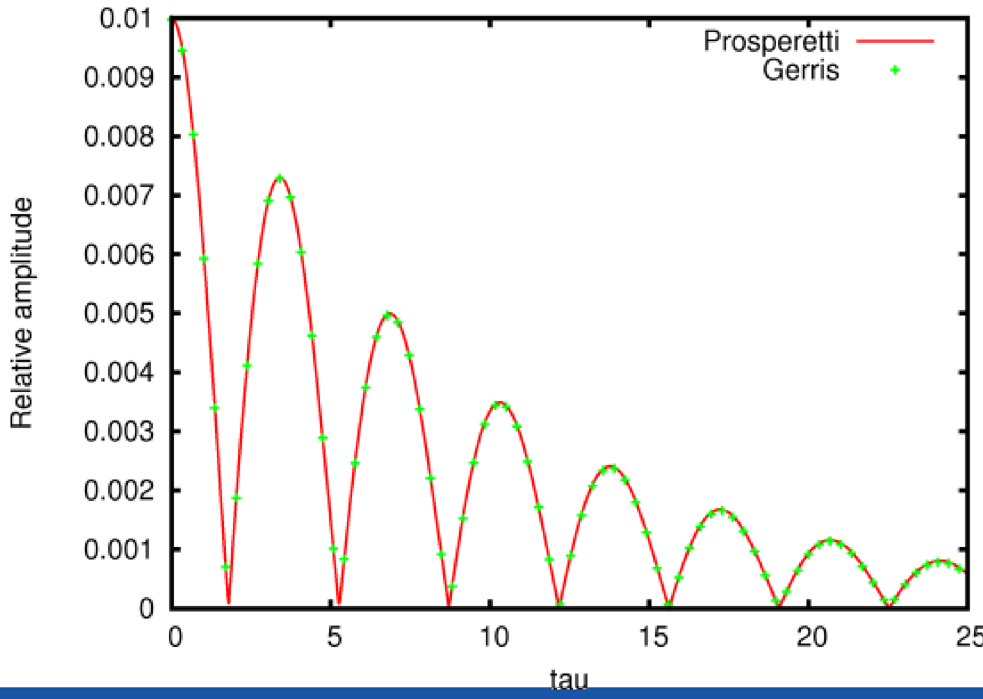
$$\rho_m = \alpha \rho_1 + (1 - \alpha) \rho_2$$
$$\mu_m = \alpha \mu_1 + (1 - \alpha) \mu_2$$

- mixture governing equations for continuity and momentum



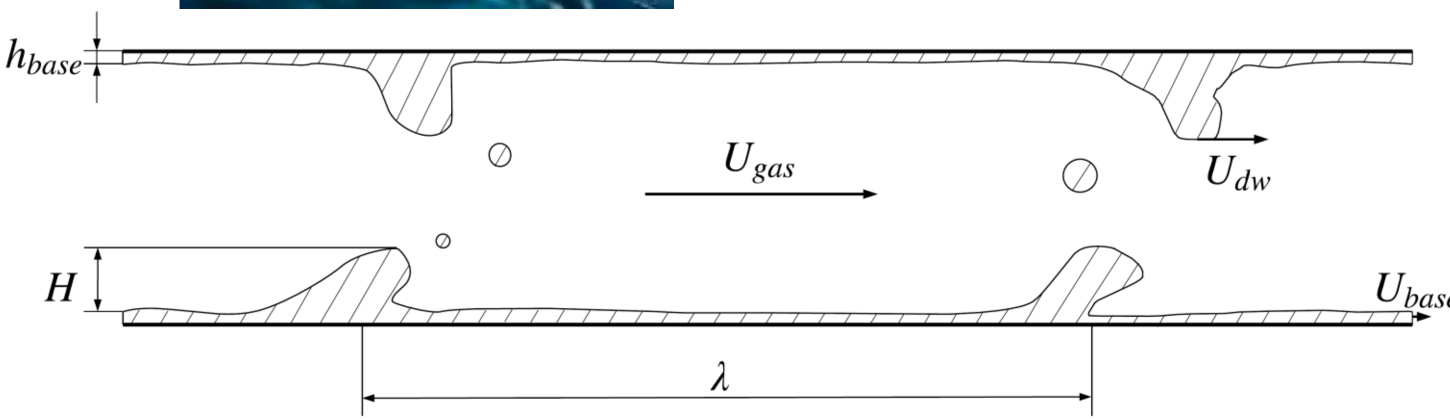
Background – achievements of VOF





http://gfs.sourceforge.net/tests/tests/capwave.html

Background – challenges of VOF



What do they have in common?

The fundamentals – incompressible flow

A flow is incompressible if it satisfies:

$$\nabla \cdot \vec{u} = 0,$$

or the equivalent form:

$$\frac{\partial \rho}{\partial t} + \vec{u} \cdot \nabla \rho = 0.$$

-constant density -> strict incompressible flow

-variable density -> variable-density incompressible flow

- incompressible VOF:

$$\rho_m = \alpha \rho_1 + (1 - \alpha) \rho_2$$

The fundamentals – turbulence modeling

The momentum equation:

$$\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot (\rho \vec{u}\vec{u}) = -\nabla p^* + \nabla \cdot \left[(\mu + \mu_t) \left(\nabla \vec{u} + (\nabla \vec{u})^T - \frac{2}{3} (\nabla \cdot \vec{u}) I \right) \right] + \vec{F}_b$$

Different treatments for μ_t to close the system:

- 0 for laminar flow
- turbulent viscosity for RANS
- subgrid-scale viscosity for LES

PDE like $$\frac{\partial \rho k}{\partial t} + \nabla \cdot (\rho \vec{u} k) = \rho P - \rho \epsilon + \nabla \cdot \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \nabla k \right]$$

Turbulence models in OpenFOAM

$$\frac{\partial \rho k}{\partial t} + \nabla \cdot (\rho \vec{u} k) = \rho P - \rho \epsilon + \nabla \cdot \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \nabla k \right]$$

```
tmp<fvScalarMatrix > kEqn
(
      fvm::ddt(alpha, rho, k_)
   +  fvm::div(alphaRhoPhi , k_)
   -  fvm::laplacian(alpha*rho*DkEff(), k_)
   ==
      alpha()*rho()*G
   -  fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
   -  fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
   +  kSource()
   +  fvOptions(alpha, rho, k_)
);
```

A class template has multiple objects:

- incompressible:
  alpha->geometricOneField,
  rho->geometricOneField
- compressible:
  alpha->geometricOneField
- phase incompressible
  rho->geometricOneField
- phase compressible

alpha is not α!

Issues with incompressible VOF solvers

Incompressible VOF solvers in OpenFOAM:

- interFoam, interIsoFoam, multiphaseInterFoam, etc.

Designed to solve variable-density incompressible flows, but use strict incompressible turbulence models:

```
autoPtr<incompressible::turbulenceModel> turbulence
(
    incompressible::turbulenceModel::New(U, phi, mixture)
);
```

Inconsistent!

Why does it matter?

Transform the variable-density k equation to

$$\left(\frac{\partial k}{\partial t} + \nabla \cdot (\vec{u}k)\right) + \frac{k}{\rho_m}\left(\frac{\partial \rho_m}{\partial t} + \vec{u} \cdot \nabla \rho_m\right) = P - \epsilon + \nabla \cdot \left[\left(\nu_m + \frac{\nu_t}{\sigma_k}\right)\nabla k\right] + \frac{\nabla \rho_m}{\rho_m} \cdot \left[\left(\nu_m + \frac{\nu_t}{\sigma_k}\right)\nabla k\right]$$

The strict incompressible k equation reads

$$\frac{\partial k}{\partial t} + \nabla \cdot (\vec{u}k) = P - \epsilon + \nabla \cdot \left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\nabla k\right]$$

Mathematically, only the diffusion term is affected.

Such diffusion terms exist in k, ϵ, ω, etc.

2 out of 30 turbulence models survive.

Possible solutions on the level of turbulence models

Add missing terms like $\frac{\nabla \rho_m}{\rho_m} \cdot \left[\left(\nu_m + \frac{\nu_t}{\sigma_k} \right) \nabla k \right]$

- stability issues if not done properly

Trick the solver

- get reference to the density field in variable-density incompressible turbulence models

Common disadvantages

- tens of equations involved
- neither user-friendly nor developer-friendly

Possible solutions on the solver level - I

One can always use compressible turbulence models for incompressible flows.

- no compressibility contribution

```
tmp<fvScalarMatrix > kEqn
(
        fvm::ddt(alpha, rho, k_)
    +   fvm::div(alphaRhoPhi , k_)
    -   fvm::laplacian(alpha*rho*DkEff(), k_)
    ==
        alpha()*rho()*G
    -   fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
    -   fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
    +   kSource()
    +   fvOptions(alpha, rho, k_)
);
```

Possible solutions on the solver level - II

Use compressible VOF solvers

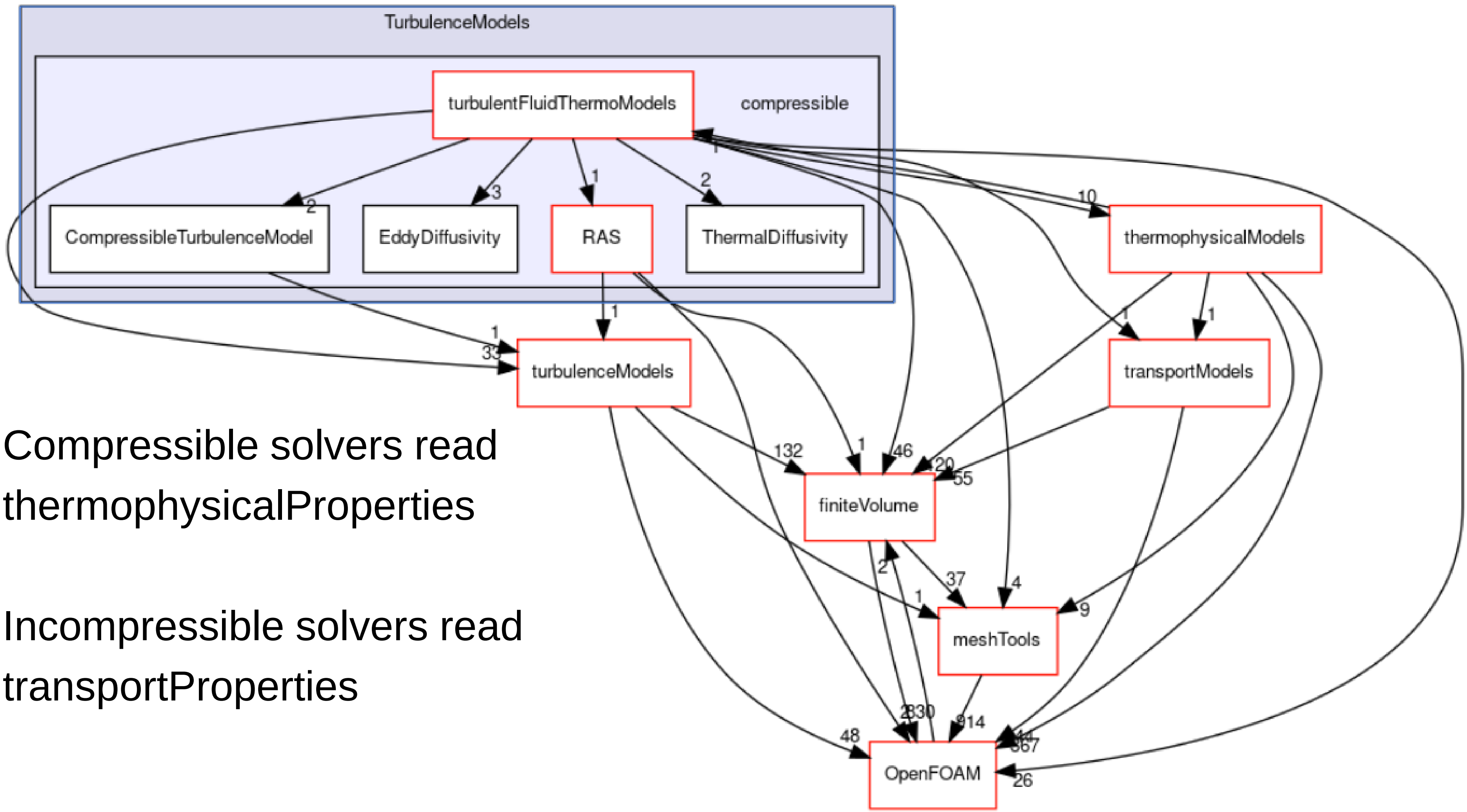
- by selecting proper equation of state
- slower, due to constructing and solving the energy equation
- not feasible for solvers without compressible versions, e.g., interMixingFoam

Construct compressible turbulence models inside incompressible VOF solvers

- not allowed in the official release

Possible solutions on the solver level - III

Compressible turbulence models must read thermal properties.



Compressible solvers read thermophysicalProperties

Incompressible solvers read transportProperties

Possible solutions on the solver level - IV

Use compressible VOF solvers

- by selecting proper equation of state
- slower, due to constructing and solving the energy equation
- not feasible for solvers without compressible versions, e.g., interMixingFoam

Construct compressible turbulence models inside incompressible VOF solvers

- not allowed in the official release
- we make it possible!

varRhoIncompressible turbulence models

$$\frac{\partial \rho k}{\partial t} + \nabla \cdot (\rho \vec{u} k) = \rho P - \rho \epsilon + \nabla \cdot \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \nabla k \right]$$

```
tmp<fvScalarMatrix > kEqn
(
      fvm::ddt(alpha, rho, k_)
   +  fvm::div(alphaRhoPhi , k_)
   -  fvm::laplacian(alpha*rho*DkEff(), k_)
   ==
      alpha()*rho()*G
   -  fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
   -  fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
   +  kSource()
   +  fvOptions(alpha, rho, k_)
);
```

A class template has multiple objects:

- incompressible: alpha->geometricOneField, rho->geometricOneField, read transportProperties
- varRhoIncompressible: alpha->geometricOneField, read transportProperties
- compressible: alpha->geometricOneField, read thermophysicalProperties
- phase incompressible rho->geometricOneField
- phase compressible

Solver modifications

Include the correct file in preprocessor directives:

- constant density `#include “turbulentTransportModel.H”`
- variable density `#include “varRhoTurbulentTransportModel.H”`

Construct the correct model:

- constant density

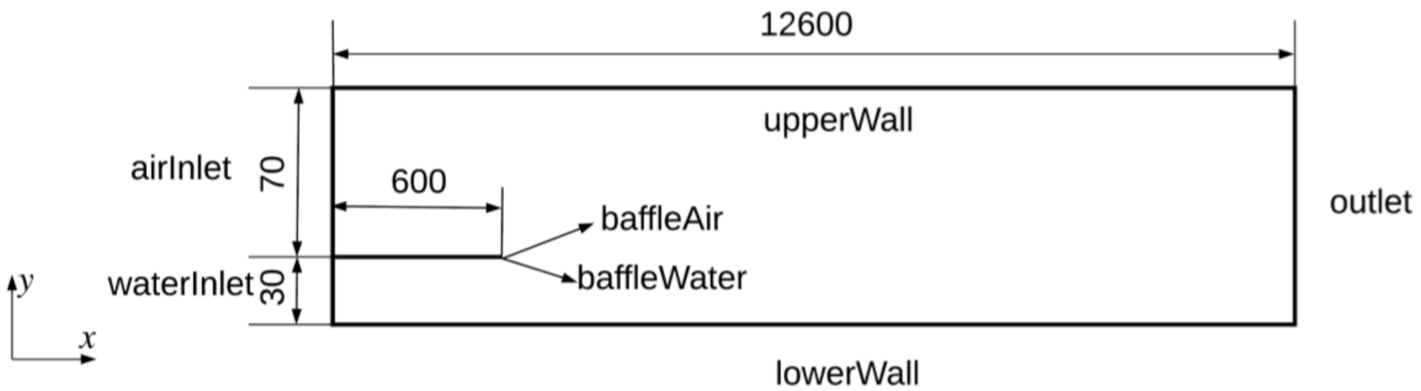
```
autoPtr<incompressible::turbulenceModel> turbulence
(
    incompressible::turbulenceModel::New(U, phi, mixture)
);
```

- variable density

```
autoPtr<incompressible::turbulenceModel> turbulence
(
    incompressible::turbulenceModel::New(rho, U, rhoPhi, phi, mixture)
);
```

Performance evaluation

Experiment on air-water stratified flow

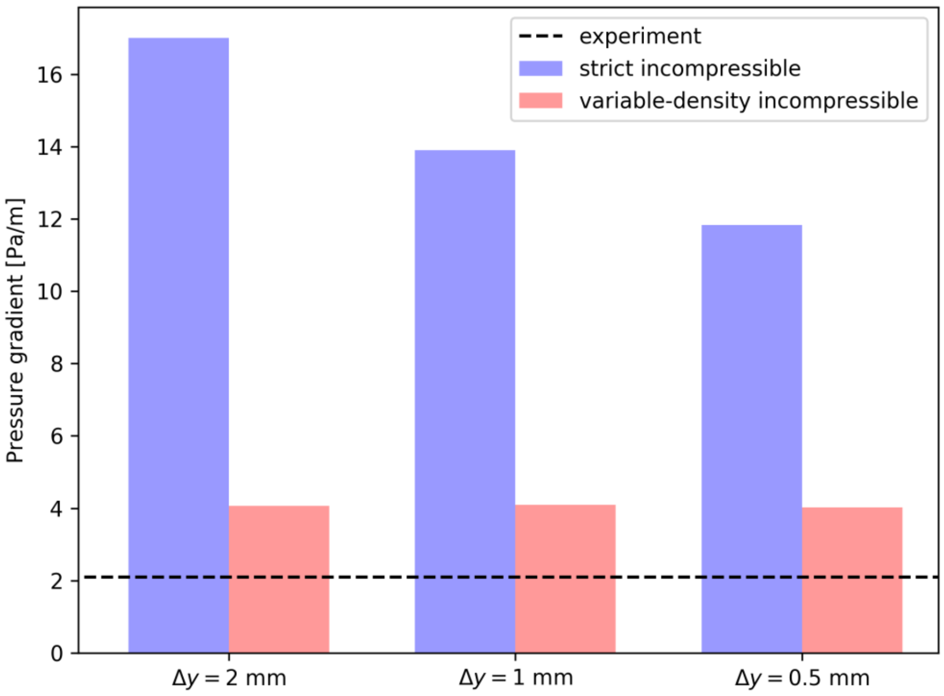


Flow configuration

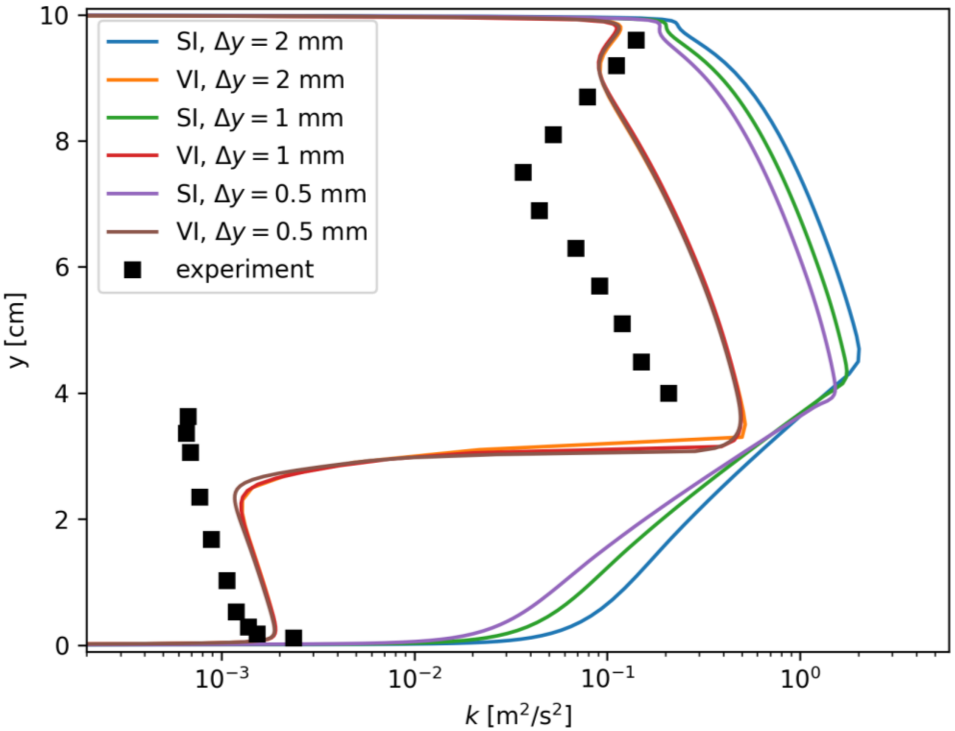
Run reference	Water flow rate [L/s]	Air flow rate [L/s]
250	3.0	45.4
400	3.0	75.4
600	3.0	118.7

Performance evaluation – Run 250

Pressure gradient

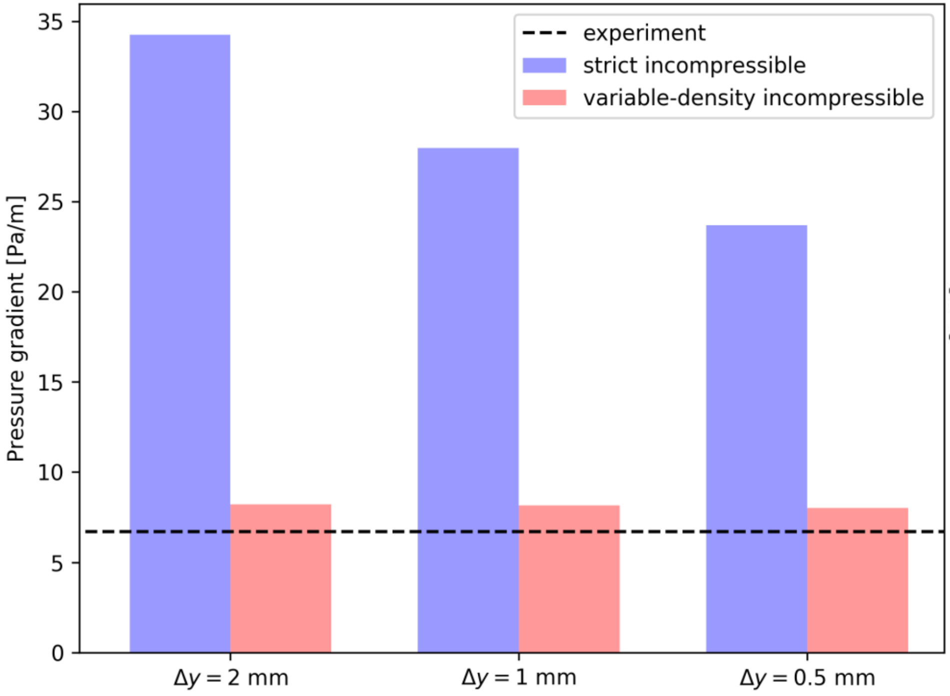


Turbulent kinetic energy

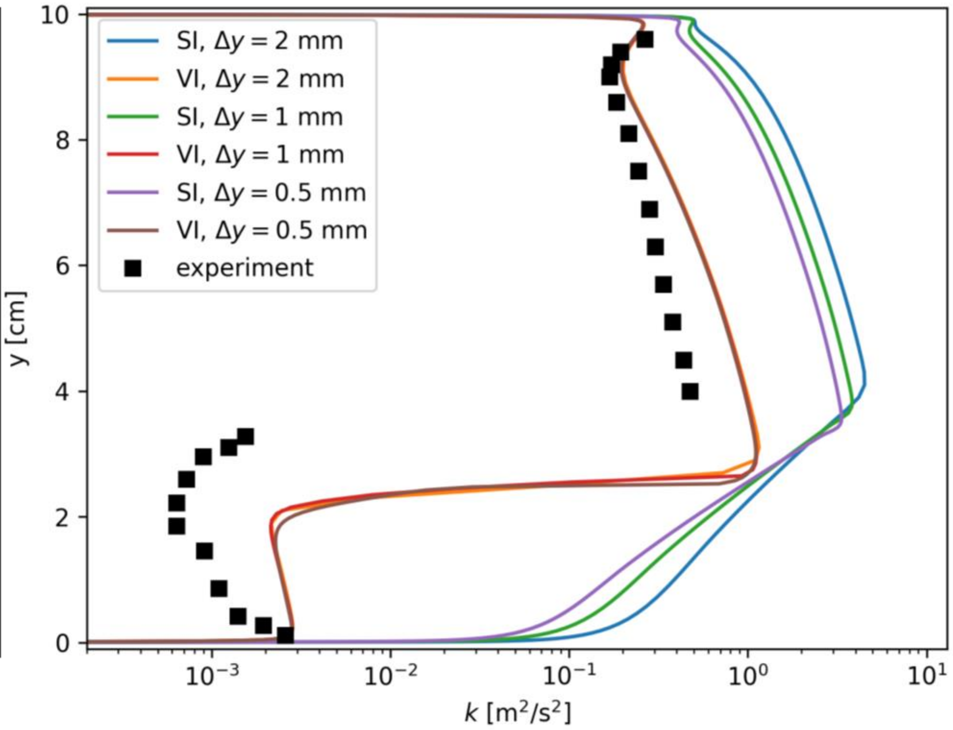


Performance evaluation – Run 400

Pressure gradient

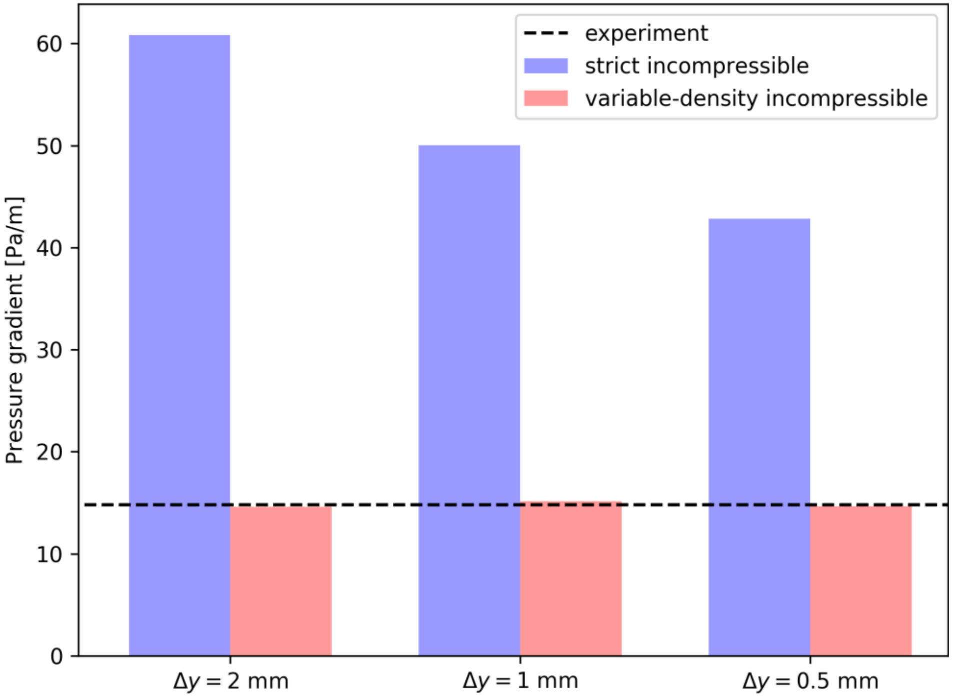


Turbulent kinetic energy

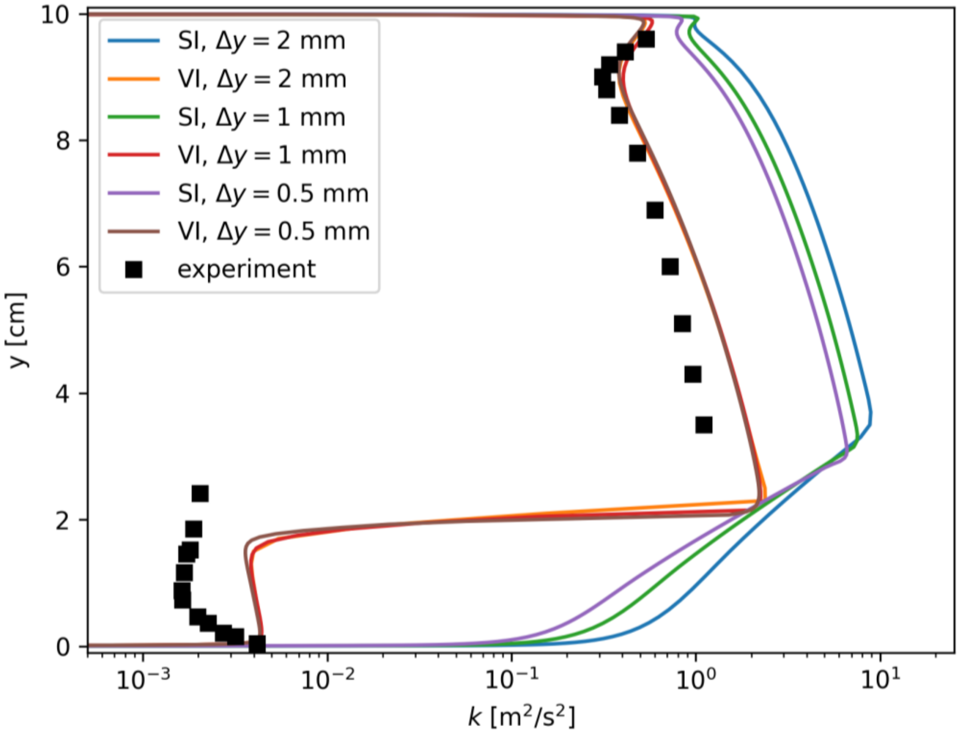


Performance evaluation – Run 600

Pressure gradient



Turbulent kinetic energy



Conclusions

A modification has been made to isothermal VOF solvers in OpenFOAM

- utilize conceptually consistent turbulence models
- minimal changes to existing codes
- better match with the experimental data
- better performance under mesh refinement

Related links

GitHub repository (additional fvOptions for turbulence damping):

https://github.com/wenyuan-fan/varRhoTurbVOF

Journal paper:

https://doi.org/10.1016/j.cpc.2019.106876

Free accepted manuscript on arXiv:

https://arxiv.org/abs/1811.12580v2

Bug report to OpenFOAM Foundation:

https://bugs.openfoam.org/view.php?id=3344

Bug report to ESI Group:

https://develop.openfoam.com/Develop...us/issues/1433