Online Training – Advanced session
February 2022

Multiphase flows modeling in OpenFOAM: Theory and applications

wolf dynamics

Copyright and disclaimer

This offering is not approved or endorsed by OpenCFD Limited, the producer of the OpenFOAM software and owner of the OPENFOAM® and OpenCFD® trademarks.

Wolf Dynamics makes no warranty, express or implied, about the completeness, accuracy, reliability, suitability, or usefulness of the information disclosed in this training material. This training material is intended to provide general information only. Any reliance the final user place on this training material is therefore strictly at his/her own risk. Under no circumstances and under no legal theory shall Wolf Dynamics be liable for any loss, damage or injury, arising directly or indirectly from the use or misuse of the information contained in this training material.

Revision 1-2022

JG

Before we begin

On the training material

- **This training is based on OpenFOAM 9.**
- In the USB key/downloaded files you will find all the training material (tutorials, slides, and lectures notes).
- You can extract the training material wherever you want. From now on, this directory will become:
    - **`$TM`**
      (abbreviation of **T**raining **M**aterial)
- To uncompress the tutorials go to the directory where you copied the training material (**`$TM`**) and then type in the terminal,
    - `$> tar -zxvf file_name.tar.gz`
- In the case directory of every single tutorial, you will find a few scripts with the extension .sh, namely, `run_all.sh`, `run_mesh.sh`, `run_sampling.sh`, `run_solver.sh`, and so on.
- These scripts can be used to run the case automatically by typing in the terminal, for example,
    - `$> sh run_all.sh`
- These scripts are human-readable, and we highly recommend you open them, get familiar with the steps, and type the commands in the terminal. In this way, you will get used with the command line interface and OpenFOAM commands.
- If you are already comfortable with OpenFOAM, run the cases automatically using these scripts.
- In the case directory, you will also find the `README.FIRST` file. In this file, you will find some additional comments.

Conventions used

The following typographical conventions are used in this training material

- Text in `Courier new` font indicates Linux commands that should be typed literally by the user in the terminal.
- Text in **`Courier new bold`** font indicates directories.
- Text in *`Courier new italic`* font indicates human readable files or ascii files.
- Text in **Arial bold font** indicates program elements such as variables, function names, classes, statements and so on. It also indicates environment variables, and keywords. They also highlight important information.
- Text in Arial underline in blue font indicates URLs and email addresses.
- This icon indicates a warning or a caution.
- This icon indicates a tip, suggestion, or a general note.
- This icon indicates a folder or directory.
- This icon indicates a human readable file (ascii file).
- This icon indicates that the figure is an animation (animated gif).
- These characters `$>` indicate that a Linux command should be typed literally by the user in the terminal.

Conventions used

The following typographical conventions are used in this training material

- Large code listing, ascii files listing, and screen outputs can be written in a square box, as follows:

```
#include <iostream>
using namespace std;

// main() is where program execution begins.  It is the main function.
// Every program in c++ must have this main function declared

int main ()
{
        cout << "Hello world";                  //prints Hello world
        return 0;                               //returns nothing
}
```

- To improve readability, the text might be colored.
- The font can be `Courier new` or **Arial bold**.
- And when required, the line number will be shown.

Roadmap

1. Introduction to multiphase flows
2. Modeling approaches for multiphase flows
3. Governing equations and interfacial momentum transfer models
4. Multiphase solvers in OpenFOAM
5. Selecting physical properties, phase interaction, and advanced models
6. Final remarks – Tips and tricks
7. Additional tutorials

Roadmap

1. **Introduction to multiphase flows**
2. Modeling approaches for multiphase flows
3. Governing equations and interfacial momentum transfer models
4. Multiphase solvers in OpenFOAM
5. Selecting physical properties, phase interaction, and advanced models
6. Final remarks – Tips and tricks
7. Additional tutorials

“Essentially, all models are wrong,
but some are useful”

G. E. P. Box

George Edward Pelham Box
18 October 1919 – 28 March 2013. Statistician, who worked in the areas of quality control, time-series analysis, design of experiments, and Bayesian inference. He has been called *“one of the great statistical minds of the 20th century”*.

Introduction to multiphase flows

What is a multiphase flow?

- A multiphase flow is a fluid flow consisting of more than one phase component and has some level of phase separation above the molecular level.
- Multiphase flows exist in many different forms. Two phase flows can be classified according to the state of the different phases:
    - Gas-Liquid mixture.
    - Gas-Solid mixture.
    - Liquid-Solid mixture.
    - Immiscible liquid-liquid.




Multiphase vs. Multispecies

- Multiphase flows have some level of phase separation at a scale well above molecular level.
- Multispecies flows have a mixing on the molecular level, there is no interphase that indicates a separation at macroscopic level.

Water + oil = multiphase
Clear and distinct interface

Water + tea = multispecies
Molecular mixing, no interface

Multiphase flows in industry and nature based on the state of the different phases

- Multiphase flows are very common in industry and in nature, the following are a few examples depending on the state of the different phase.

- **Gas–particle flows:**
    - Natural: sandstorms, volcanoes, avalanches, rain droplets, mist formation.
    - Biological: aerosols, dust particles, smoke (finely soot particles),
    - Industrial: pneumatic conveyers, dust collectors, fluidized beds, solid propellant rockets, pulverized solid particles, spray drying, spray casting.

- **Liquid–solid flows:**
    - Natural: sediment transport of sand in rivers and sea, soil erosion, mud slides, debris flows, iceberg formation.
    - Biological: blood flow, eyes, lungs.
    - Industrial: slurry transportation, flotation, fluidized beds, water jet cutting, sewage treatment plants, bio-reactors.

Multiphase flows in industry and nature based on the state of the different phases

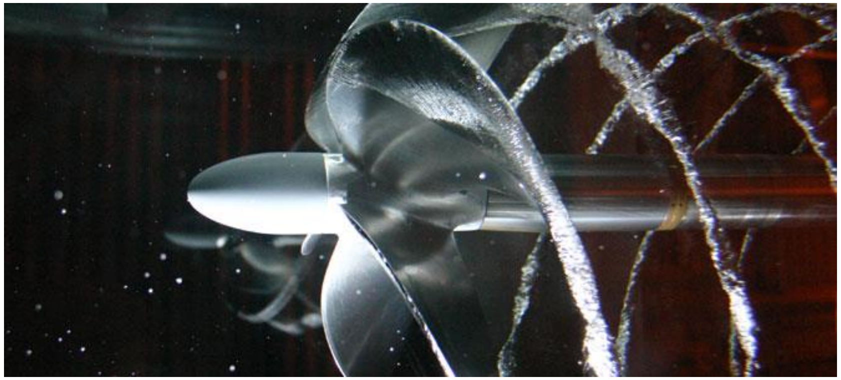
- **Gas–liquid flows:**
  - Natural: ocean waves.
  - Biological: blood flow, eyes, lungs.
  - Industrial: boiling water and pressurized water nuclear reactors, chemical reactor, desalination systems, sewage treatment plants, boilers, heat exchangers, internal combustion engines, liquid propellant rockets, fire sprinkler suppression systems.
- **Liquid–liquid flows:**
  - Industrial: emulsifiers, fuel-cell systems, micro-channel applications, extraction systems.
- **Gas–liquid–solid flows:**
  - Industrial: air lift pumps, fluidized beds, oil transportation.

Examples of multiphase flows – Transportation

Submarine wake
http://commons.wikimedia.org/wiki/File:HMAS_Rankin_2007.jpg#/media/

Cargo ship wake
http://developeconomies.com/development-economics/how-to-get-america-back-on-track-free-trade-edition/

High speed boat wake and spray
http://gizmodo.com/5830571/new-technology-tricks-water-into-thinking-your-ship-isnt-there



Propeller cavitation
http://www.veempropellers.com/features/cavitationresistance

Introduction to multiphase flows

Examples of multiphase flows – Flows in nature

Light rain and snowfall
http://www.esa.int/

Volcano eruption
http://americanpreppersnetwork.com/2014/08/preparing-volcano-eruption.html

Landslide, mudslide, mudflow, debris transport
http://www.britannica.com/EBchecked/topic/395994/mudflow

Sand storm
http://www.vizrt.com/news/newsgrid/35339/Sand_storm_potential_forecasts_with_StormGeo__Viz_Weather

Examples of multiphase flows – Flows in nature

Siltation & Sedimentation
http://blackwarriorriver.org/siltation-sedimentation/

Debris transport
https://walrus.wr.usgs.gov/elwha/river.html

Point source pollution – Shipyard
http://commons.wikimedia.org/wiki/File:Jacuecanga_Angra_dos_Reis_Rio_de_Janeiro_Brazil_Brasfels.JPG#/media/File:Jacuecanga_Angra_dos_Reis_Rio_de_Janeiro_Brazil_Brasfels.JPG

Coastal structures – Waves interaction.
http://californiabeachblog.blogspot.it/2013_10_01_archive.html

Examples of multiphase flows – Water resources management

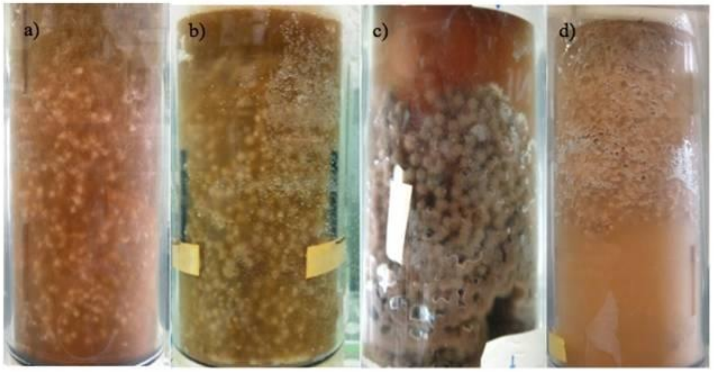
Municipal and industrial water treatment
http://www.asiapacific.basf.com/apex/AP/en/upload/Press2010/BASF-Water-Chem-2010-Paper-Chem-2010-Intex-Shanghai

Desalinization plant
http://www.logicom.net/EN/WaterPowerGas/Pages/Desalination.aspx

Dam spillway
http://www.abc.net.au/news/2011-10-11/jindabyne-dam-spillway-gates-and-cone-valves-open-rotate840x8/3497340

Slurry flow
A suspension of solid particles in a liquid, as in a mixture of cement, clay, coal dust, manure, meat, etc. - with water is often called a slurry.
https://aarondembskibowden.wordpress.com/tag/slurry/

Examples of multiphase flows – Biofuels and bioenergy

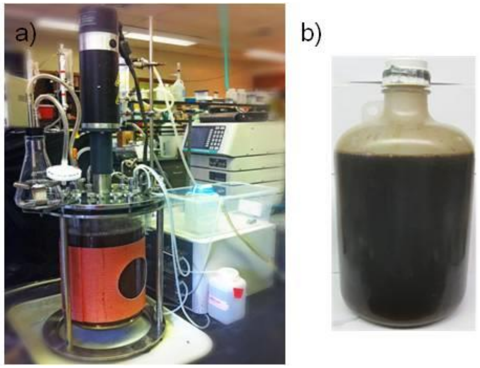


Changes in fungal morphologies during fungal fermentation in airlift reactor at 1.5 vvm: 16 h (a); 24 h (b); 48 h (c); and 72 h (d)

Photo credit: http://www2.hawaii.edu/~khanal/fungal

An airlift bioreactor system

Photo credit: http://www2.hawaii.edu/~khanal/fungal



Ethanol fermentation for vinasse preparation (a) and vinasse (b)

Photo credit: http://www2.hawaii.edu/~khanal/fungal

Moss bioreactor– University Freiburg

Photo credit: http://commons.wikimedia.org/wiki/File:Bioreaktor_quer2.jpg#/mediaFile:Bioreaktor_quer2.jpg

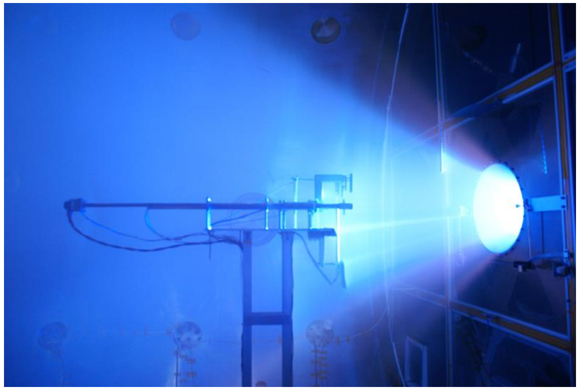
Introduction to multiphase flows

Examples of multiphase flows – Energy generation

Cooling Towers
https://whatiswatertreatment.wordpress.com/what-are-the-systems-associated-with-water-treatment-and-how-are-they-treated/103-2/

Fuel cells
http://www.h2fc-fair.com/hm12/highlights.html

A bus fueled by biodiesel
http://commons.wikimedia.org/wiki/File:Soybeanbus.jpg#/media/File:Soybeanbus.jpg



Plasma engine
http://en.wikipedia.org/wiki/File:VX-200_operation_full_power.jpg#/media/File:VX-200_operation_full_power.jpg

Examples of multiphase flows – Food processing

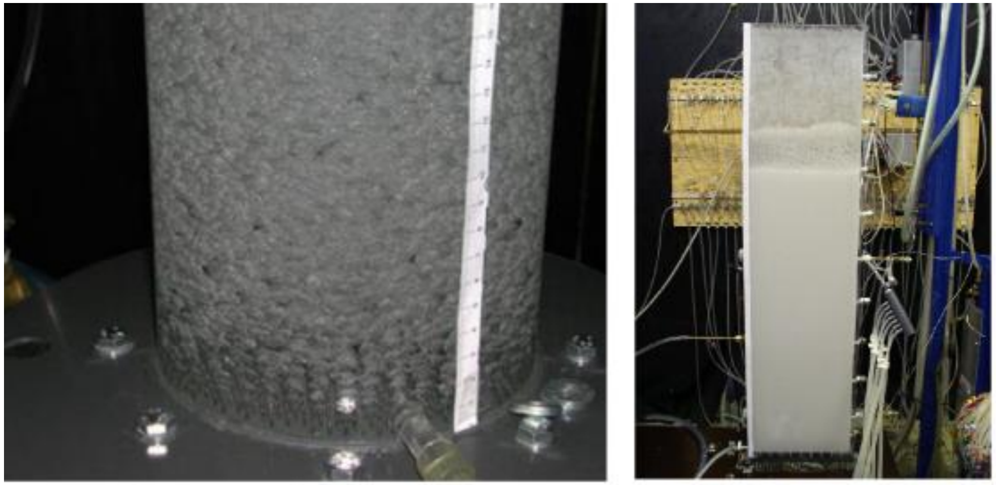
Grain/rice/Wheat/Seed hopper – Controlled atmosphere storage using carbon dioxide
http://www.co2meter.com/blogs/news/6077164-controlled-atmosphere-storage-using-carbon-dioxide

Fermentation of beer and spirits
http://www.distillingliquor.com/2015/02/05/how-to-make-alcohol-and-spirits/

Machine used for drying the coffee beans
https://www.travelblog.org/Photos/4865694

Visco-elastic fluids – Weissenberg effect. Dough tend to climb up rotating shafts
https://raweb.inria.fr/rapportsactivite/RA2010/concha/uid20.html

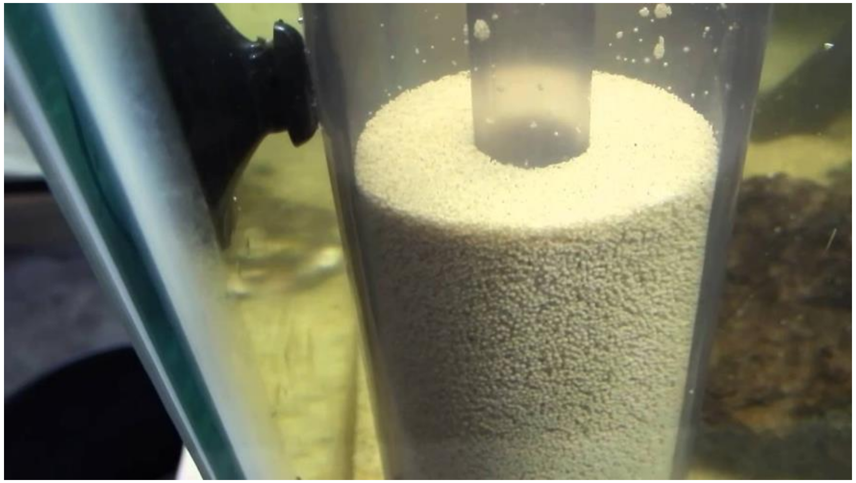
Examples of multiphase flows – Bubble columns and fluidized beds



3-D bubble column　　2-D column

Bubble columns

Photo credit: http://www.cheme.nl/tp/people/hooshyar.shtml



Fluidized bed filter

Photo credit: https://www.youtube.com/watch?v=fPnr4ZsoJwE

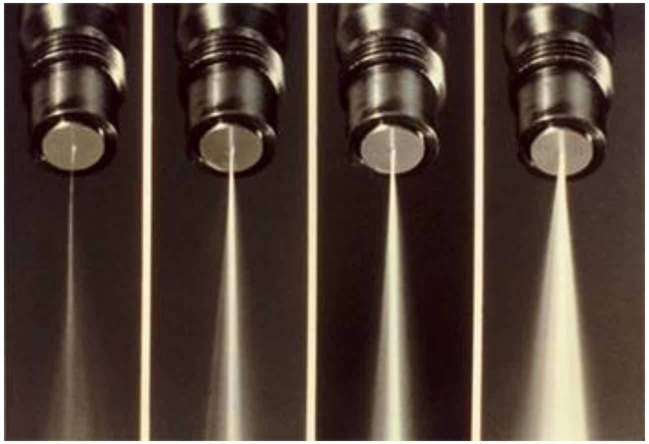
Fluidized bed drying

Photo credit: https://www.glatt.com/en/processes/fluidized-bed-drying/

Introduction to multiphase flows

Examples of multiphase flows – Additional applications

Molten iron
http://www.castingsolutions.com/production



Nozzle spray
http://www.cdxetextbook.com/asearticles/quietdieselevol.html

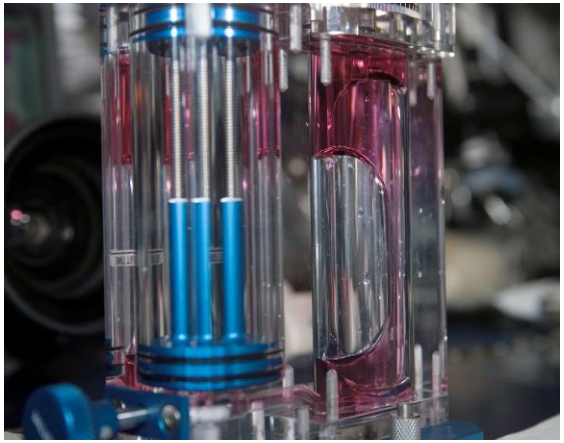
Chemical reactor for the pharmaceutical and biotechnology industry
http://www.total-mechanical.com/Industrial/CaseStudies.aspx

Pipelines
http://www.downstreamtoday.com/%28S%282cugvw45vzs5a04541d1e4jy%29%29/news/article.aspx?a_id=18931&AspxAutoDetectCookieSupport=1

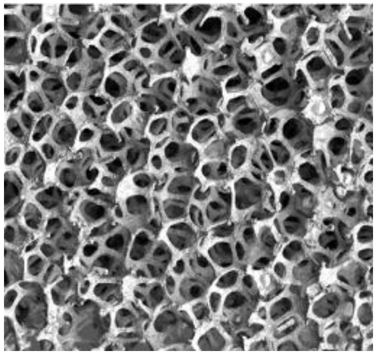
Introduction to multiphase flows

Examples of multiphase flows – Additional applications

Hydrophobic and Oleophobic coatings textiles
http://www.aculon.com/industrial.php



Capillary flows
http://commons.wikimedia.org/wiki/File:Capillary_Flow_Experiment.jpg#/media/File:Capillary_Flow_Experiment.jpg



Porous media for air filter and noise reduction
https://www.newcastle.edu.au/research-and-innovation/centre/cgmm/research/thermal-transport-in-composites-and-porous-media

Oil well drilling and water injection
https://www.osha.gov/SLTC/etools/oilandgas/servicing/special_services.html

- **As you can see, multiphase flows are present in many industrial processes and natural systems.**
- **Hence the importance of understanding, modeling, and simulating multiphase flows.**

Introduction to multiphase flows

Classifying multiphase flow according to phase morphology

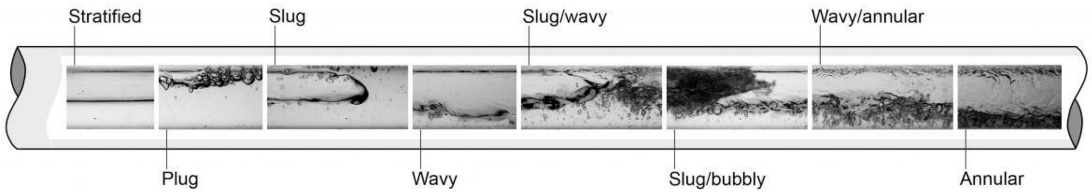
- **Disperse phase**: the phase is dispersed as non-contiguous isolated regions within the other phase (the continuous phase) . When we work with a disperse phase we say that the system is dispersed: disperse-continuous flow.
- **Continuous phase**: the phase is contiguous throughout the domain and there is one well defined interphase with the other phase. When we work with continuous phases we say that the system is separated: continuous-continuous flow.

Dispersed system

Separated system

Multiphase flow regimes

- Flow maps and flow patterns in horizontal pipes



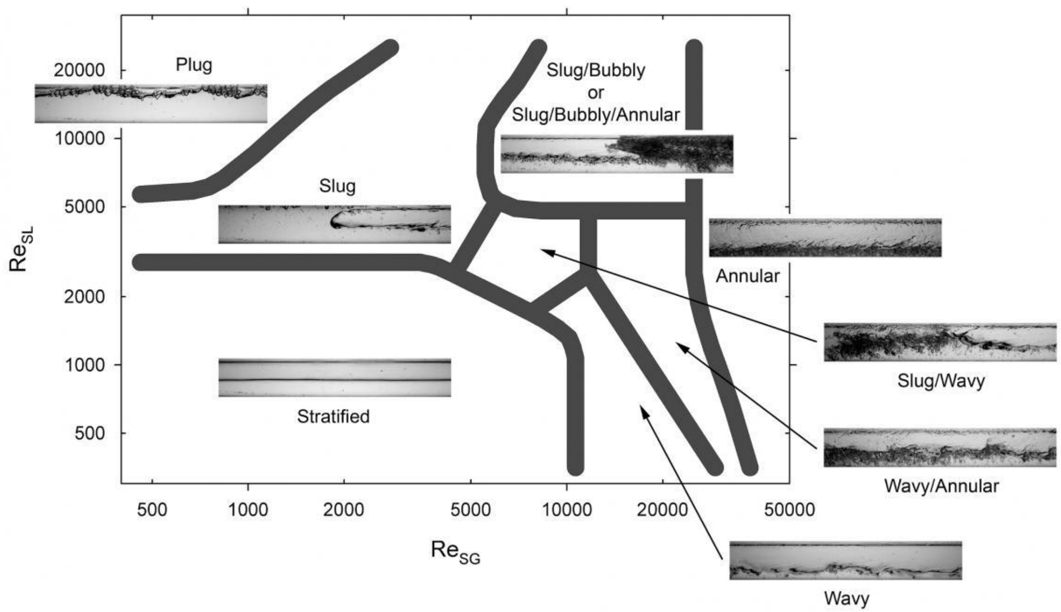
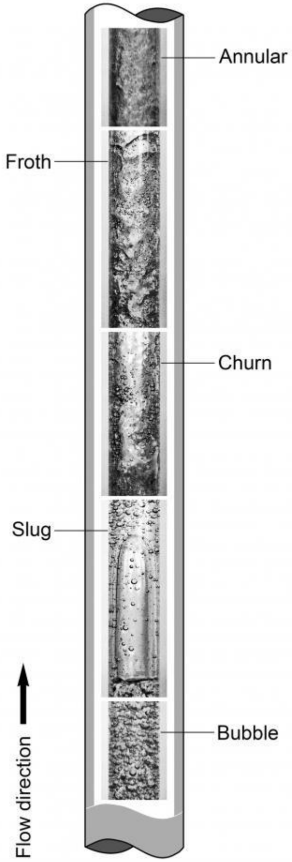


Photo credit: J. Ghajar. Oklahoma State University. Single/Two-phase Heat Transfer Laboratory. http://aghajar.okstate.edu/

Multiphase flow regimes

- Flow maps and flow patterns in vertical pipes (upward flow)



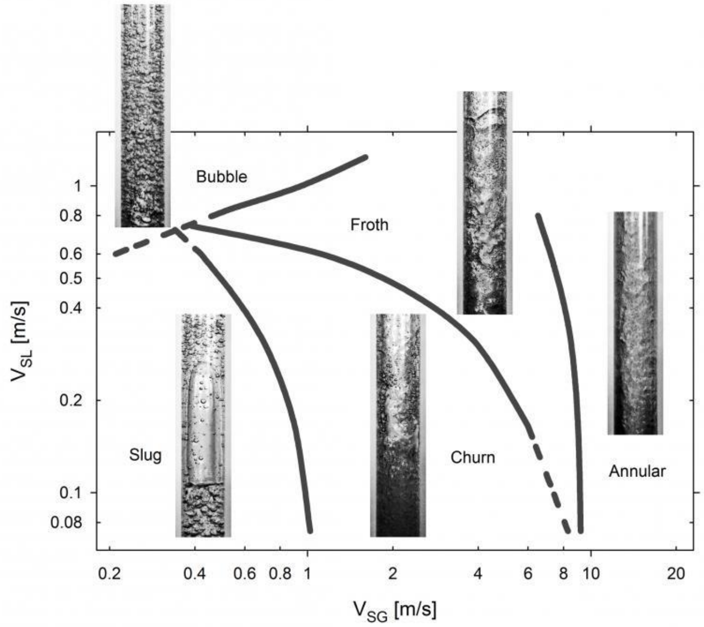


Photo credit: J. Ghajar. Oklahoma State University. Single/Two-phase Heat Transfer Laboratory. http://aghajar.okstate.edu/

Multiphase flow regimes

- Flow maps and flow patterns in vertical pipes (downward flow)

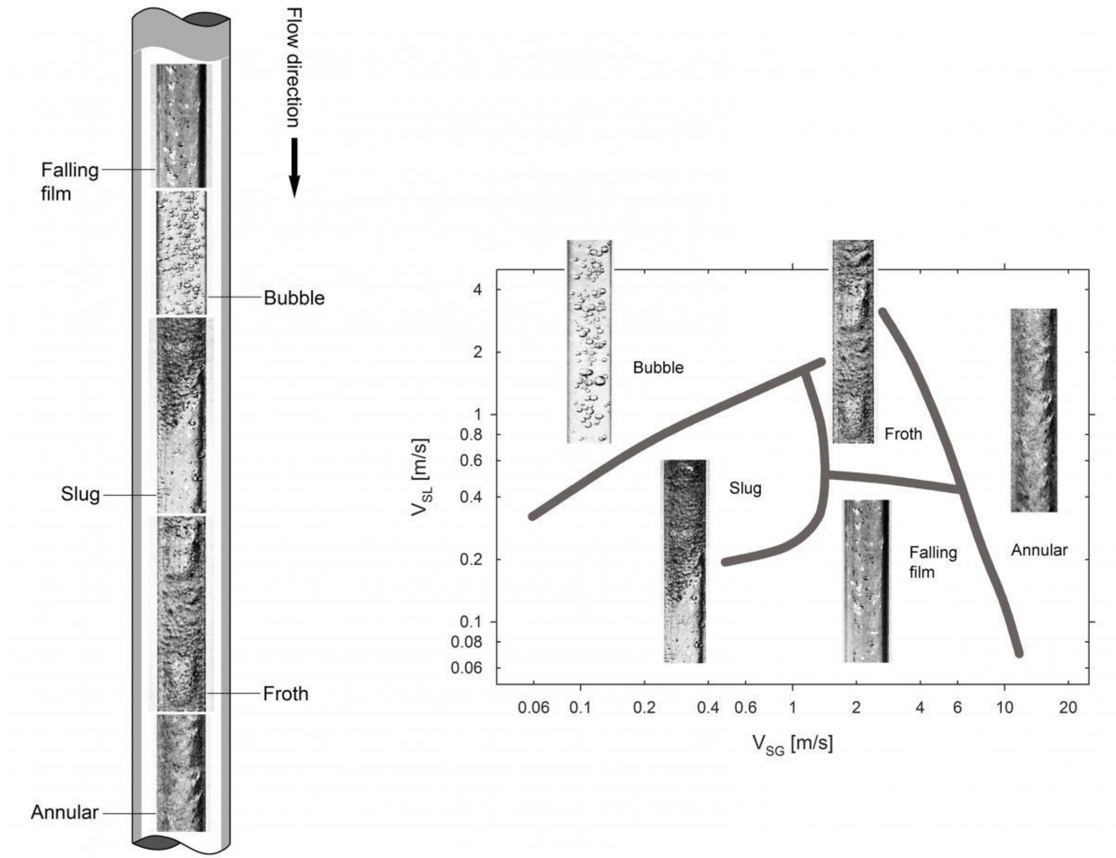
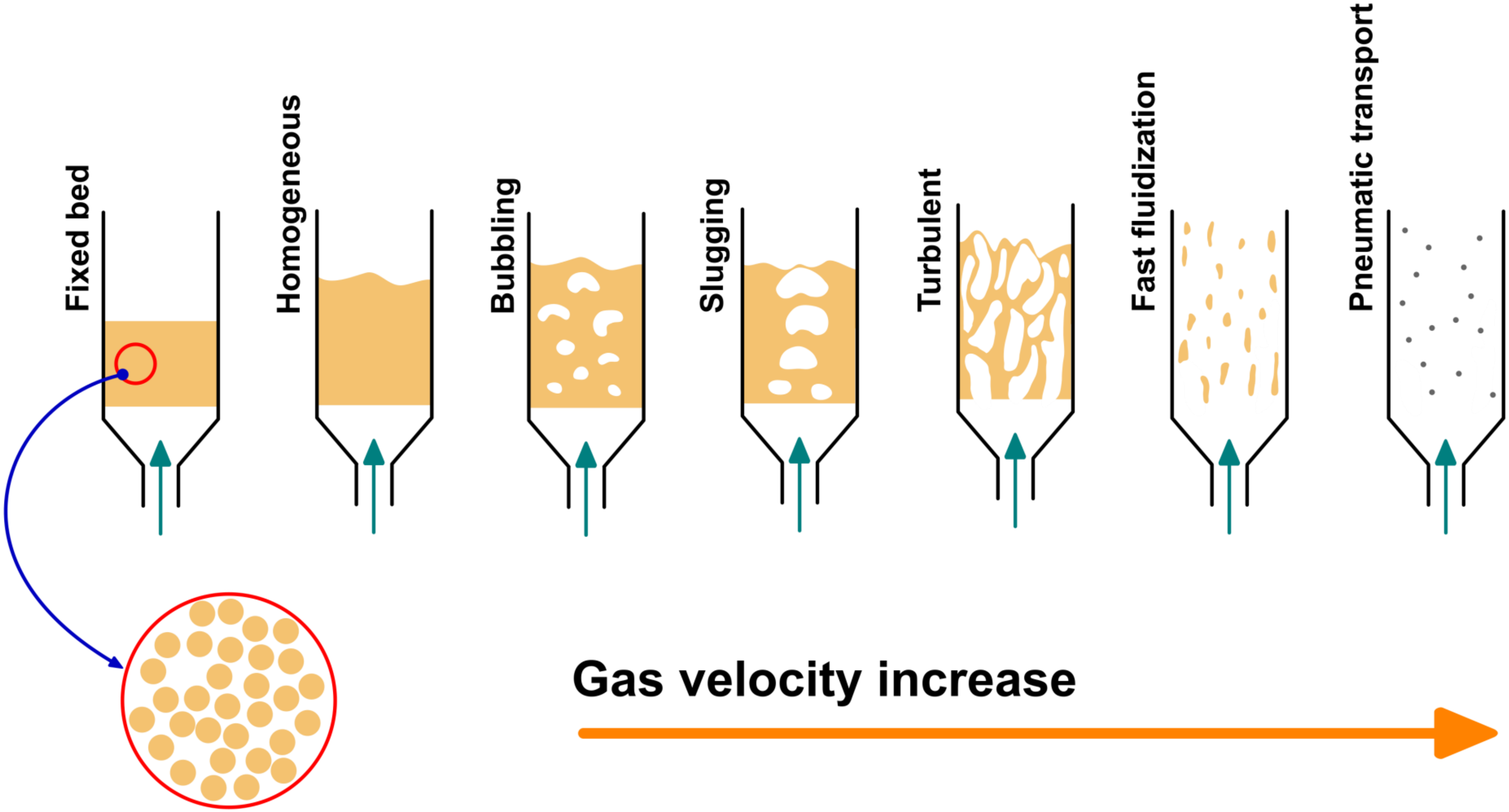


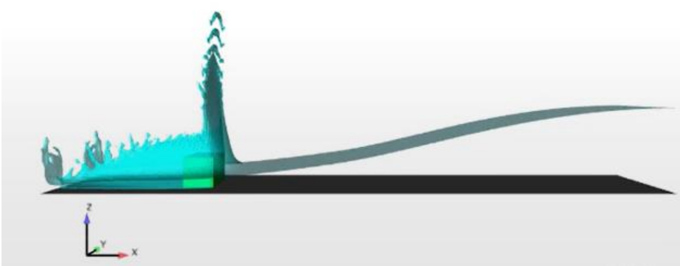
Photo credit: J. Ghajar. Oklahoma State University. Single/Two-phase Heat Transfer Laboratory. http://aghajar.okstate.edu/

Multiphase flow regimes

- Flow regimes in fluidized beds (solid-gas)



Multiphase flow videos



3D dam-break simulation
Comparison of numerical results obtained with OpenFOAM-4.x using a multiphase solver, against the experimental results obtained by the Maritime Research Institute Netherlands (MARIN).

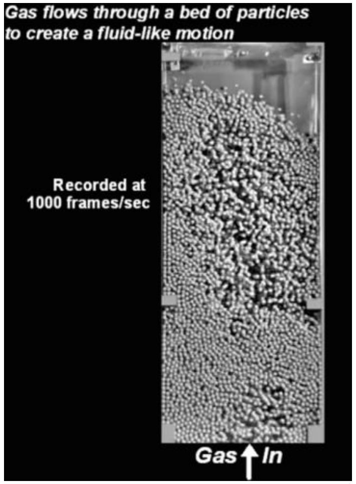
Useful references:

http://app.spheric-sph.org/sites/spheric/tests/test-2
http://www.wolfdynamics.com/tutorials.html?id=95
http://www.wolfdynamics.com/training/mphase/dbreak.mp4

A Volume-of-Fluid Based Simulation Method for Wave Impact Problems. Journal of Computational Physics 206(1):363-393. June, 2005.

Open link to videos

Multiphase flow videos



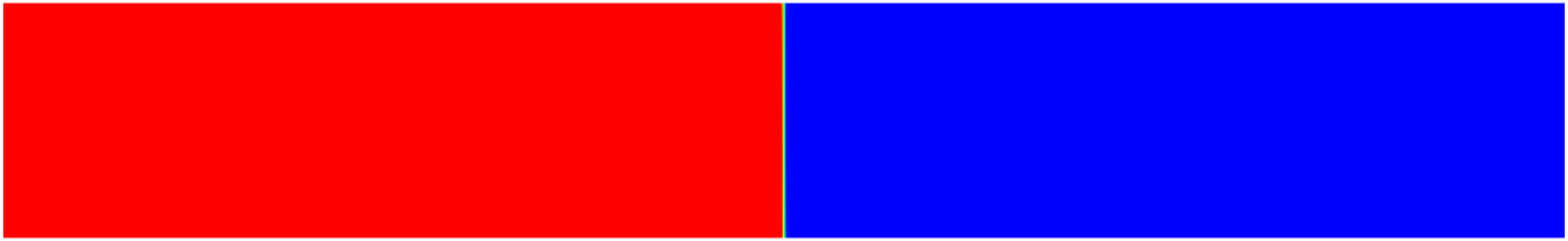
2013 GFM, Video 064: The Science and Beauty of Fluidization: High Speed Imaging of Particle Flow Fields
Authors: Frank Shaffer and Balaji Gopalan
DOE National Energy Technology Laboratory (USA)
http://www.wolfdynamics.com/training/mphase/2013_GFM_Video_054.mp4

2013 GFM, Video 054: Hydrodynamics Causes and Effects of Air Bubbles Rising in Very Viscous Media
Authors: Sharad Chand Ravinuthala
Department of Mechanical Engineering, West Virginia University (USA)
http://www.wolfdynamics.com/training/mphase/2013_GFM_Video_064.mp4

Open link to videos

Multiphase flows simulations using OpenFOAM

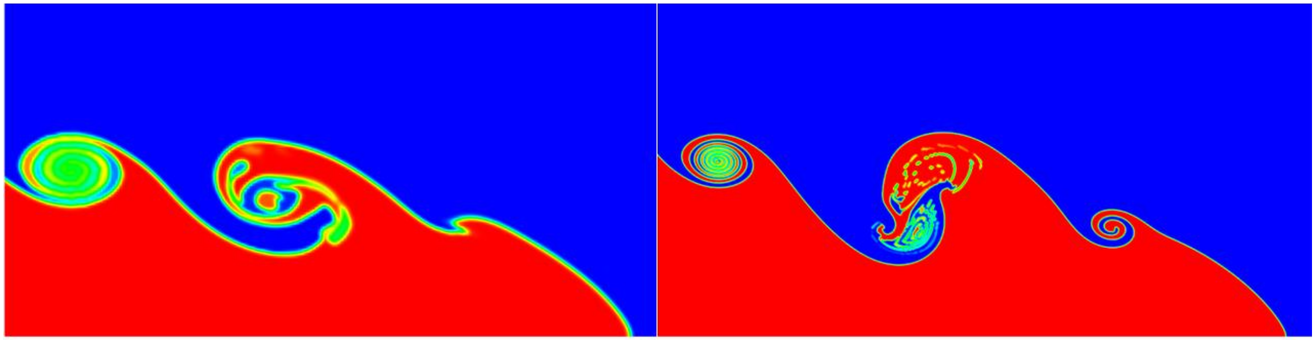
Simulation of gravity current – Onset of Kelvin-Helmholtz instability
Liquid-liquid interaction (VOF)



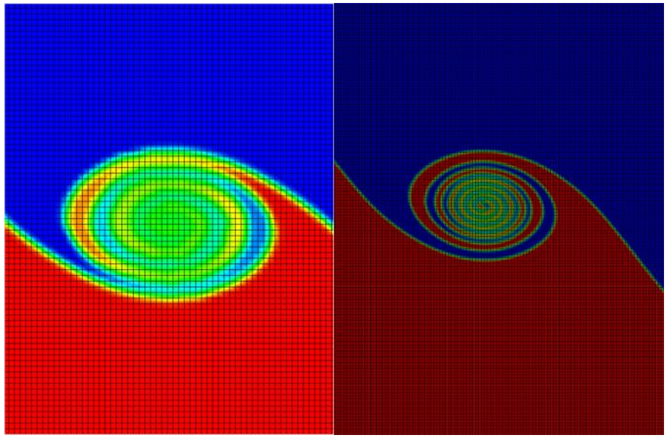
http://www.wolfdynamics.com/training/mphase/image1.gif

Multiphase flows simulations using OpenFOAM

Onset of Kelvin-Helmholtz instability – Liquid-liquid interaction (VOF)



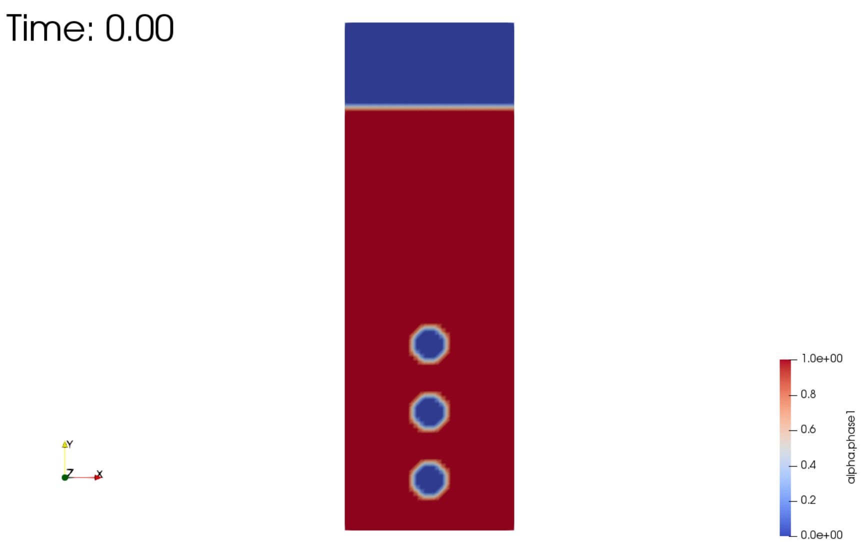
Medium grid size | Extra-fine grid size

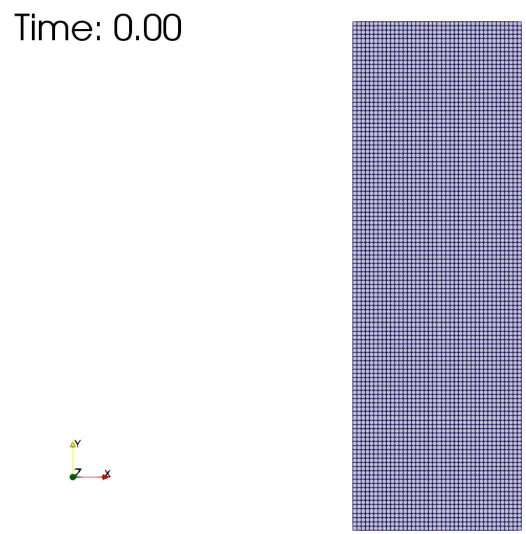


Medium grid size | Extra-fine grid size

Multiphase flows simulations using OpenFOAM

Three rising bubbles
(VOF with AMR)



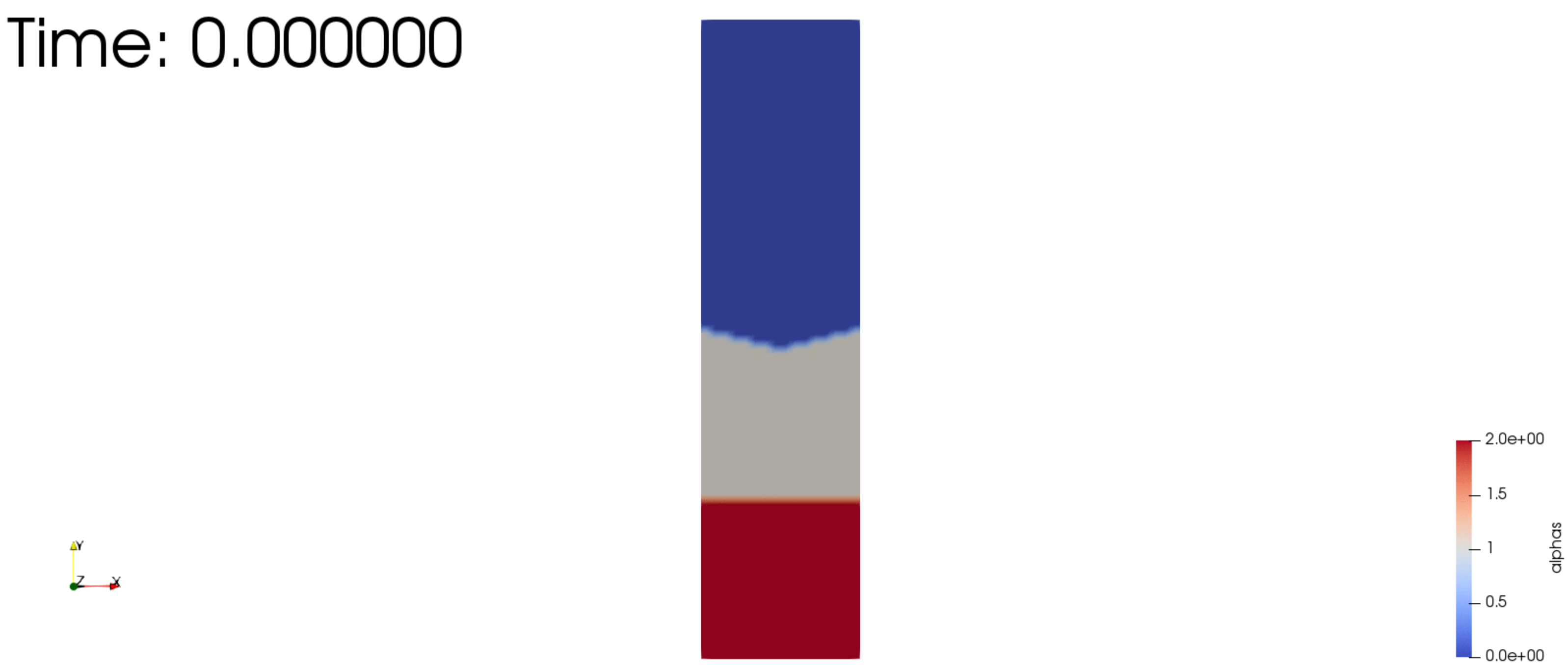


http://www.wolfdynamics.com/training/mphase/image2.gif

http://www.wolfdynamics.com/training/mphase/image3.gif

Multiphase flows simulations using OpenFOAM

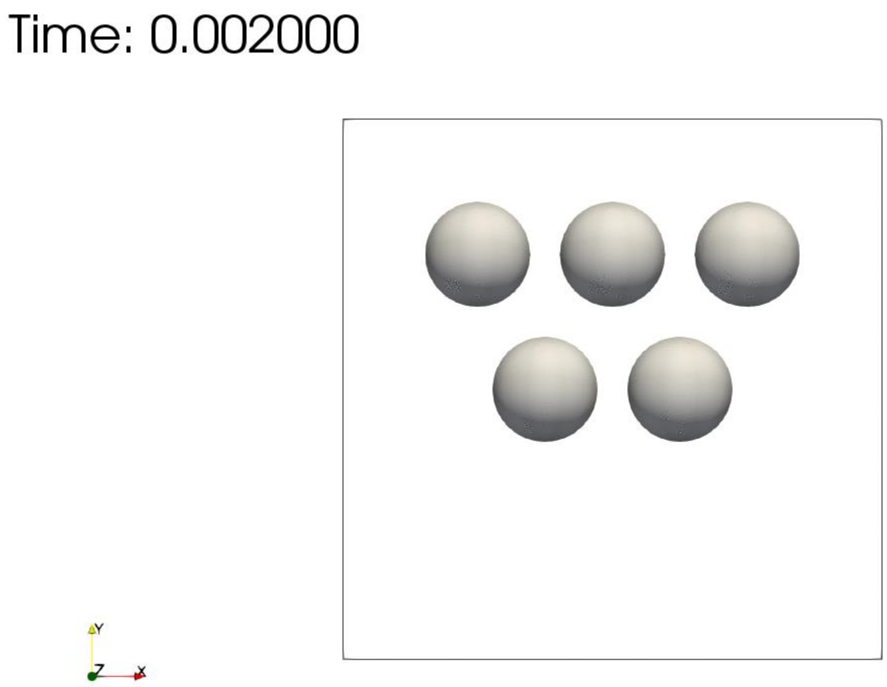
Rayleigh-Taylor instability – Liquid-liquid-liquid interaction (VOF with 3 phases)



www.wolfdynamics.com/training/mphase/image4.gif

Multiphase flows simulations using OpenFOAM

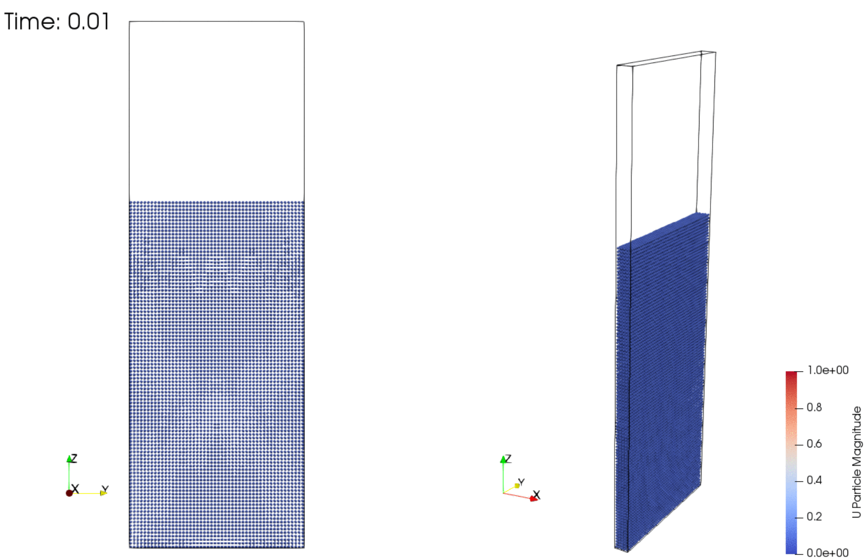
Particle interaction (no hydrodynamic coupling)



www.wolfdynamics.com/training/mphase/image5.gif

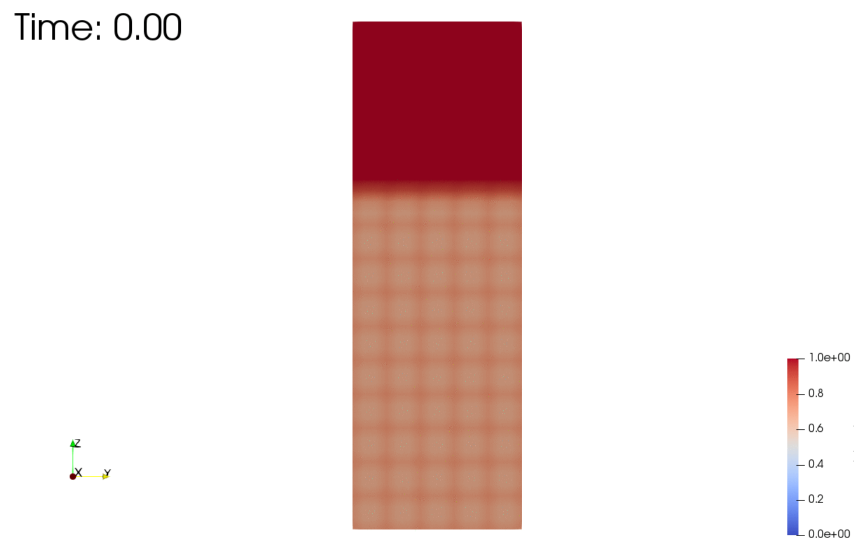
Multiphase flows simulations using OpenFOAM

Fluidized bed hydrodynamics – Gas-solid interaction with hydrodynamic coupling



Particles interaction

www.wolfdynamics.com/training/mphase/image6.gif

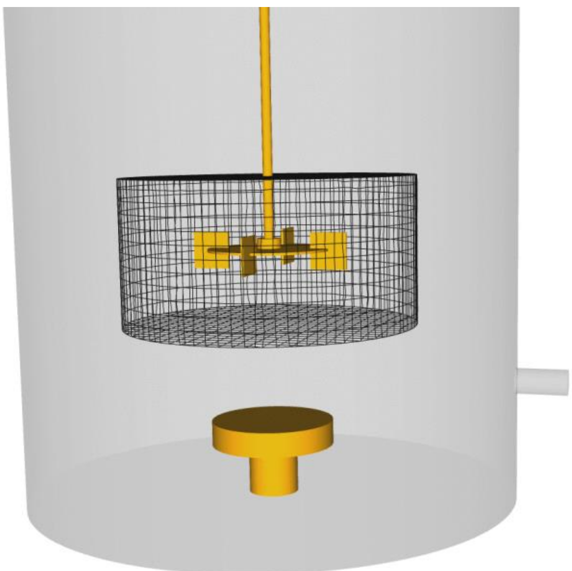


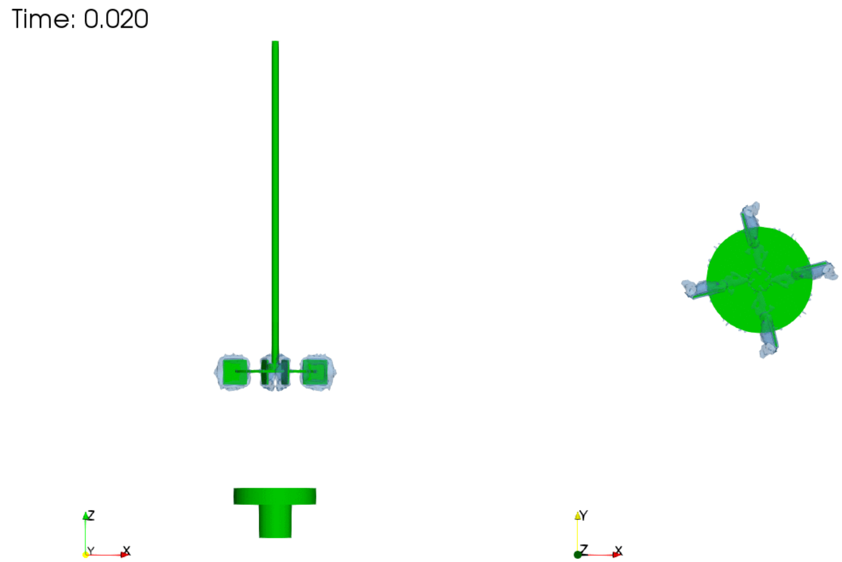
VOF air

www.wolfdynamics.com/training/mphase/image7.gif

Multiphase flows simulations using OpenFOAM

Continuous stirring tank reactor (CSTR)



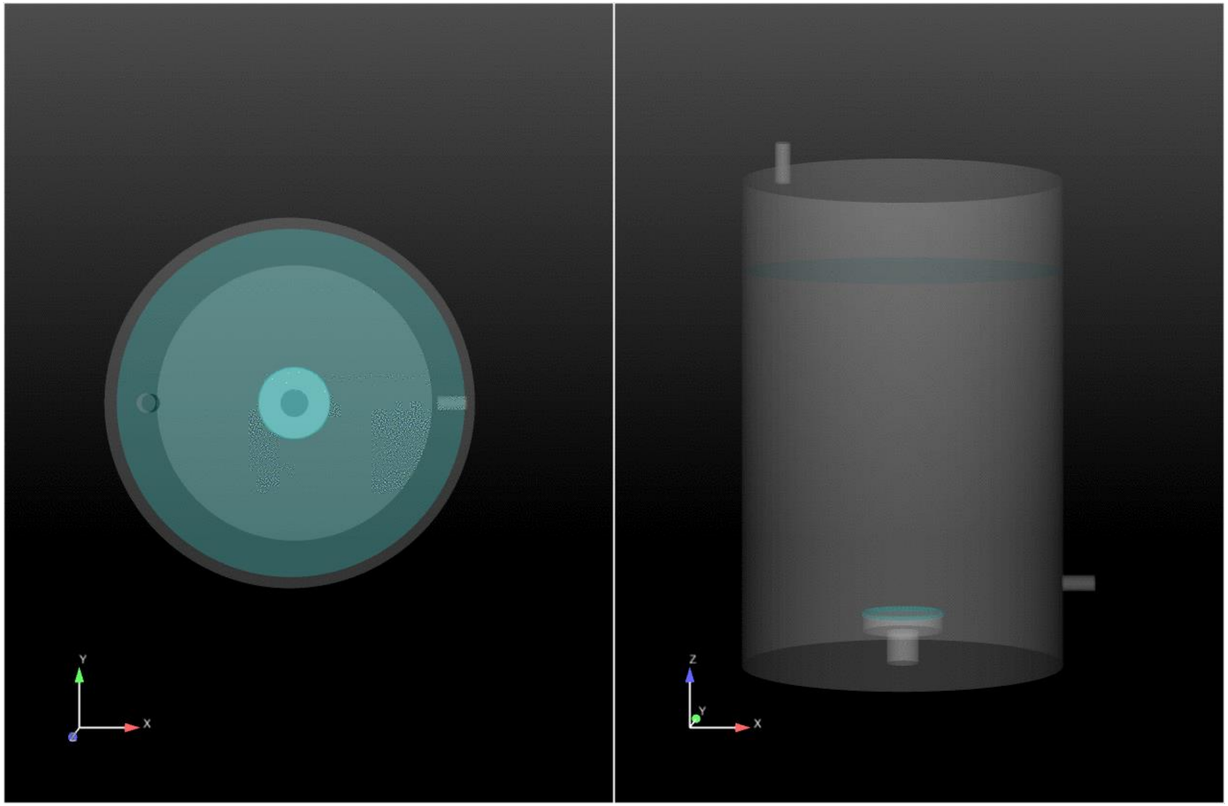


www.wolfdynamics.com/training/mphase/image8.gif

http://www.wolfdynamics.com/training/movingbodies/image13.gif

Multiphase flows simulations using OpenFOAM

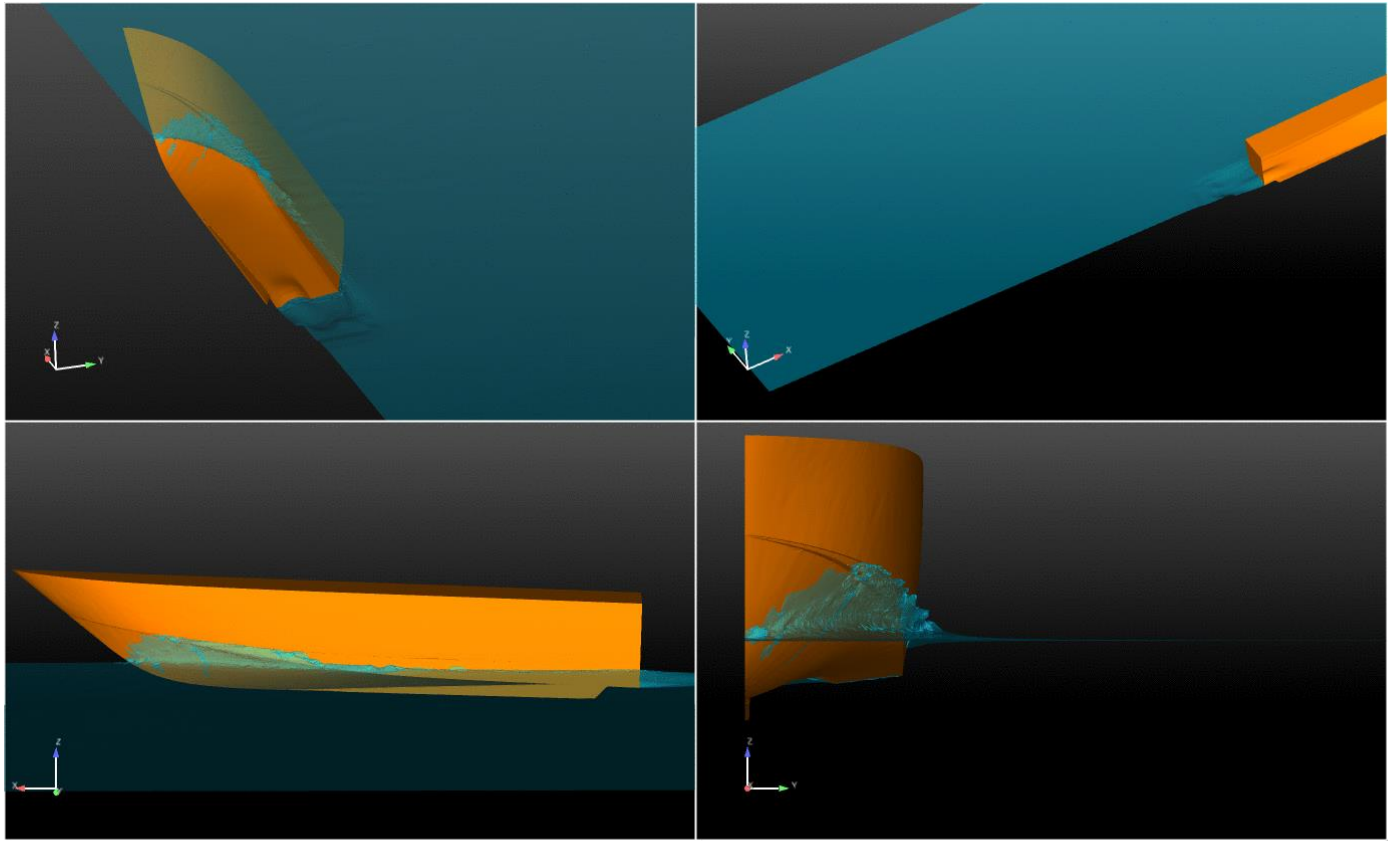
Dynamics of gas-liquid flow in a reactor tank (Euler-Euler)



www.wolfdynamics.com/training/mphase/image9.gif

Multiphase flows simulations using OpenFOAM

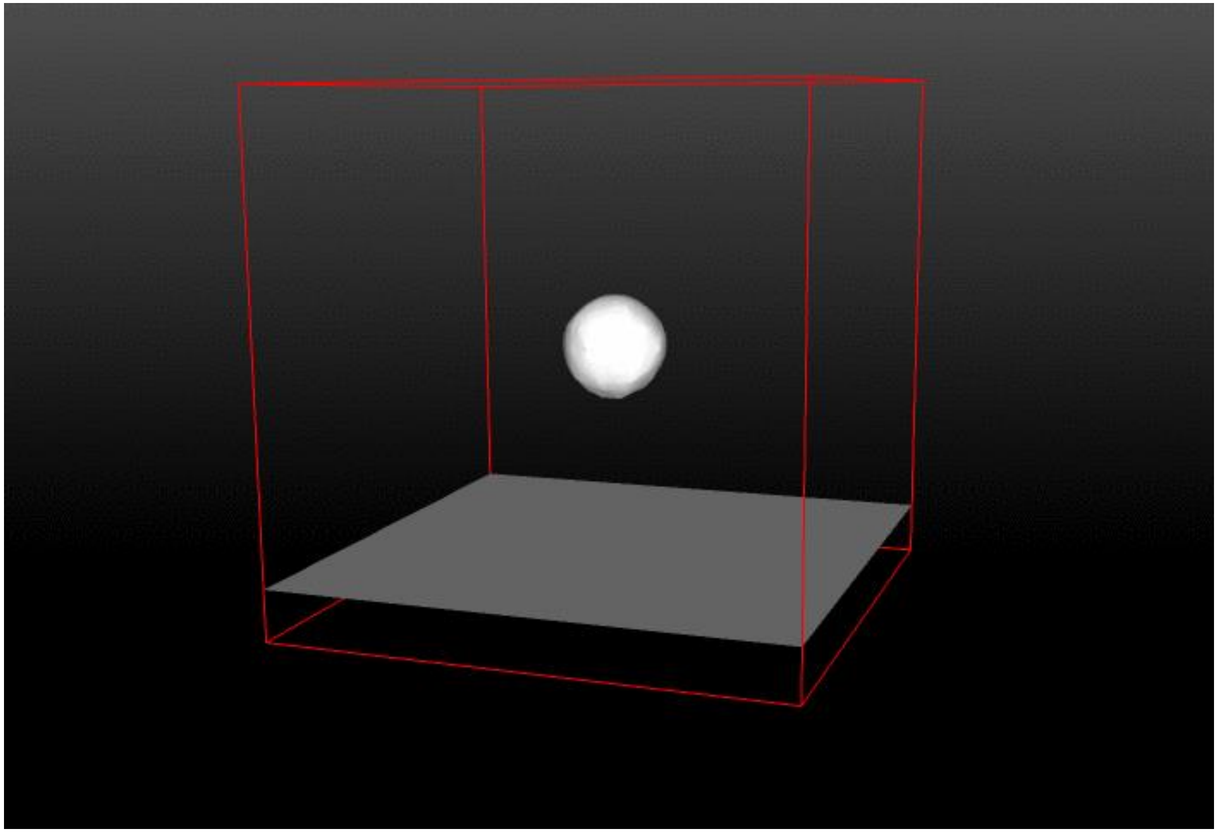
Free surface – Sea keeping (VOF)



www.wolfdynamics.com/training/mphase/image10.gif

Multiphase flows simulations using OpenFOAM

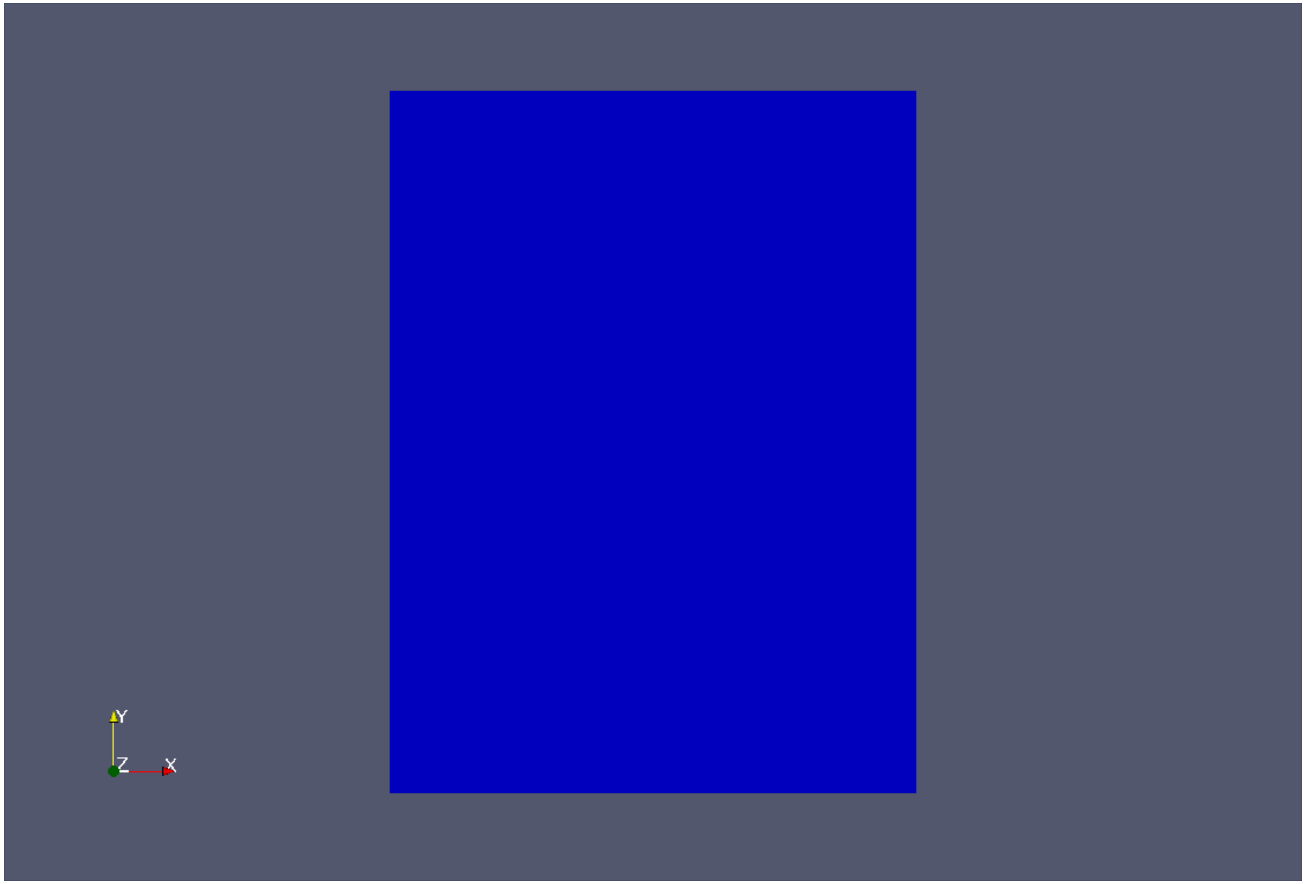
Free surface – Water splash (VOF)



www.wolfdynamics.com/training/mphase/image11.gif

Multiphase flows simulations using OpenFOAM

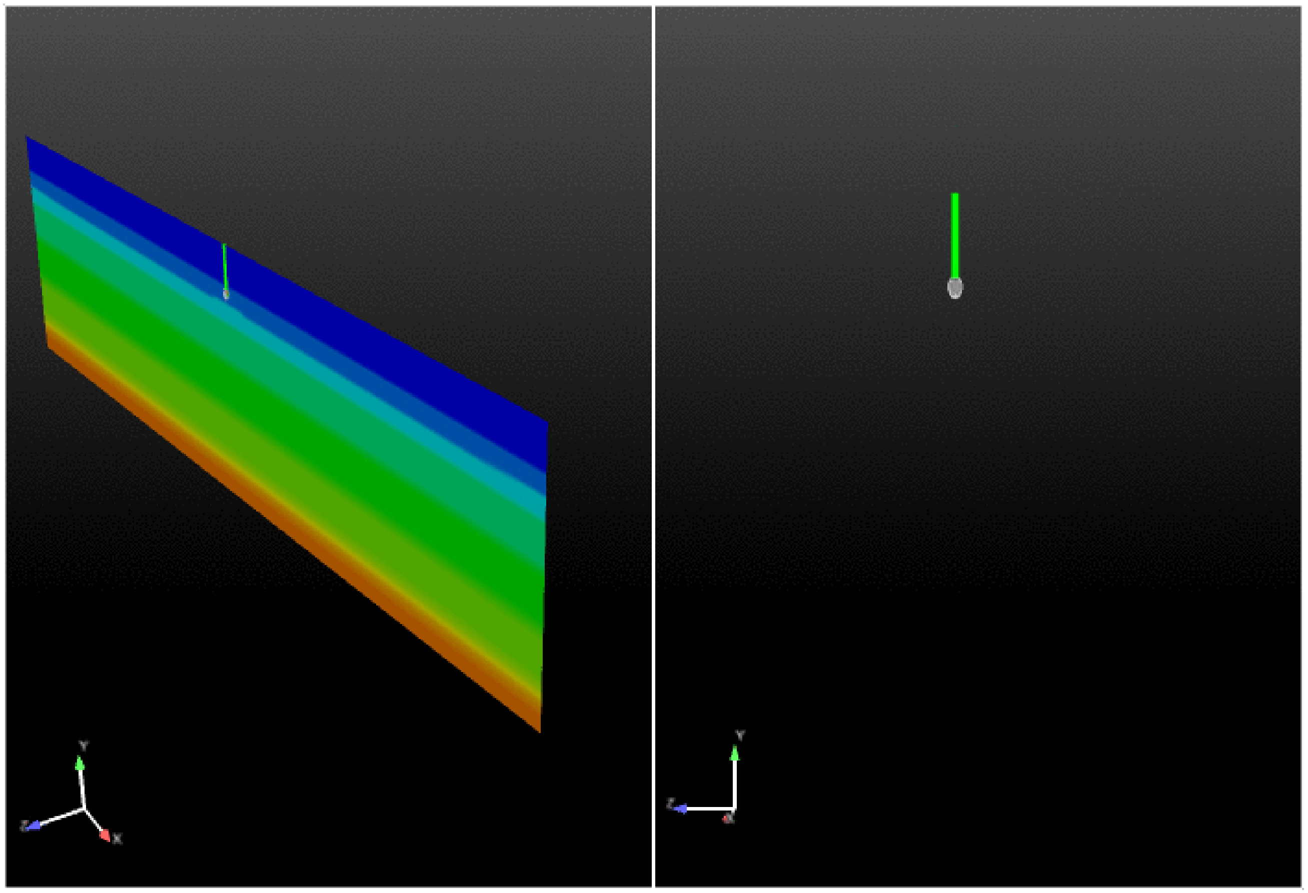
Surface film (Euler-Euler plus wall film)



www.wolfdynamics.com/training/mphase/image12.gif

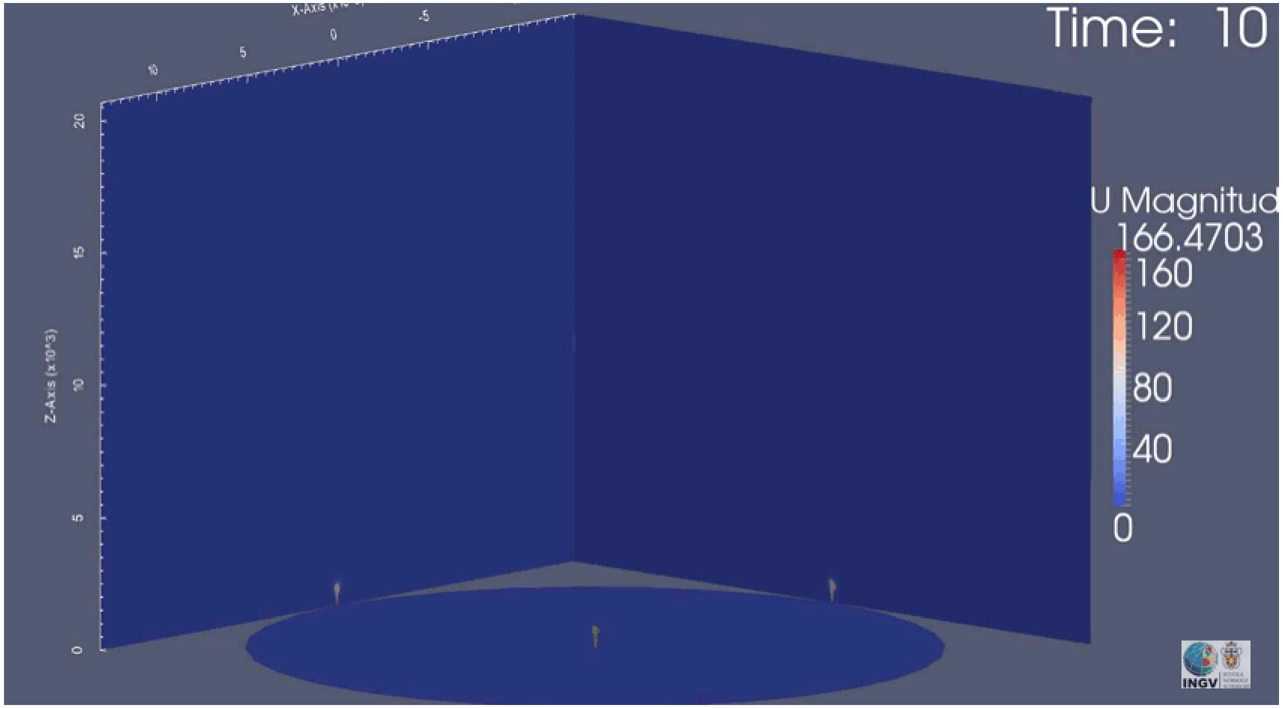
Multiphase flows simulations using OpenFOAM

Density current with pollutant transport – 8 phases (Euler-Euler)



www.wolfdynamics.com/training/mphase/image13.gif

Multiphase flows simulations using OpenFOAM



www.wolfdynamics.com/training/mphase/image14.gif

Disperse Multiphase Turbulence with OpenFOAM
Gas-particle decoupling in volcanic plumes and density currents
Simulation of a volcanic weak Plume with the dusty-gas (pseudo-gas) model with ash, H_2O inside the stratified atmosphere.

Matteo Cerminara
Istituto Nazionale di Geofisica e Vulcanologia Sezione di Pisa, Italy

Roadmap

1. ~~Introduction to multiphase flows~~
2. **Modeling approaches for multiphase flows**
3. Governing equations and interfacial momentum transfer models
4. Multiphase solvers in OpenFOAM
5. Selecting physical properties, phase interaction, and advanced models
6. Final remarks – Tips and tricks
7. Additional tutorials

Modeling approaches for multiphase flows

Overview/remarks on modeling approaches

- <u>Simulating multiphase flows is not an easy task.</u>
- The complex nature of multiphase flows is due to:
    - The transient nature of the flows.
    - The existence of dynamically changing interfaces.
    - Significant discontinuities (fluid properties and fluid separation).
    - Complicated flow field near the interface.
    - Interaction of small-scale structures (bubbles and particles).
    - Particle-particle interactions.
    - Mass transfer and phase change.
    - Turbulence.
    - Phase dispersion, mixing and transport of quantities.
    - Heterogenous and homogenous reactions.

Overview/remarks on modeling approaches

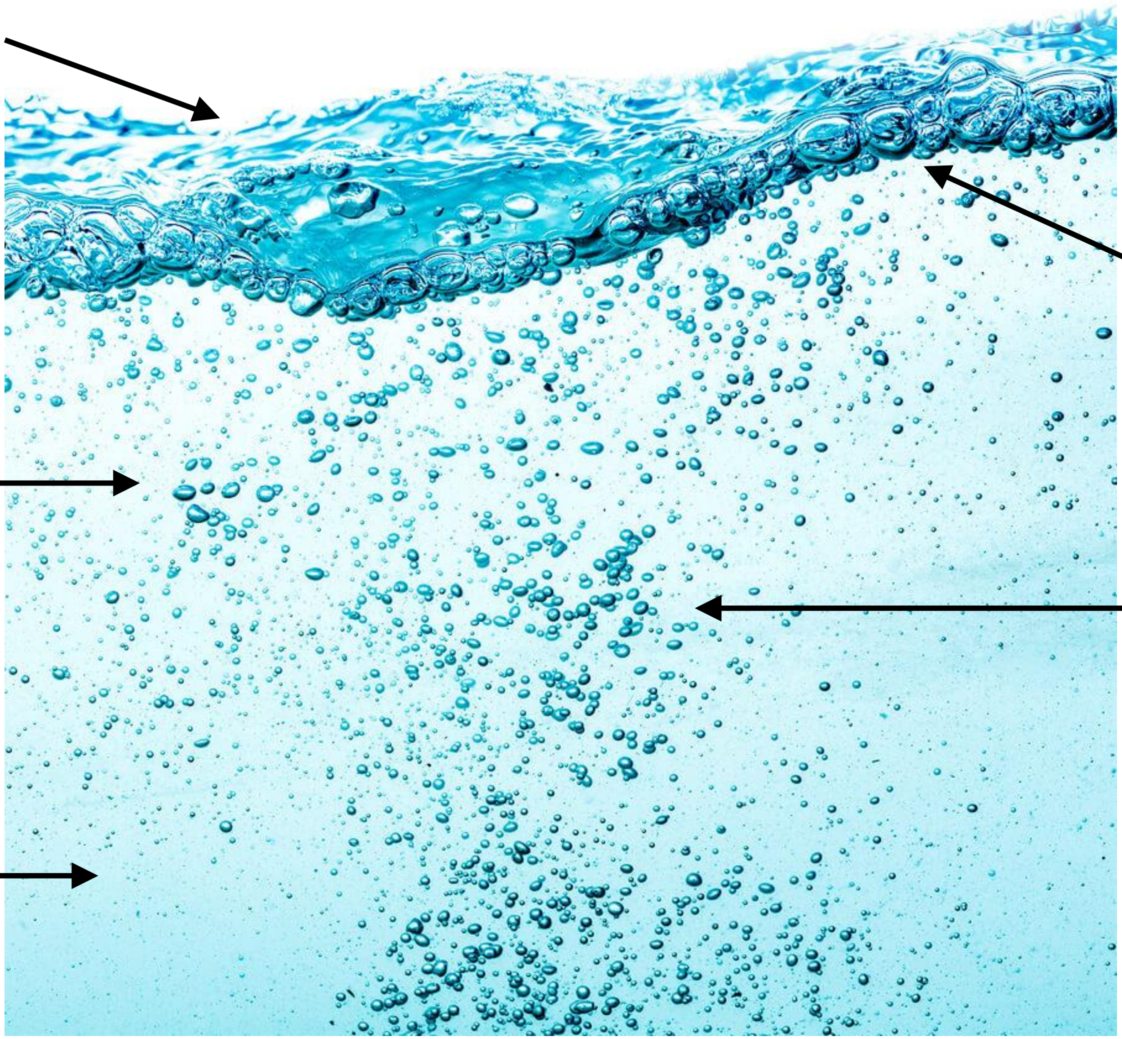
- Multiphase flows are inherently multi-scale in nature.
- Like in turbulence, we must account for the cascade effect of the various flow physics at different scales:
  - Large flow structures within the fluid flow (system-scale).
  - Local structural changes due to coalescence and breakage processes (meso-scales).
  - Motion and interaction of discrete constituents or small particles (micro-scale).
- And on top of this, we have turbulence models.
- You can also have cavitation.
- And to make it even more difficult, you can have phase change, mass transfer, and chemical reactions.

Modeling approaches for multiphase flows

Overview/remarks on modeling approaches

Free surface (system scale)

Multiphase flows are transient and multiscale



Large bubbles (system scale)

Medium bubbles (system-scales)

Bubble break-up and coalescence (meso-scales)

Small bubbles interaction and motion of small particles (micro-scales)

Modeling approaches for multiphase flows

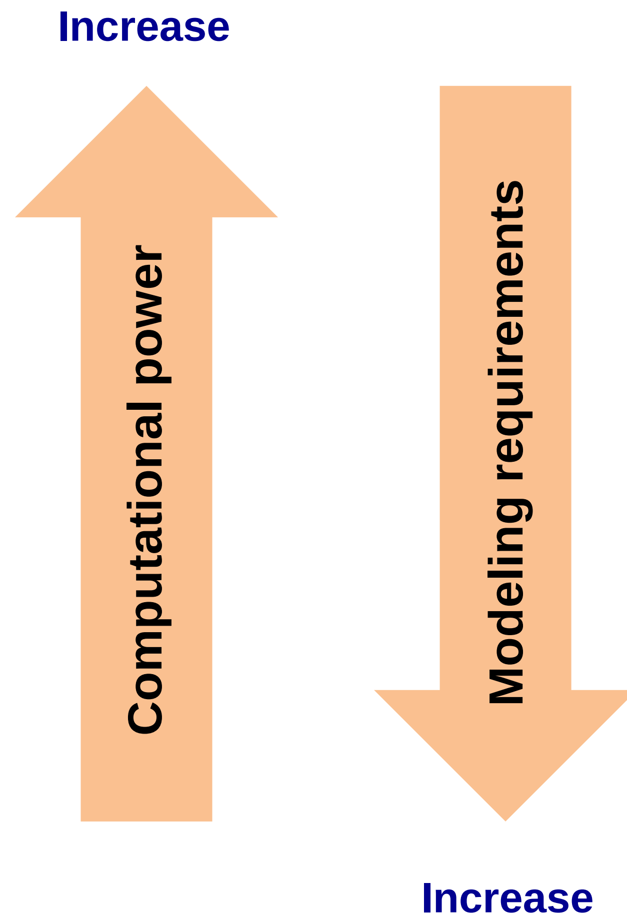
Overview/remarks on modeling approaches

- With today's computational resources, it is possible to solve directly multiphase flows and compute every detail of the flow. These are fully resolved simulations (kind of DNS).
- In the fully resolved approach we have a complete insight of the motion of the fluid system and every particle transported, bubbles and droplets, and the position of every interface.
- Such detailed and comprehensive treatment is restricted to low Reynolds number flows and for a limited number of particles, bubbles and droplets, due to limits in computational resources and simulation time.
- Macroscopic formulation of multiphase flows, based on models, enables simulation of large-scale, highly turbulent, and complex multiphase systems at a reduced computational cost.
- The predictions highly depends on realistic closure models for the interfacial exchange of mass, momentum and energy, as well as turbulent effects.
- Models, models, models.

Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

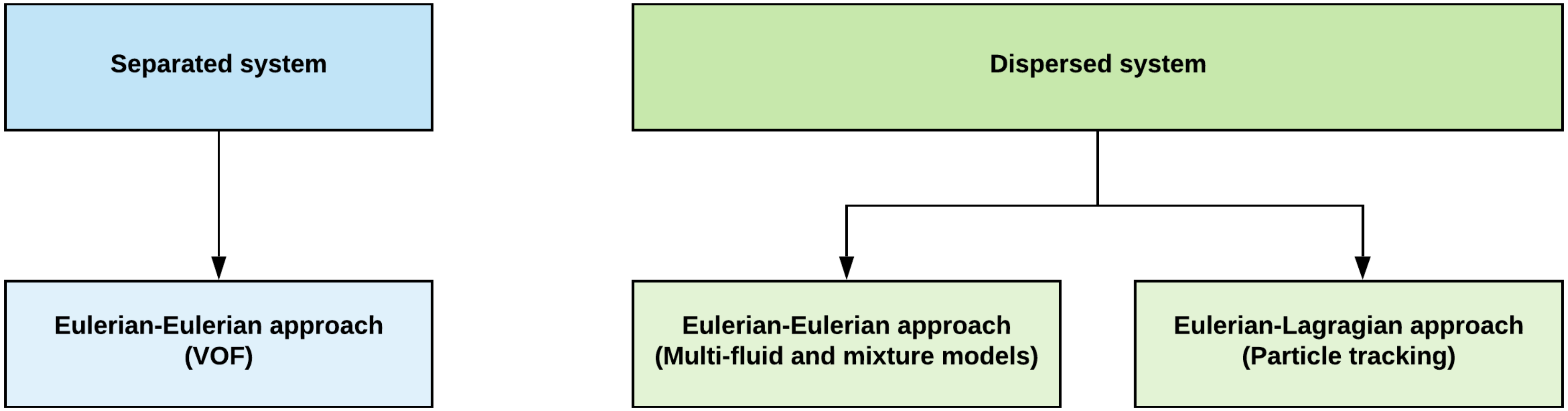
- **Fully resolved:** solves complete physics. All spatial and temporal scales are resolved.
- **Eulerian-Lagrangian:** solves idealized isolated particles that are transported with the flow. One- or two-way coupling is possible. Can account for turbulence, momentum transfer, and mass transfer.
- **Eulerian-eulerian:** solves two or more co-existing fluids. The system can be dispersed or separated, and can account for turbulence, momentum transfer, and mass transfer.



Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

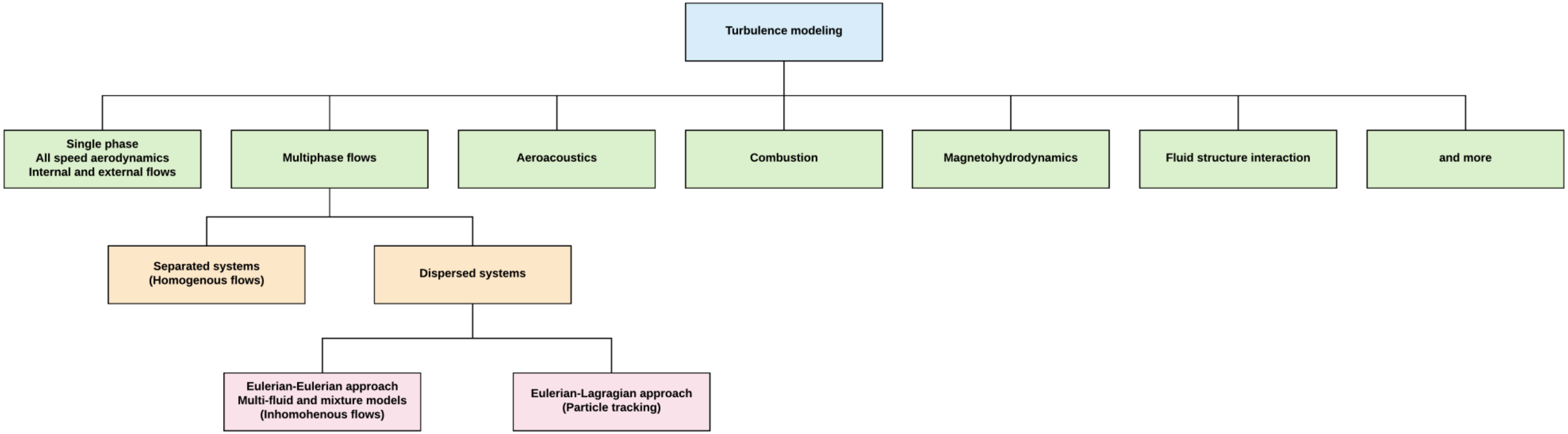
- As in turbulence modeling, we can take an approach that resolves all scales, but it is too expensive.
- Therefore, the need of using models.
- And depending on the multiphase system, there are models very specific for the multiphase physics involved.
- <u>Multiphase flows are heavily modeled.</u>



Modeling approaches for multiphase flows

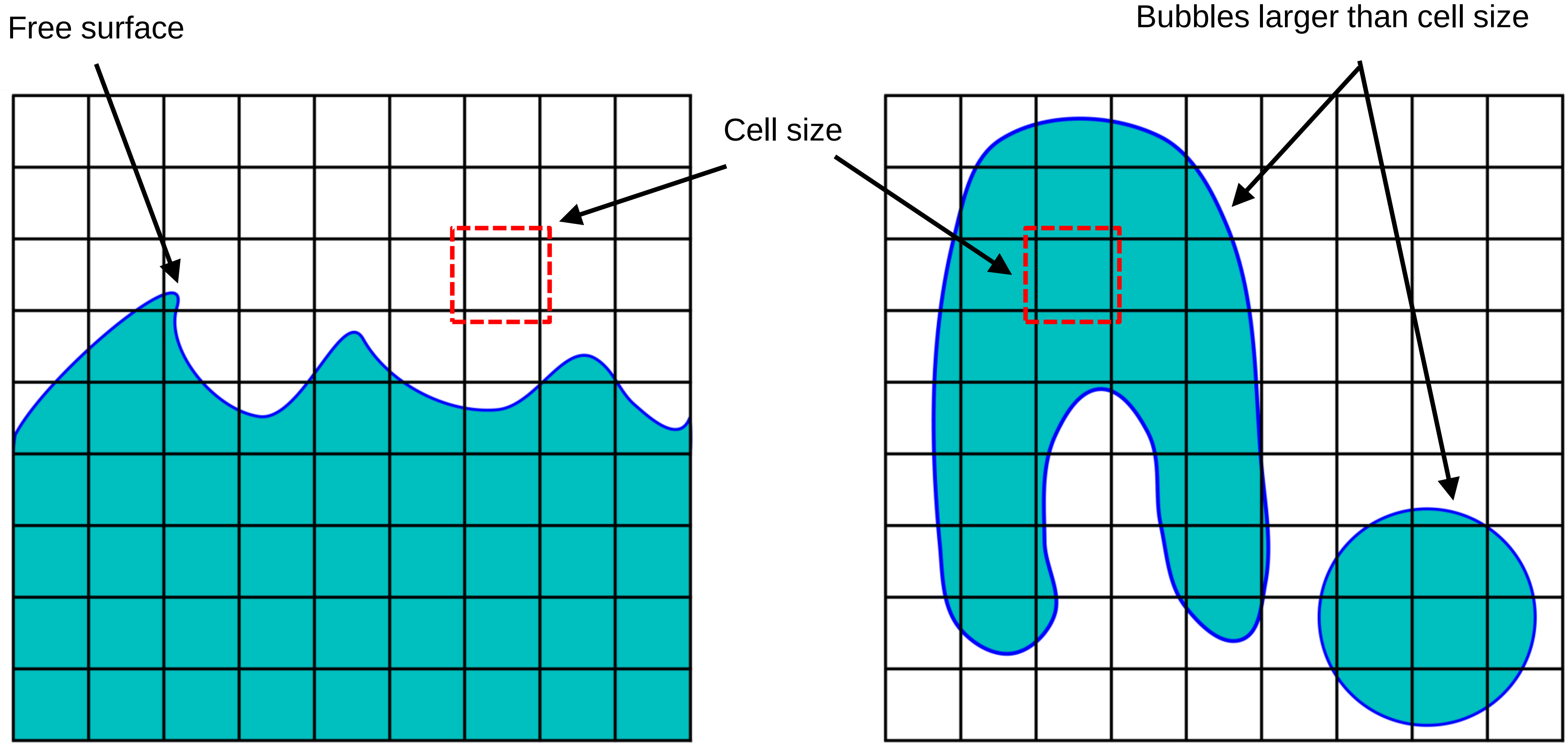
How to treat the wide range of behaviors in multiphase flows

- Turbulence modeling is on top of every type of physics to be simulated, including multiphase flows.
- Turbulence has a strong influence of the dynamics of multiphase flows, at large and small scales.



Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

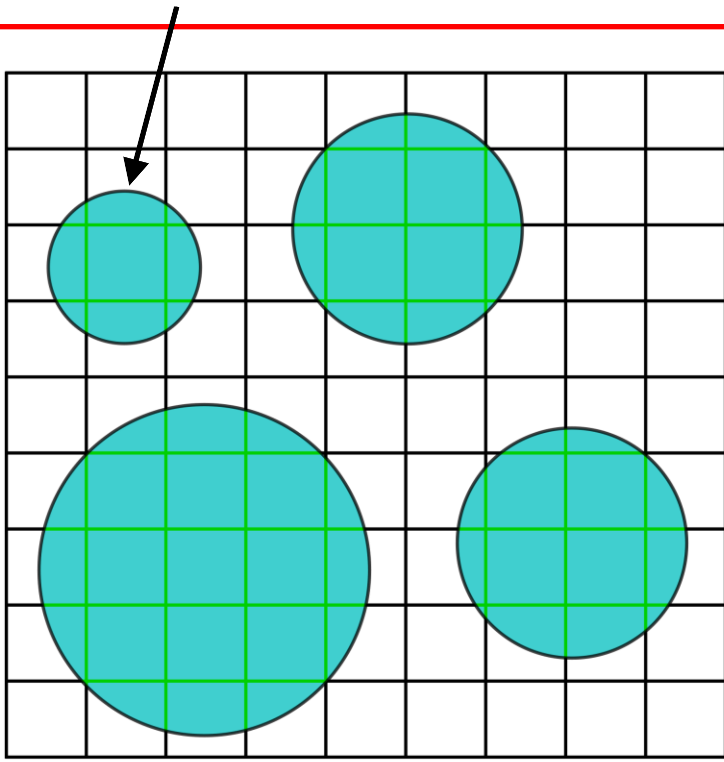


Free surface flows and surface tracking at scales larger than grid size Applicability of VOF method to separated systems (non-interpenetrating continua)

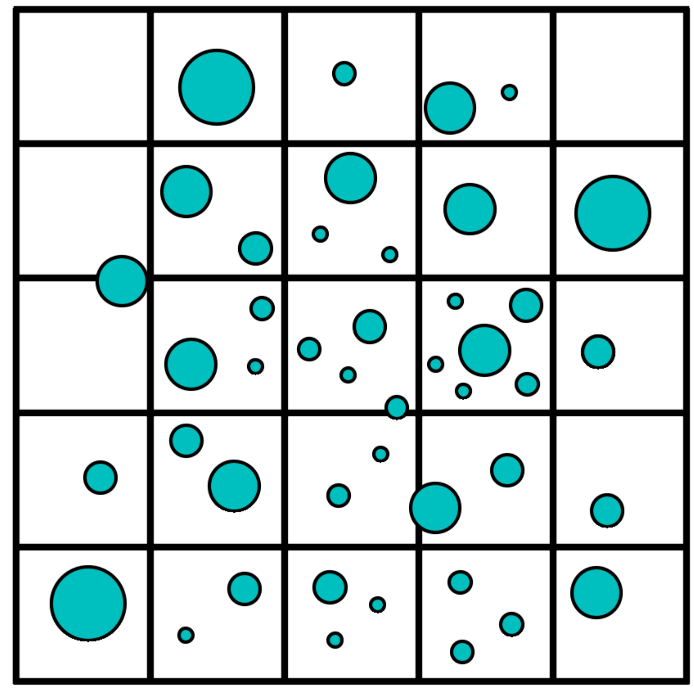
Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

Not ok. Only one cell to resolve the bubble.



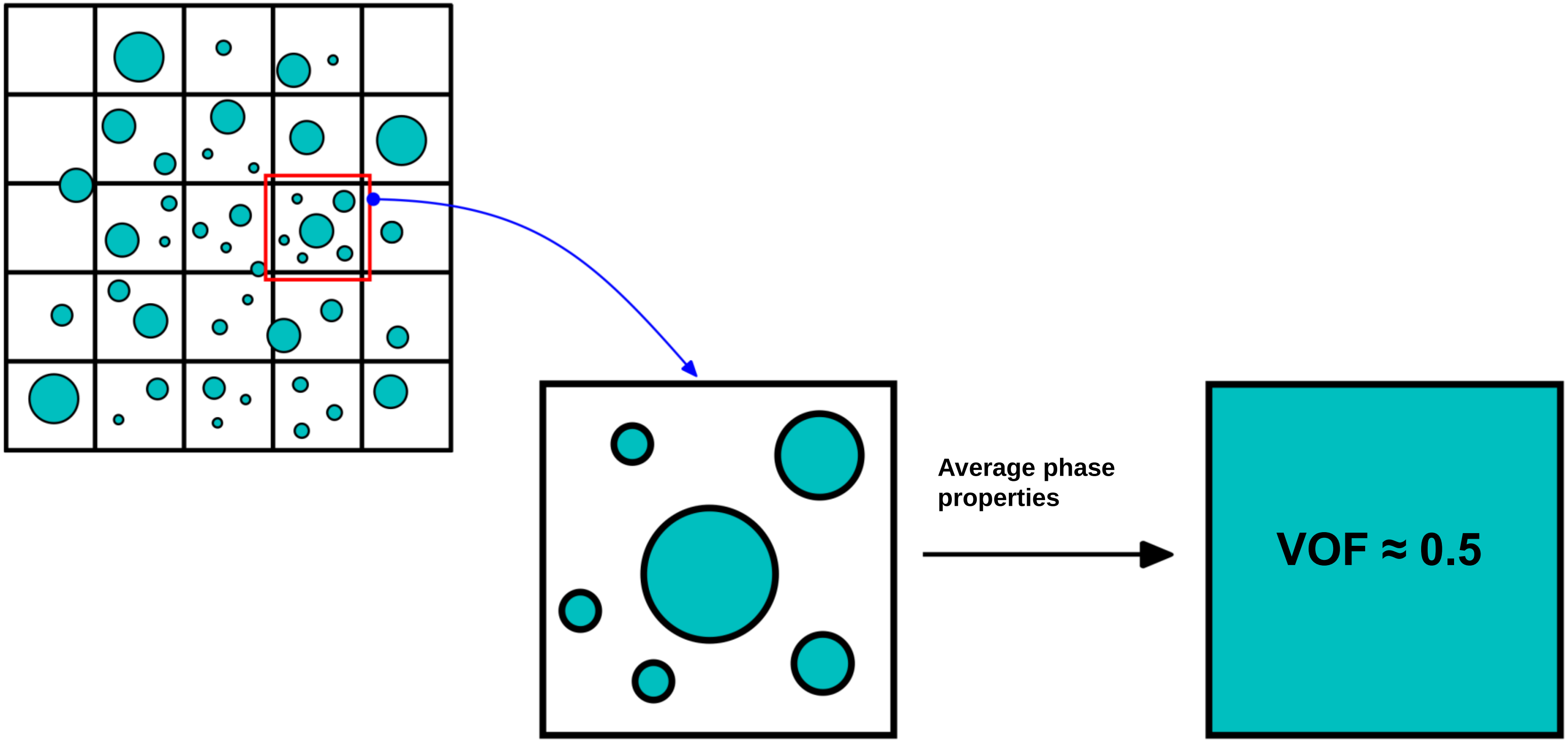
- Bubbles, droplets and/or particles bigger than grid scales (GS), can be resolved using VOF.
- To roughly resolve a bubble, you will need to use at least two cells in every direction.



- Bubbles, droplets and/or particles smaller than grid scales (sub-grid scales or SGS), can not be resolve using the VOF method.
- In this case, we need to use models to compute the disperse phase.

Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

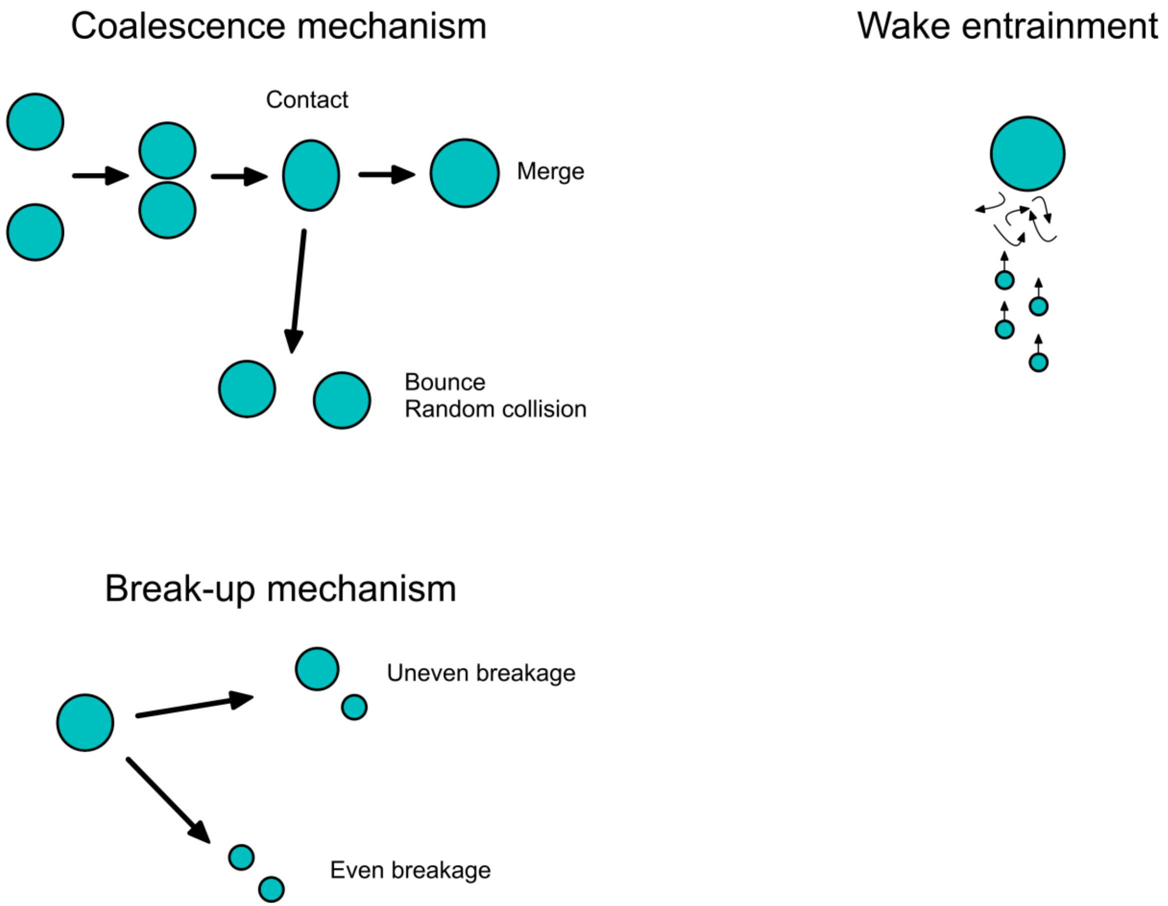


Dispersed phase in a continuous phase

VOF is not able to handle bubbles smaller than grid scales

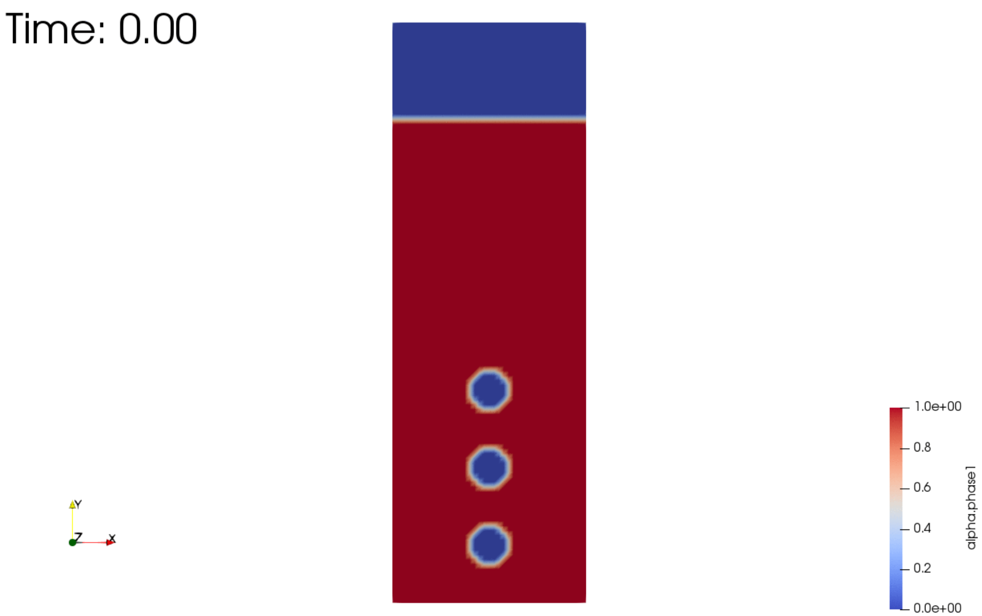
Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

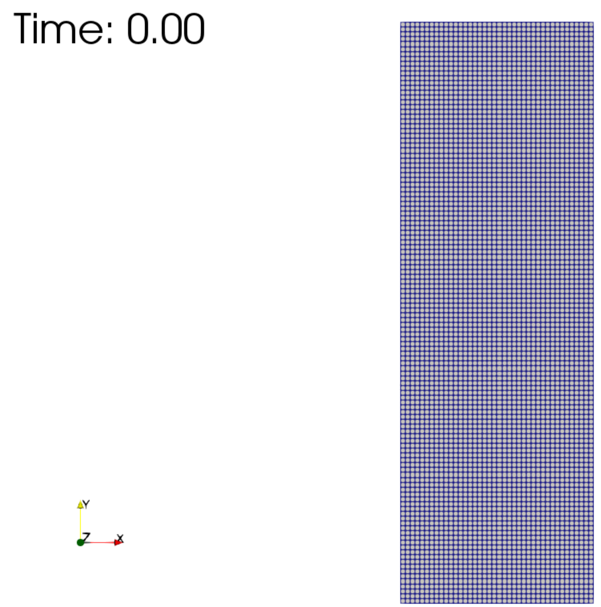


Bubble coalescence, bubble break-up and wake entrainment in dispersed systems

How to treat the wide range of behaviors in multiphase flows



http://www.wolfdynamics.com/training/mphase/image2.gif



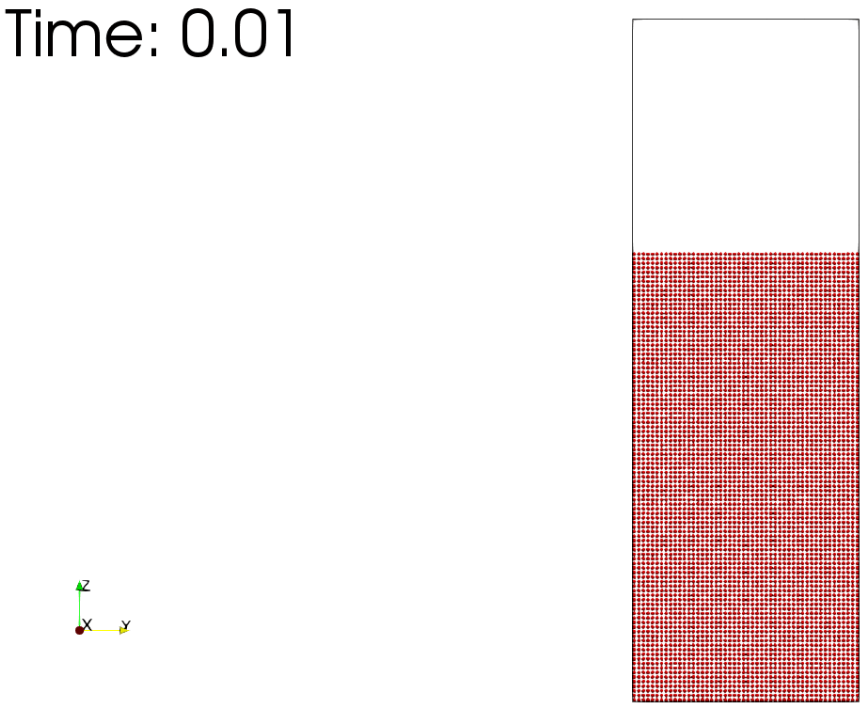
http://www.wolfdynamics.com/training/mphase/image3.gif

Bubble coalescence, bubble break-up and wake entrainment

In this simulation the bubbles are capture by using AMR. However, the smallest bubble that can be resolved is at the smallest grid size.

Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows



http://www.wolfdynamics.com/training/mphase/image46.gif

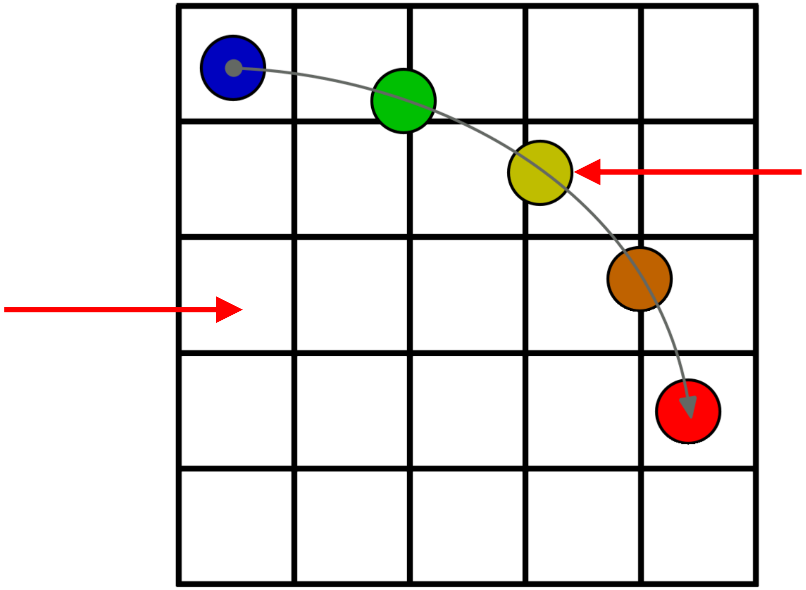
Particulate flows – Gas-solid interaction and particle-particle interaction

In this simulation the particle are simulated using an Eulerian-Lagrangian approach. The model takes into account particle-particle interaction and turbulence modeling.

Modeling approaches for multiphase flows

How to treat the wide range of behaviors in multiphase flows

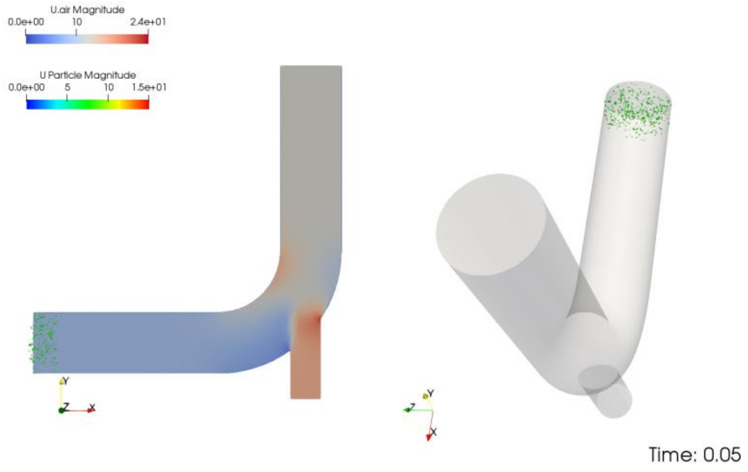
- Eulerian mesh.
- This is the mesh where the NSE are solved.



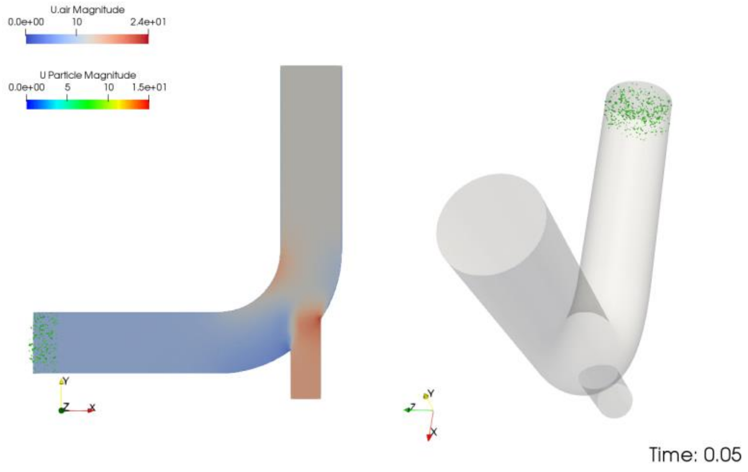
- Lagrangian mesh.
- This is not an actual mesh (like in the Eulerian case), it is a terminology for the particles.
- The equations of motion are solved for each single particle.

- **Eulerian-Lagrangian approach:**
  - The particles can be smaller or larger than the grid size.
  - Can have couple or uncoupled hydrodynamics.
  - For the hydrodynamic coupling, different approaches are available.
  - The particles move according to the flow momentum, and their position is tracked at all times.
  - The particles can interact with the boundaries: escape, bounce, stick, wall film, react, and so on.
  - Can account for particles interaction and mass transfer.
  - Different injection models and particle interaction models can be selected.

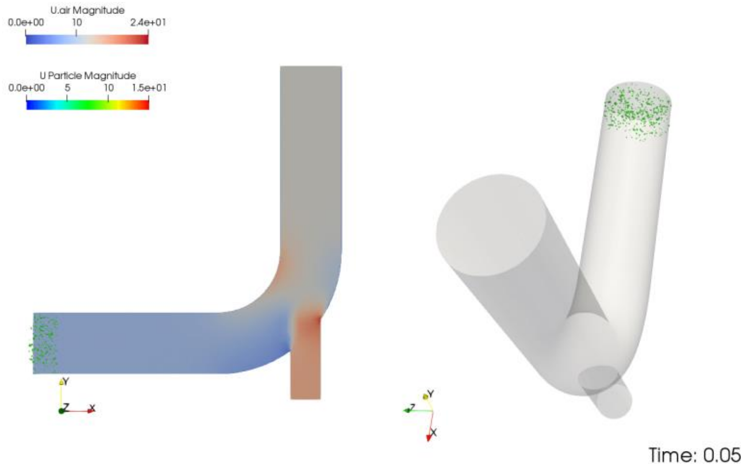
How to treat the wide range of behaviors in multiphase flows



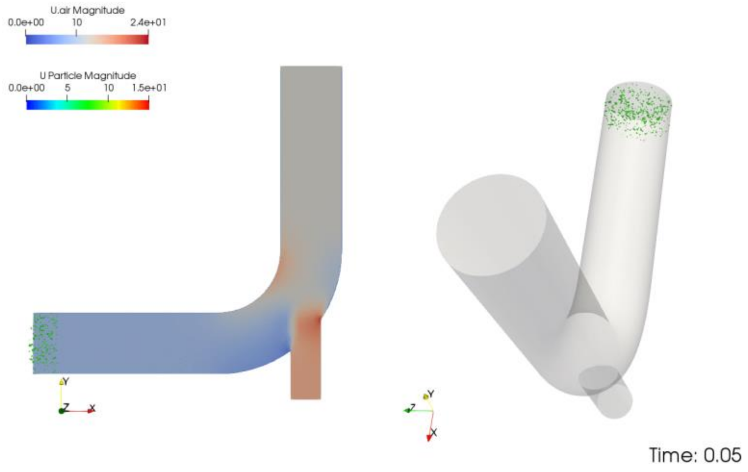
Hydrodynamic coupling with gravity
http://www.wolfdynamics.com/training/mphase/image30.gif



Hydrodynamic coupling with no gravity
http://www.wolfdynamics.com/training/mphase/image30a.gif



No hydrodynamic coupling with gravity
http://www.wolfdynamics.com/training/mphase/image31.gif

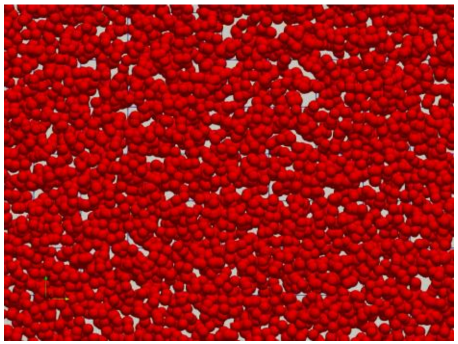


No hydrodynamic coupling with no gravity
http://www.wolfdynamics.com/training/mphase/image31a.gif

Modeling approaches for multiphase flows

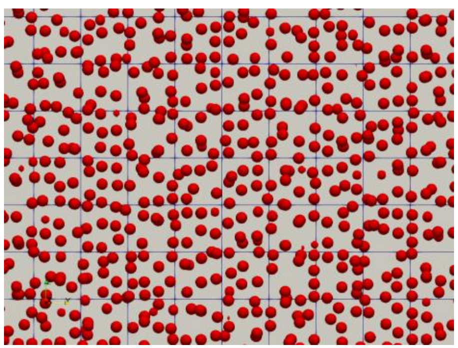
Characteristics of particulate flows

- Particle-particle interactions.

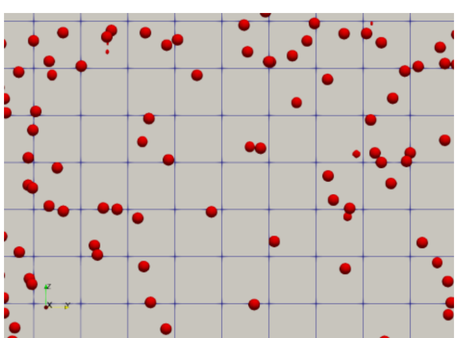


Dense flow

- Frictional dominated.
- Slow flows.
- Strain rate independent.
- Four-way coupling.
- Solid mechanics (Schaeffer, 1987).

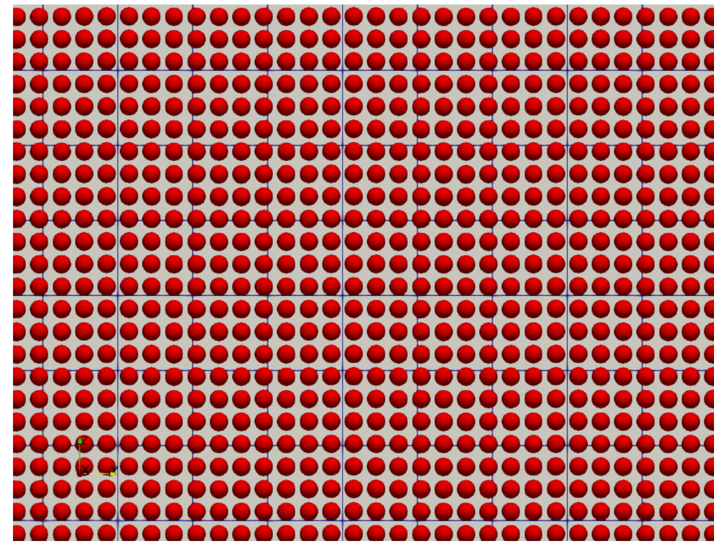


Intermediate flow – Transition regime



Dilute flow

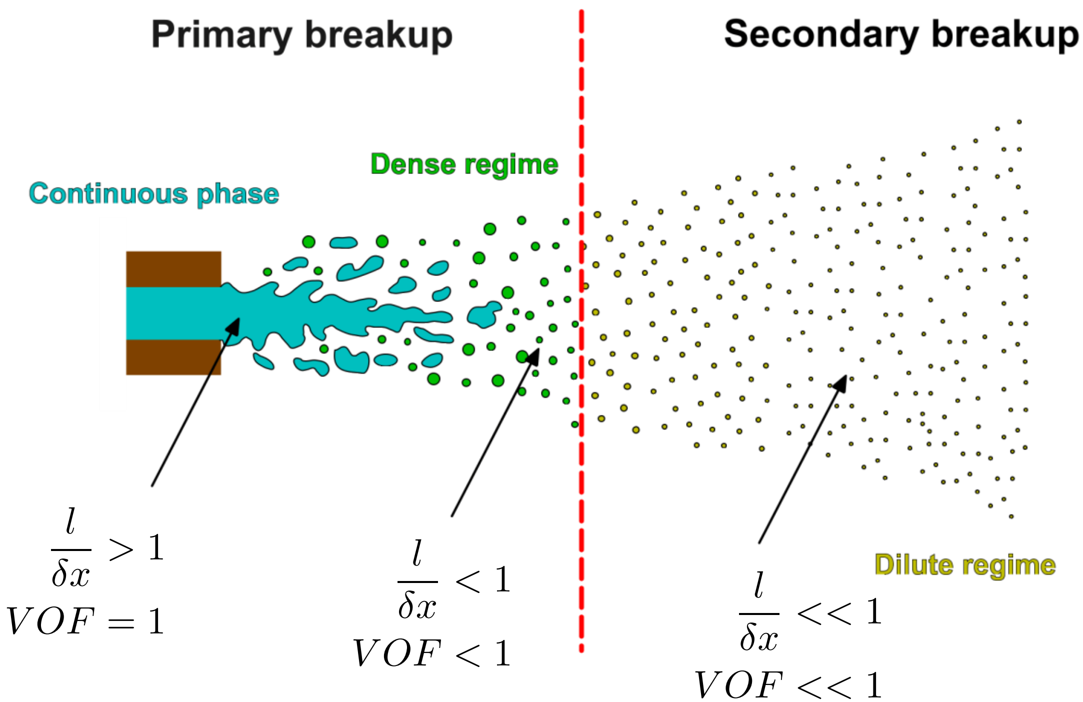
- Collision dominated
- Rapid flow
- Strain rate dependent
- One-way and two-way coupling.
- Kinetic Theory (Lun, 1984)



http://www.wolfdynamics.com/training/mphase/image47.gif

Challenges of modeling multiphase flows

- Interface separating the two phases is extremely thin and discontinuous.
- Density change across the interface is large.
- Interface exerts surface tension force on the liquid phase.
- Topology changes in vast length and time scales.
- Separated flow system, dispersed flow system and particles can be present in the same problem.
- Reactions and phase change.
- Example of a challenging problem: atomization of a liquid jet and combustion.



Dense regime: particles motion is controlled primarily by particle-particle collisions.

Dilute regime: particle motion is controlled by the drag and lift forces on the particle (particles follow the fluid flow).

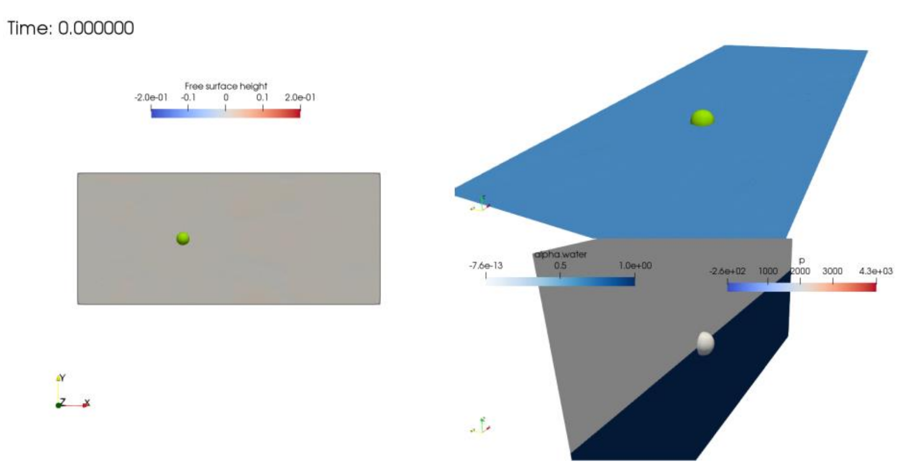
$$\text{volume fraction of a phase} = \frac{\text{volume of the phase in a cell/domain}}{\text{volume of the cell/domain}}$$

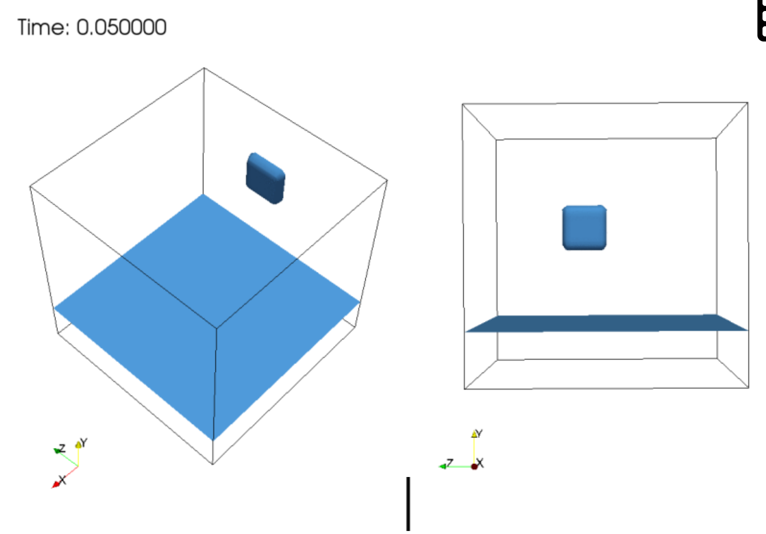
$$\delta x = \text{grid resolution}$$
$$VOF = \text{liquid volume fraction}$$
$$l = \text{liquid characteristic length scale}$$

Roadmap

1. ~~Introduction to multiphase flows~~
2. ~~Modeling approaches for multiphase flows~~
3. **Governing equations and interfacial momentum transfer models**
4. Multiphase solvers in OpenFOAM
5. Selecting physical properties, phase interaction, and advanced models
6. Final remarks – Tips and tricks
7. Additional tutorials

Volume-of-Fluid (VOF) governing equations for separated systems





http://www.wolfdynamics.com/training/mphase/image15.gif

http://www.wolfdynamics.com/training/mphase/image16.gif

Volume-of-Fluid (VOF) governing equations for separated systems

- The incompressible, isothermal governing equations can be written as follows,

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

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla p + \nabla \cdot \tau + \rho g + f_\sigma + \rho S$$

Surface tension

Source terms:

- Porous media
- Coriolis forces
- Centrifugal forces
- Mass transfer
- and so on …

Volume-of-Fluid (VOF) governing equations for separated systems

- Phase transport equation and interface tracking with surface compression,

$$\frac{\partial \gamma}{\partial t} + \nabla \cdot \mathbf{U}\gamma + \nabla \cdot (\mathbf{U}_r \gamma (1-\gamma)) = 0$$

Interface compression velocity

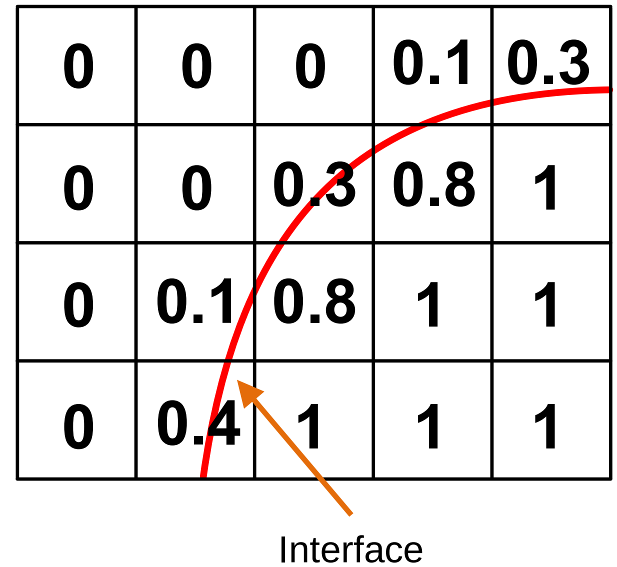
$$0 < \gamma < 1$$

Volume fraction

- You can see the volume fraction γ as a pointer that indicates what phase (with the corresponding physical properties), is inside each cell of the computational domain.

Volume-of-Fluid (VOF) governing equations for separated systems

- For example, in the case of two phases where phase 1 is represented by $\gamma = 1$ and phase 2 is represented by $\gamma = 0$; a volume fraction value of 1 indicates that the cell is fill with phase 1; a volume fraction of 0.8 indicates that the cell contains 80% of a phase 1; and a volume fraction of 0, indicates that the cell is fill with phase 2.
- The values between 0 and 1 can be seen as the interface between the phases.
- The fluid properties can be written on either side of the interface as follows,



$$\rho = \gamma_1 \rho_1 + (1 - \gamma_1)\rho_2$$

$$\mu = \gamma_1 \mu_1 + (1 - \gamma_1)\mu_2$$

Volume-of-Fluid (VOF) governing equations for separated systems

- Phase transport equation and interface tracking with surface compression,

$$\frac{\partial \gamma}{\partial t} + \nabla \cdot \mathbf{U}\gamma + \nabla \cdot (\mathbf{U}_r \gamma (1-\gamma)) = 0$$

$$\mathbf{U}_r = \min(c_\gamma |\mathbf{U}|, \max(|\mathbf{U}|)) \cdot \mathbf{n}$$

- Coefficient to control the magnitude of the compression (cAlpha).
- Usually 0 < cAlpha < 2.
- Recommended value 1.

Volume-of-Fluid (VOF) governing equations for separated systems

- Continuum surface force (CSF) model for the calculation of the surface tension,

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla p + \nabla \cdot \tau + \rho g + \boxed{f_\sigma} + \rho S$$

Surface tension coefficient

$$f_\sigma = \sigma k \nabla \gamma$$

$$k = \nabla \cdot \left(\frac{\nabla \gamma}{|\nabla \gamma|} \right)$$

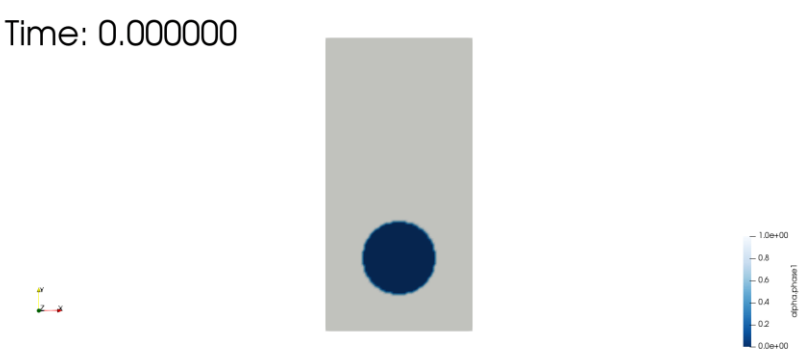
- *k* represents the local curvature, based on local gradients.
- The CSF model neglects the effects of a variable surface tension coefficient.

Volume-of-Fluid (VOF) governing equations for separated systems

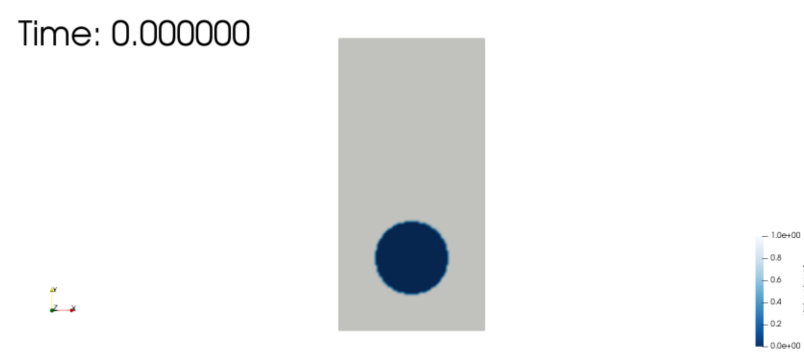
- In the VOF method, the curvature calculations are based on the volume fraction field.

$$k = \nabla \cdot \left(\frac{\nabla \gamma}{|\nabla \gamma|} \right)$$

- Calculation of curvature based on volume fractions can be inaccurate and cause convergence issues in problems dominated by surface tension.
- It is recommended not to use aggressive slope limiters when computing the local curvature.
- The curvature resolution should be as smooth as possible.



Smooth slope limiter used for curvature computations
http://www.wolfdynamics.com/training/mphase/image48.gif



Aggressive slope limiter used for curvature computations
http://www.wolfdynamics.com/training/mphase/image49.gif

Governing equations and interfacial momentum transfer models

Volume-of-Fluid (VOF) governing equations for separated systems

- To determine if the effects of surface tension are important, first evaluate the Reynolds number and then,
  - For *Re* << 1 compute the Capillary number:

$$Ca = \frac{\mu \mathbf{U}}{\sigma}$$

  - For *Re* >> 1 compute the Weber number:

$$We = \frac{\rho L \mathbf{U}^2}{\sigma}$$

  - Surface tension is important when *Ca* << 1 or *We* << 1.

- At large scales, the effects of surface tension can be neglected.

Volume-of-Fluid (VOF) governing equations for separated systems

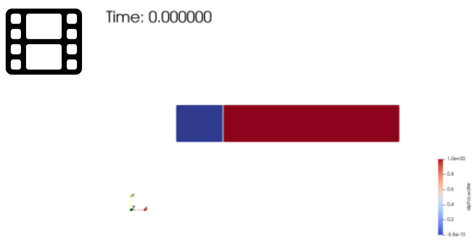
- Wall adhesion and contact angle:
    - It is possible to model contact angles and adhesives forces acting between fluid and walls.
    - They are important when modeling meniscus, capillary effects, and wettability.
    - Can impose static equilibrium contact angle or dynamic contact angle at the walls.

Hydrophilic

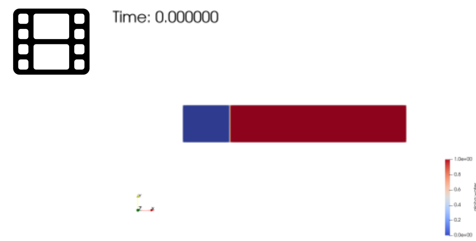
Hydrophobic

Volume-of-Fluid (VOF) governing equations for separated systems

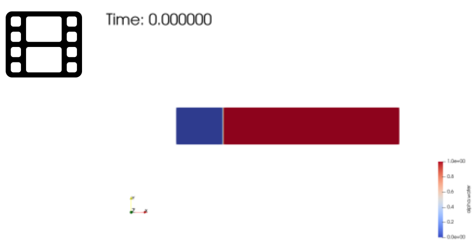
- Wall adhesion and contact angle:



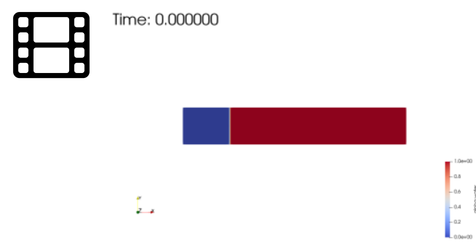
Contact angle: 45
Sigma: 0.072
Execution time:
Approx. 200 seconds



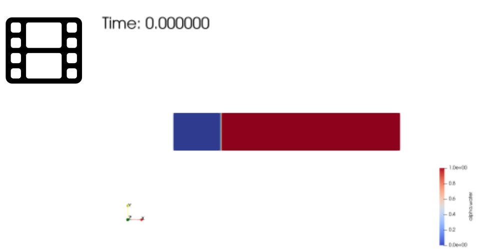
Contact angle: 90
(Equivalent to zeroGradient)
Sigma: 0.072
Execution time:
Approx. 200 seconds



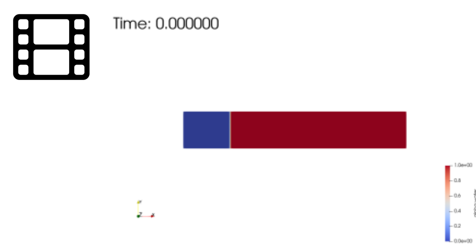
Contact angle: 10
Sigma: 0.072
Execution time:
Approx. 200 seconds



Contact angle: 45
Sigma: 1.0
Execution time:
Approx. 1500 seconds



Contact angle: 135
Sigma: 0.072
Execution time:
Approx. 200 seconds



Contact angle: 45
Sigma: 0.072
Slip walls
Execution time:
Approx. 200 seconds

Volume-of-Fluid (VOF) governing equations for separated systems

Additional notes on the solution of the phase/mass-fractions.

$$\frac{\partial \gamma}{\partial t} + \nabla \cdot \mathbf{U}\gamma + \nabla \cdot (\mathbf{U}_r \gamma (1 - \gamma)) = 0$$

- The phase/mass-fractions in the volume of fluid (VoF) and dispersed system solvers for multiphase flow in OpenFOAM is solved using the multi-dimensional limiter for explicit solution (MULES).
- The MULES method can solve system of two or more phases.
- The original formulation of the MULES is explicit, which imposes terrible constrains on the CFL number.
- The new formulation of the MULES*, Predictor-Corrector Semi-Implicit MULES (implicit prediction and explicit correction), overcome the limitations of the explicit method.
- The user has the choice between the fully explicit MULES and semi-implicit MULES.

* https://openfoam.org/release/2-3-0/multiphase/

Volume-of-Fluid (VOF) governing equations for separated systems

Additional notes on the solution of the phase/mass-fractions.

- For interface capturing, OpenFOAM uses,
  - The MULES algorithm (semi-implicit and second order in time) to ensure that the volume fraction (alpha) remains between strict bounds of 0 and 1.
  - The interface compression scheme, based on counter-gradient transport, to maintain sharp interfaces during a simulation.

- At the following link, you can find the release notes related latest developments related to the interface capturing added in OpenFOAM 8,
  - https://cfd.direct/openfoam/free-software/multiphase-interface-capturing/

- Among the improvements and developments, was the addition of the Piecewise-linear interface calculation (PLIC) family of interpolation schemes.
  - PLIC represents an interface by surface-cuts which split each cell to match the volume fraction of the phase in that cell.
  - The surface-cuts are oriented according to the point field of the local phase fraction.
  - The phase fraction on each cell face (the interpolated value), is then calculated from the amount submerged below the surface-cut.

Volume-of-Fluid (VOF) governing equations for separated systems

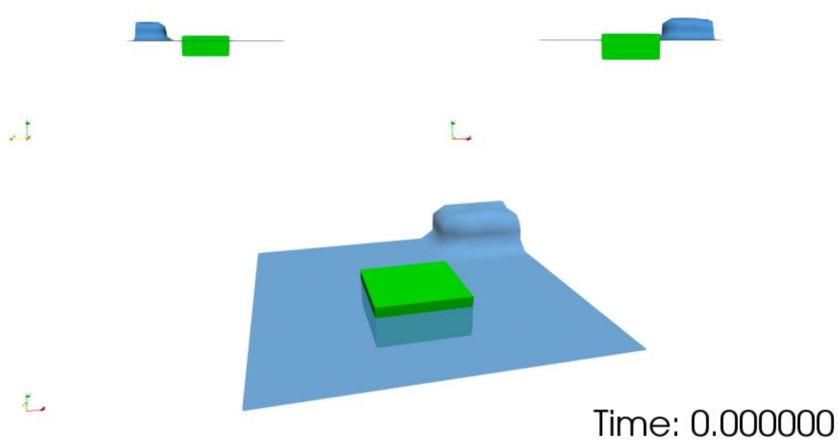
Additional notes on the solution of the phase/mass-fractions.

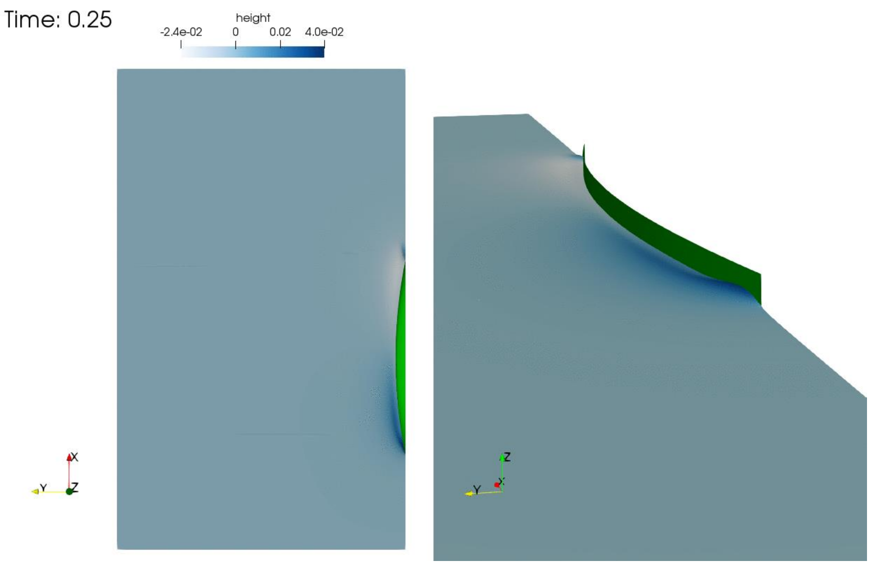
- The basic PLIC method generates a single cut so cannot handle cells in which there are multiple interfaces or where the interface is not fully resolved.
- In those cells, the interpolation reverts to an alternative scheme, typically standard interface compression.
- The PLIC method, with a fallback to interface compression, produces robust solutions and it can run with large time steps.
- In addition, the MPLIC or multicut PLIC method was also introduced in OpenFOAM 8.
- Where a single cut is insufficient, MPLIC performs a topological face-edge-face walk to produce multiple splits of a cell.
- If that is still insufficient, MPLIC decomposes the cell into tetrahedrons on which the cuts are applied.
- The extra cutting carries an additional computational cost but requires no fallback to the interface compression method.
- The PLIC and MPLIC methods are more precise than interface compression for meshes with refinement patterns.
- The PLIC and MPLIC methods potentially provides extra accuracy at the cost of an additional computational cost.

Governing equations and interfacial momentum transfer models

Volume-of-Fluid (VOF) governing equations for separated systems

- The best know applications of the VOF method are towing tank simulations and rigid body motion.
- These type of simulations are relatively easy.
- We will not deal with them; they are closely related to naval applications.
- However, we will give you the building blocks.
- Have in mind that when setting boundary and initial conditions for turbulent flows, it is recommended to use the primary phase properties to compute the turbulent quantities.

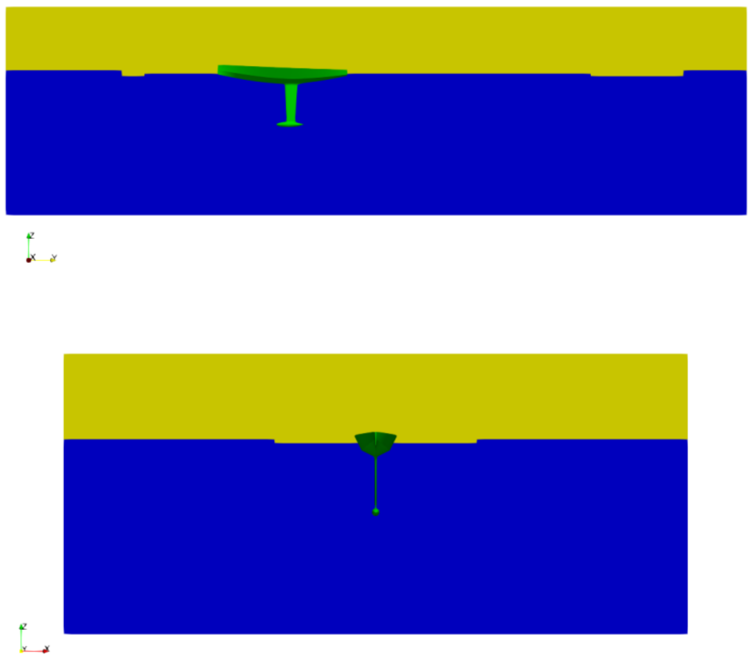




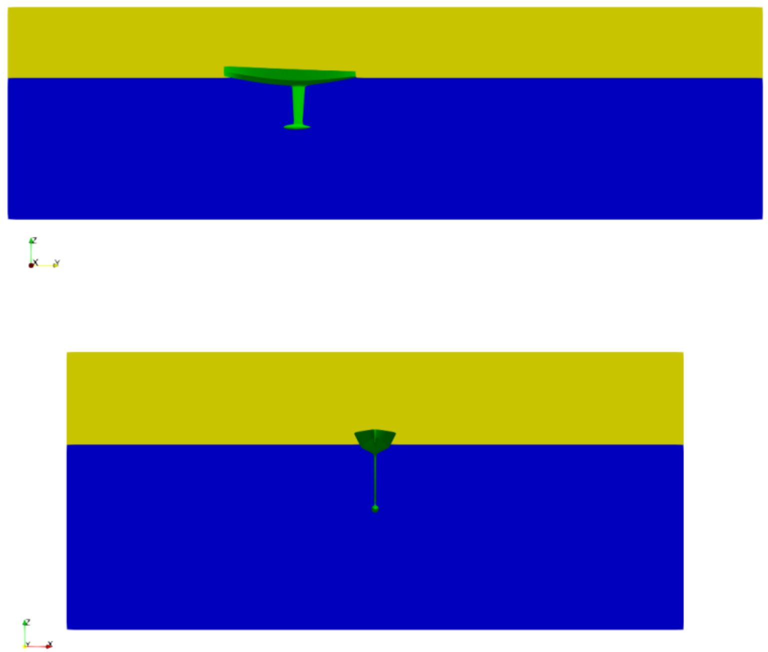
http://www.wolfdynamics.com/training/movingbodies/image11.gif

http://www.wolfdynamics.com/training/mphase/image45.gif

Volume-of-Fluid (VOF) governing equations for separated systems



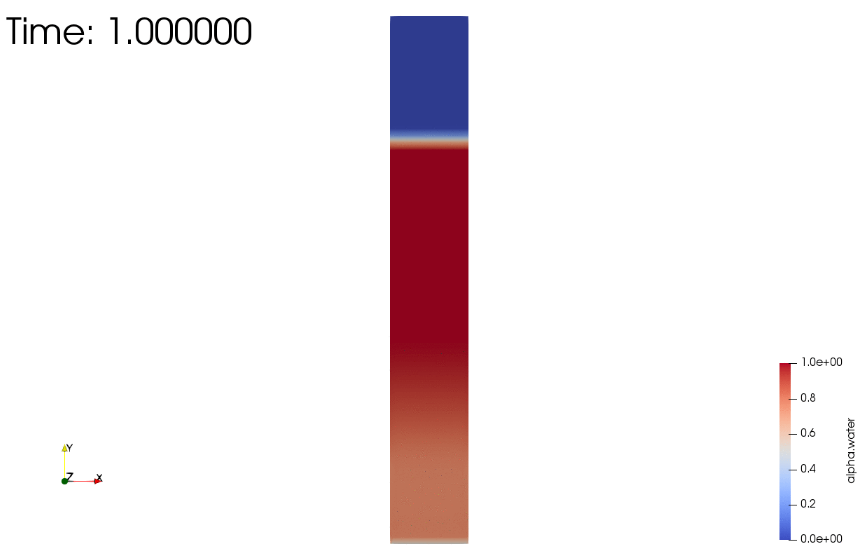
Non-uniform cell size at the interface

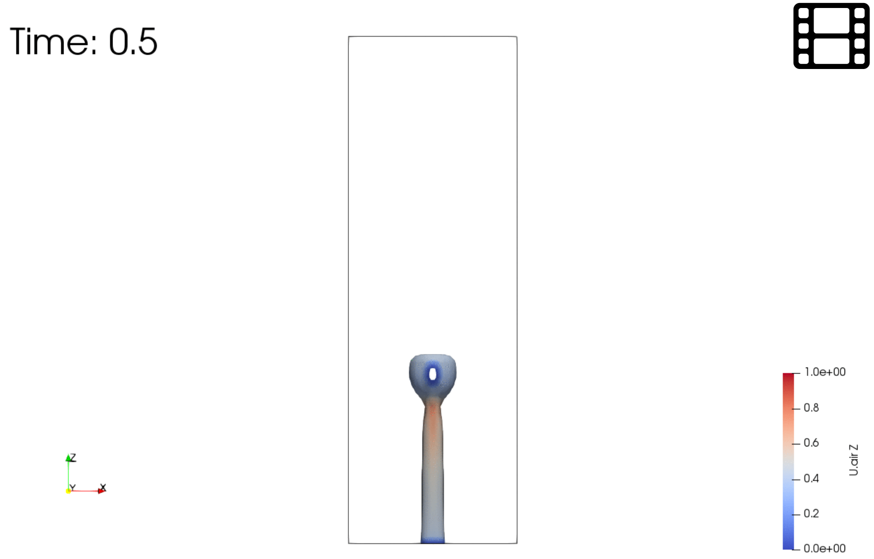


Uniform cell size at the interface

- A very important requirement of towing tank applications (and similar applications), is that the mesh should be uniform at the mesh interface.
- A non-uniform interface will generate unphysical perturbations due to the change of cell size and cell center location at the interface.

Eulerian-Eulerian governing equations for dispersed systems





http://www.wolfdynamics.com/training/mphase/image17.gif

http://www.wolfdynamics.com/training/mphase/image18.gif

Governing equations and interfacial momentum transfer models

Eulerian-Eulerian governing equations for dispersed systems

- The Eulerian-Eulerian approach solves the governing equations for each phase, it treats the phases as interpenetrating continua.
- The incompressible, isothermal governing equations can be written as follows,

Interface forces or momentum transfer

Surface tension

$$\frac{\partial \alpha_k \rho_k}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k) = 0$$

$$\frac{\partial (\alpha_k \rho_k \mathbf{U}_k)}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k \mathbf{U}_k) = -\nabla \cdot (\alpha_k \tau_k) - \alpha_k \nabla p + \alpha_k \rho_k \mathbf{g} + \mathbf{M}_k + f_\sigma + \mathbf{S}_k$$

$$\sum_k \alpha_k = 1.0 \qquad \rho_m = \sum_k \alpha_k \rho_k \qquad \mathbf{U}_m = \frac{\sum_k \alpha_k \rho_k \mathbf{U}_k}{\rho_m}$$

Source terms:

- Porous media
- Coriolis forces
- Centrifugal forces
- Mass transfer
- and so on …

Governing equations and interfacial momentum transfer models

Eulerian-Eulerian governing equations for dispersed systems

- The incompressible, isothermal governing equations with interface tracking can be written as follows,

Interface forces or momentum transfer

Surface tension

$$\frac{\partial \alpha_k \rho_k}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k) = 0$$

$$\frac{\partial (\alpha_k \rho_k \mathbf{U}_k)}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k \mathbf{U}_k) = -\nabla \cdot (\alpha_k \tau_k) - \alpha_k \nabla p + \alpha_k \rho_k \mathbf{g} + \mathbf{M}_k + f_\sigma + \mathbf{S}_k$$

$$\frac{\partial \alpha_k \rho_k}{\partial t} + \nabla \cdot \mathbf{U}_k \alpha_k \rho_k + \nabla \cdot (\mathbf{U}_r \alpha_k \rho_k (1 - \alpha_k)) = 0$$

Source terms:

- Porous media
- Coriolis forces
- Centrifugal forces
- Mass transfer
- and so on …

$$\sum_k \alpha_k = 1.0 \qquad \rho_m = \sum_k \alpha_k \rho_k \qquad \mathbf{U}_m = \frac{\sum_k \alpha_k \rho_k \mathbf{U}_k}{\rho_m}$$

Interfacial momentum transfer models

- Closure relations for the interface forces (models).
- Hereafter, we will describe the most common ones.

$$\mathbf{M}_l = -\mathbf{M}_g = \mathbf{M}_D + \mathbf{M}_L + \mathbf{M}_{VM} + \mathbf{M}_{TD}$$

Drag

$$\mathbf{M}_D = \frac{3}{4}\alpha_g \rho_l \frac{C_d}{d_b}|\mathbf{U}_l - \mathbf{U}_g|(\mathbf{U}_l - \mathbf{U}_g)$$

Lift

$$\mathbf{M}_L = \alpha_g \rho_l C_l (\mathbf{U}_l - \mathbf{U}_g) \times \nabla \times \mathbf{U}_l$$

Virtual Mass

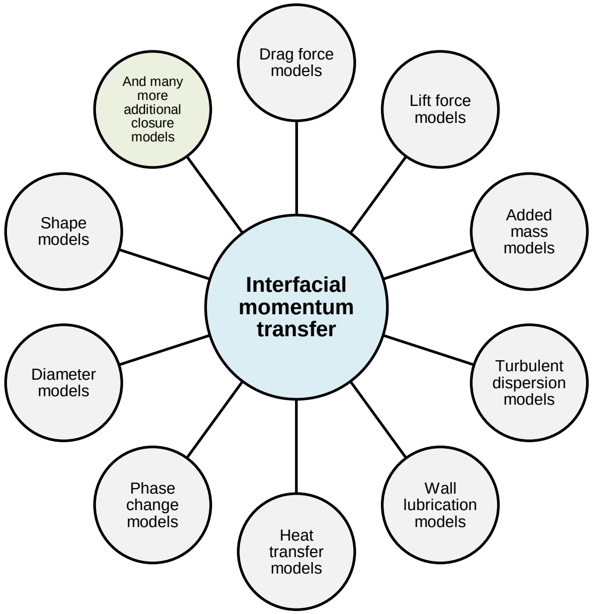
$$\mathbf{M}_{VM} = \alpha_g \rho_l C_{VM} (\mathbf{U}_l \left(\frac{D_l \mathbf{U}_l}{D_t} - \frac{D_g \mathbf{U}_g}{D_t} \right)$$

Turbulence Dispersion

$$\mathbf{M}_{TD} = \rho_l C_{TD} \kappa \nabla \alpha_g$$

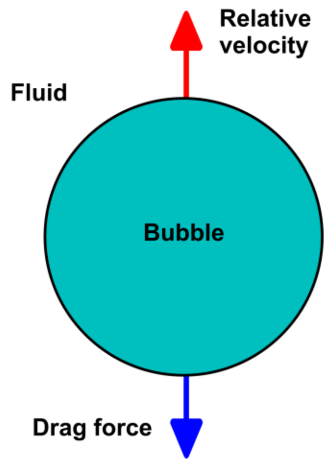
Interfacial momentum transfer models

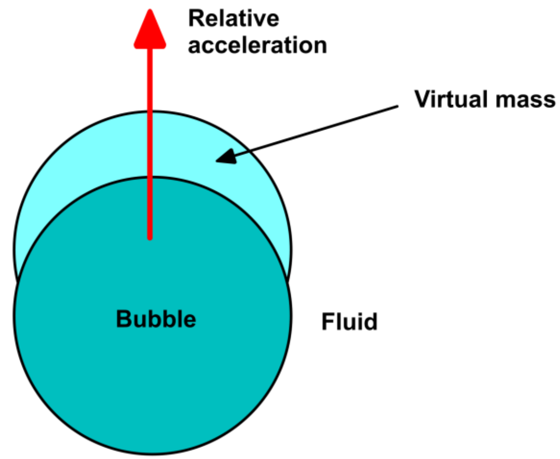
- To deal with interfacial momentum transfer, there are many models in the multiphase literature.
- We are going to briefly address a few of models implemented in OpenFOAM.
- We want to remind you that there is no universal model, it is up to you to choose the model that best fit the problem you are solving.
- Depending on the physics involved and the solver you are using, you will find different models and formulations
- You need to know the applicability and limitations of each model (refer to the literature).
- Most of the times, it is enough to use a drag force and virtual mass models.

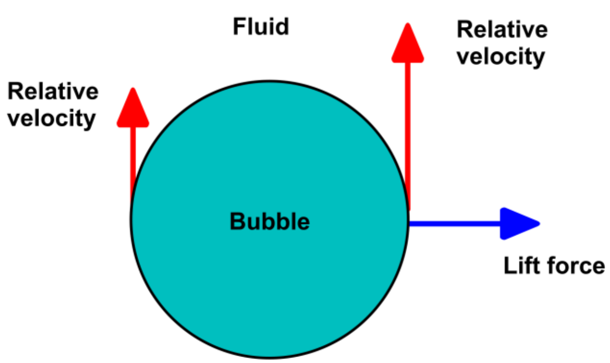


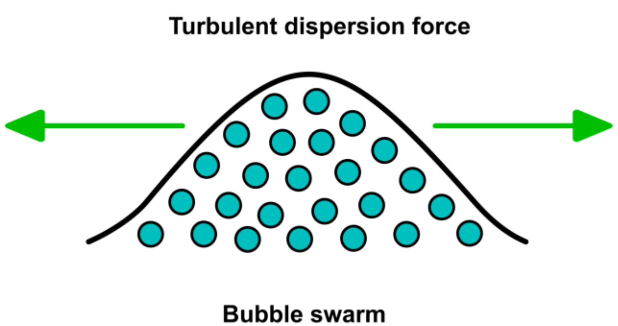
Governing equations and interfacial momentum transfer models

Interfacial momentum transfer models









Governing equations and interfacial momentum transfer models

Interfacial momentum transfer models

- The drag force can be written as follows,

$$\mathbf{M}_D = \frac{3}{4}\alpha_g \rho_l \frac{C_d}{d_b}|\mathbf{U}_l - \mathbf{U}_g|(\mathbf{U}_l - \mathbf{U}_g)$$

- Using Schiller and Naumann model for C_d,

$$C_d = \begin{cases} \dfrac{24}{Re}(1 + 0.15 Re^{0.687}) & Re \leq 1000 \\ 0.44 & Re > 1000 \end{cases}$$

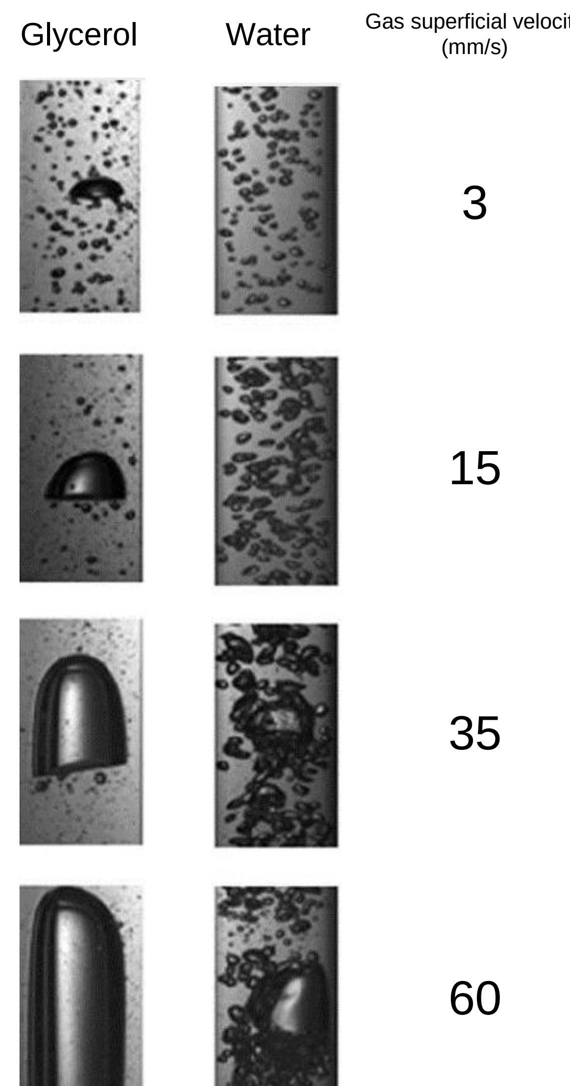
$$Re = \frac{d_b|\mathbf{U}_l - \mathbf{U}_g|}{\nu_l}$$

When the Re is less than 10 most of the drag models reduce to Stokes law

- For small and constant bubble size this model work very well.

Interfacial momentum transfer models

- Additional notes on drag models:
  - For bubble columns operating at low gas superficial velocities (< 5 cm/s), drag models using mean bubble size approach work fine.
  - For bubble columns operating at higher gas superficial velocities (> 5 cm/s), bubble breakup and coalesce dominate and bubble size in no longer uniform and mean bubble size approach may not be adequate.
  - For gas superficial velocities higher than > 5 cm/s, it is recommended to use population balance models.
  - When the bubble size is small (< 1 mm in water), bubble size is approximately spherical.
  - When bubble size is large (> 18 mm in water), bubble is approximately a spherical cap.
  - For intermediate bubble sizes, bubbles exhibit complex and random shapes.



Interfacial momentum transfer models

- The lift force can be written as follows,

$$\mathbf{M}_L = \alpha_g \rho_l C_l (\mathbf{U}_l - \mathbf{U}_g) \times \nabla \times \mathbf{U}_l$$

- Using Tomiyama model for C_l,

$$C_l = \begin{cases} min \begin{cases} 0.2888\, tanh(0.121 Re) \\ 0.00105 Eo^3 - 0.0159 Eo^2 - 0.0204 Eo + 0.474 \end{cases} & Eo < 4 \\ 0.00105 Eo^3 - 0.0159 Eo^2 - 0.0204 Eo + 0.474 & 4 \leq Eo \leq 10 \\ -0.29 & Eo > 10 \end{cases}$$

$$Re = \frac{d_b |\mathbf{U}_l - \mathbf{U}_g|}{\nu_l} \qquad Eo = \frac{\Delta \rho g d_g^2}{\sigma}$$

Interfacial momentum transfer models

- Additional notes on lift models:
  - The lift force depends on the bubble diameter, the relative velocity between the phases, and the vorticity.

$$\mathbf{M}_L = \alpha_g \rho_l C_l (\mathbf{U}_l - \mathbf{U}_g) \times \nabla \times \mathbf{U}_l$$

  - The lift coefficient is often constant for Reynolds number less than 500. If this is the case, it can be set to 0.5.
  - Lift forces are responsible for inhomogeneous radial distribution of the dispersed phase.
  - The Tomiyana model probably is the most general model. It is suitable for all shape and size of bubbles and drops.

Interfacial momentum transfer models

- The turbulence dispersion force can be written as follows,

$$\mathbf{M}_{TD} = \rho_l C_{TD} \kappa \nabla \alpha_g$$

- Using constant coefficient turbulence dispersion model C_{TD},

$$C_{TD} = 1$$

Interfacial momentum transfer models

- Additional notes on turbulence dispersion models:
    - The turbulence dispersion force accounts for an interaction between turbulent eddies and particles.
    - Results in a turbulent dispersion and homogenization of the dispersed phase distribution.
    - The simplest way to model turbulent dispersion is by assuming gradient transport as follows,

$$\mathbf{M}_{TD} = \rho_l C_{TD} \kappa \nabla \alpha_g$$

    - Many models are available.
    - For medium sized bubbles the Lopez model is a good choice.
    - For small sized bubbles, the Burns model is recommended.

Interfacial momentum transfer models

- The virtual mass force can be written as follows,

$$\mathbf{M}_{VM} = \alpha_g \rho_l C_{VM} \mathbf{U}_l \left(\frac{D_l \mathbf{U}_l}{D_t} - \frac{D_g \mathbf{U}_g}{D_t} \right)$$

- Using constant coefficient virtual mass model C_{VM},

$$C_{VM} = 0.5$$

Interfacial momentum transfer models in OpenFOAM

- Diameter models in OpenFOAM (not all of them are listed):
  - IATE
  - Constant
  - Isothermal
  - linearTsub

- These models are used to define the diameter of the bubbles or droplets.
- Most of the time is fine to use the constant diameter model.
- You can find the source code of the models in the following directories:

`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/phaseSystem/diameterModels`

Interfacial momentum transfer models in OpenFOAM

- Aspect ratio models in OpenFOAM (not all of them are listed):
  - Tomiyama
  - VakhrushevEfremov
  - Wellek
  - constant

- These models are used to define the aspect ratio of the bubbles or droplets.
- The constant model with Eo equal to 1.0 is equivalent to perfect spherical bubbles.
- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/aspectRatioModels
```

Interfacial momentum transfer models in OpenFOAM

- Drag models in OpenFOAM (not all of them are listed):
  - Ergun
  - Gibilaro
  - GidaspowErgunWenYu
  - GidaspowSchillerNaumann
  - IshiiZuber
  - Lain
  - SchillerNaumann
  - SyamlalOBrien
  - TomiyamaAnalytic
  - TomiyamaCorrelated
  - WenYu
  - segregated

- These models are used to compute the drag forces on the bubbles or droplets.
- Most of the times is fine to use the Schiller and Naumann model.
- The Tomiyama model is a good alternative for deforming models.
- If the bubbles do not deform (they are rigid), the Syamlal model is the best option.
- You can find the source code of the models in the following directories:

`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/dragModels`

Interfacial momentum transfer models in OpenFOAM

- Virtual mass models in OpenFOAM (not all of them are listed):
    - Lamb
    - constantCoefficient
    - none

- Virtual mass force represents the force due to inertia of the dispersed phase due to acceleration.
- Most of the times is fine to use the constant coefficient with a value of 0.5.
- You can find the source code of the models in the following directories:

`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/virtualMassModels`

Interfacial momentum transfer models in OpenFOAM

- Lift models in OpenFOAM (not all of them are listed):
  - LegendreMagnaudet
  - Moraga
  - Tomiyama
  - constantCoefficient
  - none

- The lift force is mainly due to velocity gradients in the continuous phase. The lift force is equivalent to lateral forces.
- In most cases, the lift force is insignificant compared to the drag force, so there is no need to model this force.
- Tomiyama model is a good choice if you want to use a model.
- For low Reynolds number (less than 500) you can use a constant coefficient model with a value of 0.5.
- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/liftModels
```

Interfacial momentum transfer models in OpenFOAM

- Wall lubrication models in OpenFOAM (not all of them are listed):
    - Antal
    - Frank
    - Tomiyama
    - none

- Wall lubrication models take into account the repulsive effect, which bubbles are exerted to in the vicinity of the wall of the column as a consequence of an asymmetric incident flow near the wall boundary layer.
- Most of the times it is not needed to model this force.
- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/wallLubricationModels
```

Interfacial momentum transfer models in OpenFOAM

- Turbulent dispersion models in OpenFOAM (not all of them are listed):
  - Burns
  - Gosman
  - LopezDeBertodano
  - constantCoefficient
  - None

- The turbulent dispersion models account for the interaction between turbulent eddies and particles.
- Most of the times it is not needed to model this force.
- Burns model with a constant coefficient equal to 1.0 is a good choice if you want to use a model.
- You can find the source code of the models in the following directories:

`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/turbulentDispersionModels`

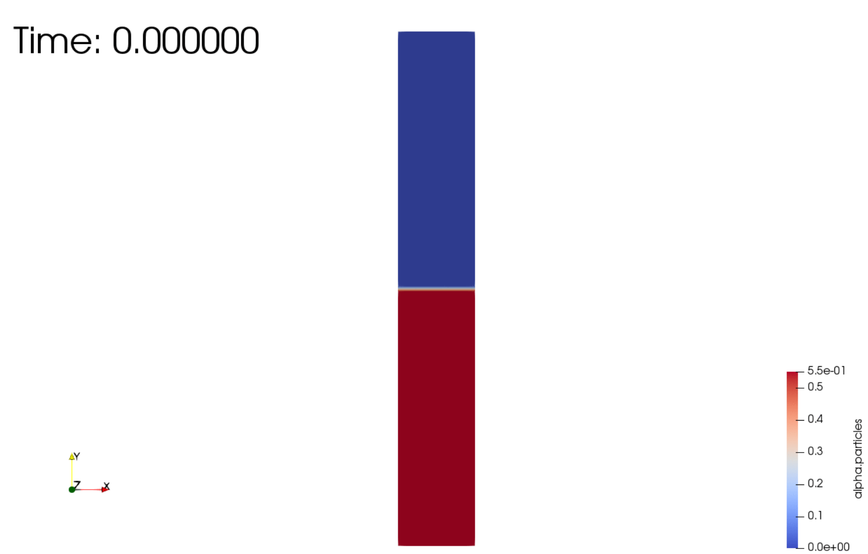
Interfacial momentum transfer models in OpenFOAM

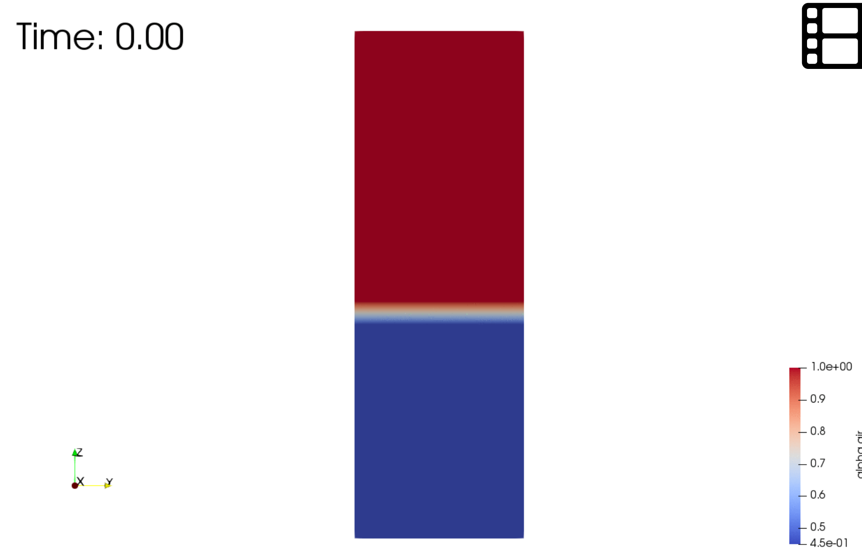
- Heat transfer models in OpenFOAM:
  - constantNu
  - Gunn
  - RanzMarshall
  - sphericalHeatTransfer
  - none

- These models are used to compute the heat transfer from idealized geometries.
- You can find the source code of the models in the following directories:

`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels/heatTransferModels`

Eulerian-Eulerian governing equations for dispersed systems of gas-solid or liquid-solid (Eulerian-Granular KTGF)





http://www.wolfdynamics.com/training/mphase/image19.gif

http://www.wolfdynamics.com/training/mphase/image20.gif

Eulerian-Eulerian governing equations for dispersed systems of gas-solid or liquid-solid (Eulerian-Granular KTGF)

- The Eulerian-Granular KTGF (kinetic theory of granular flow), is based on the kinetic theory of granular flow (analogous to kinetic theory of gases).
- In the E-G KTGF the liquid phase is solved as in the Eulerian-Eulerian method and the solid phase is solved using appropriate closure relations.
- The incompressible, isothermal governing equations can be written as follows,

$$\frac{\partial \alpha_k \rho_k}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k) = 0$$

$$\gamma_k = \frac{\alpha_{s,k}}{\sum_k \alpha_{s,k}}$$

Momentum transfer

$$\frac{\partial (\alpha_k \rho_k \mathbf{U}_k)}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k \mathbf{U}_k) = -\nabla \cdot (\alpha_k \tau_k) - \alpha_k \nabla p - \gamma_k \nabla p_k + \alpha_k \rho_k \mathbf{g} + \mathbf{M}_k$$

Stress tensor for solid and fluid

Shared pressure

Disperse phase pressure

Eulerian-Eulerian governing equations for dispersed systems of gas-solid or liquid-solid (Eulerian-Granular KTGF)

- With the following closure relations (transport equation of granular temperature or velocity fluctuations).

$$\frac{3}{2}\left[\frac{\partial \alpha_s \rho_s \Theta_s}{\partial t} + \nabla \cdot (\alpha_s \rho_s \Theta_s \mathbf{U}_s)\right] = (-\nabla p_s \mathbf{I} + \tau_s) : \nabla \mathbf{U}_s + \nabla \cdot (k_s \nabla \Theta_s) - \gamma_s + J_s$$

- Granular temperature

$$\Theta_s = \frac{1}{3}\mathbf{U}_{fs}^2 \longrightarrow$$ Particle fluctuating velocity

- Solid stress tensor

$$\tau_s = -p_s \mathbf{I} + 2\alpha_s \mu_s \frac{1}{2}(\nabla \mathbf{U}_s + \nabla \mathbf{U}_s^T) + \alpha_s(\lambda_s - \frac{2}{3}\mu_s)\nabla \cdot \mathbf{U}_s \mathbf{I}$$

- Equation of state for the dispersed phase

$$p_s = \rho_s \alpha_s \Theta_s + 2\rho_s \alpha_s^2 g_0 \Theta_s (1 + e_s)$$

Eulerian-Eulerian governing equations for dispersed systems of gas-solid or liquid-solid (Eulerian-Granular KTGF)

- Plus, the following additional terms that require closure relations (models),

λ_s	Solid bulk viscosity
μ_s	Solid shear viscosity
τ_f	Frictional stresses
g_0	Radial distribution function
κ	Conductivity of granular energy
e_s	Coefficient of restitution of colliding particles
γ_s	Dissipation of granular energy
J_s	Exchange of fluctuating energy between the phases
$\mathbf{M}_k$	Momentum transfer models (usually only drag for gas-solid)

For a complete derivation of the governing equation and closure relations, refer to:
Derivation, Implementation, and Validation of Computer Simulation Models for Gas-Solid Fluidized Bed
B. van Wachem. PhD Thesis. 2000, TUDelft.

Governing equations and interfacial momentum transfer models

Eulerian-Eulerian governing equations for dispersed systems of gas-solid or liquid-solid (Eulerian-Granular KTGF)

- In the transport equation of granular temperature

$$\frac{3}{2}\left[\frac{\partial \alpha_s \rho_s \Theta_s}{\partial t} + \nabla \cdot (\alpha_s \rho_s \Theta_s \mathbf{U}_s)\right] = (-\nabla p_s \mathbf{I} + \tau_s) : \nabla \mathbf{U}_s + \nabla \cdot (k_s \nabla \Theta_s) - \gamma_s + J_s$$

  - The first term on the RHS represents the creation of fluctuating energy due to shear in the particle phase.
  - The second term on the RHS represents the diffusion of fluctuating energy along gradients in the Θ_s.
  - γ_s represents the dissipation due to inelastic particle-particle collisions.
  - J_s represents the dissipation or creation of granular energy resulting from the working of the fluctuating force exerted by the gas through the fluctuating velocity of the particles.

Closure relations for Eulerian-Granular KTGF in OpenFOAM

- In OpenFOAM, there are many closure relations and interfacial models for Eulerian-Granular kinetic theory of granular flows.
- We want to remind you that there is no universal model, it is up to you to choose the model that best fit the problem you are solving.
- Depending on the physics involved and the solver you are using, you will find different models and formulations
- You need to know the applicability and limitations of each model (refer to the literature).

Interfacial momentum transfer models in OpenFOAM

- Conductivity models in OpenFOAM (not all of them are listed):
    - Gidaspow
    - HrenyaSinclair
    - Syamlal

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/multiphaseCompressibleMomentumTransportModels/
kineticTheoryModels/conductivityModel
```

Interfacial momentum transfer models in OpenFOAM

- Frictional stress models in OpenFOAM (not all of them are listed):
    - JohnsonJackson
    - JohnsonJacksonSchaeffer
    - Schaeffer

    Note: there is a Johnson and Jackson boundary condition for walls

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/multiphaseCompressibleMomentumTransportModels/
kineticTheoryModels/frictionalStressModel
```

Interfacial momentum transfer models in OpenFOAM

- Granular pressure models in OpenFOAM (not all of them are listed):
    - Lun
    - SyamlalRogersOBrien

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/multiphaseCompressibleMomentumTransportModels/
kineticTheoryModels/granularPressureModel
```

Interfacial momentum transfer models in OpenFOAM

- Radial distribution function models in OpenFOAM (not all of them are listed):
    - CarnahanStarling
    - LunSavage
    - SinclairJackson

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/multiphaseCompressibleMomentumTransportModels/
kineticTheoryModels/radialModel
```

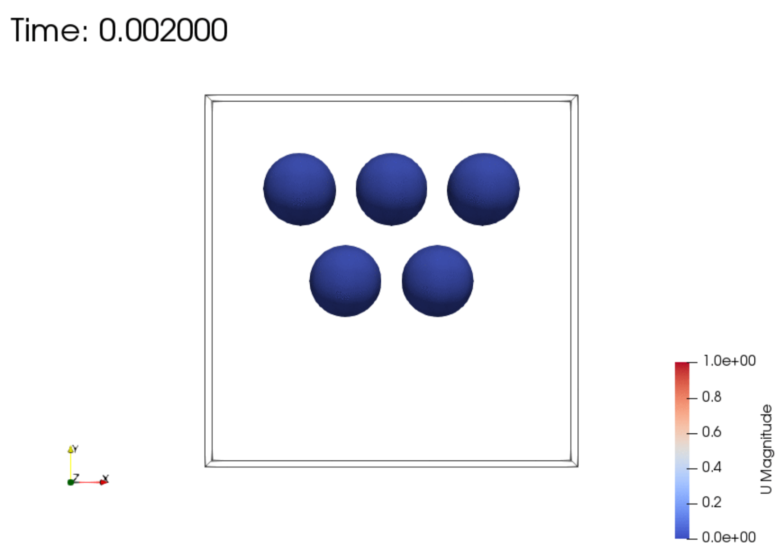
Interfacial momentum transfer models in OpenFOAM

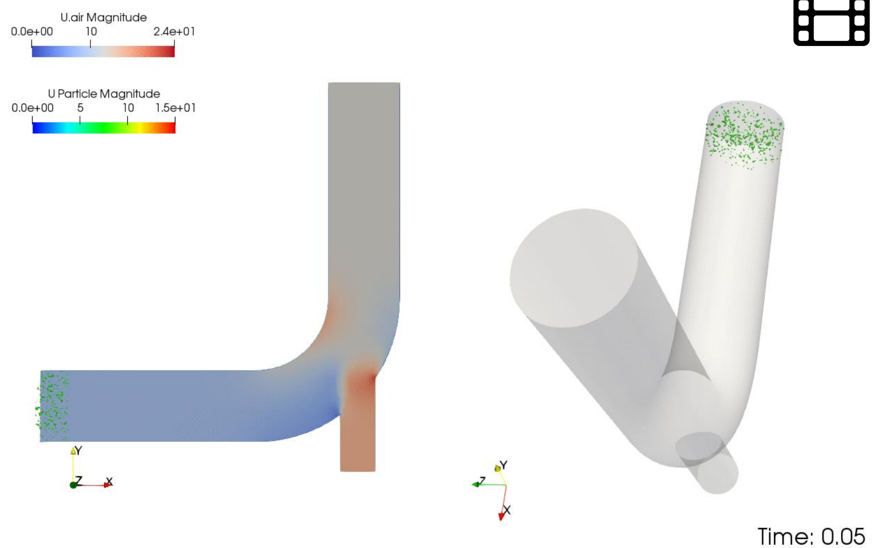
- Viscosity models in OpenFOAM (not all of them are listed):
  - Gidaspow
  - HrenyaSinclair
  - Syamlal
  - none

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/multiphaseCompressibleMomentumTransportModels/
kineticTheoryModels/viscosityModel
```

Eulerian-Lagrangian governing equations





http://www.wolfdynamics.com/training/mphase/image21.gif

http://www.wolfdynamics.com/training/mphase/image22.gif

Eulerian-Lagrangian governing equations

- In the Eulerian-Lagrangian framework, the continuous phase is solved in a Eulerian reference system and the particles or dispersed phase is solved in a Lagrangian reference system.
- The governing equations can be written as follows,

$$m\frac{d\mathbf{U}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{pressure} + \mathbf{F}_{virtual\,mass} + \mathbf{F}_{other}$$

$$+$$

Any of the Eulerian formulations (single or multi-phase)

Eulerian-Lagrangian governing equations

- In this formulation the secondary phase is treat as discrete particles dispersed in the continuous fluid.
- The particles can be smaller or larger than the grid size.
- The particles can be transported passively, or they can be coupled with the fluid governing equations. That is, the particles can modify the fluid field.
- This formulation accounts for particle interaction and mass transfer.
- The particles can interact with the boundaries and have a fate. They can escape, bounce, stick, or form a wall film.
- If you want, you can add angular momentum to the formulation.
- Depending on the number or particles tracked, this type of approach can be computational expensive.

Eulerian-Lagrangian governing equations

- There are two approaches to model Eulerian-Lagrangian systems with hydrodynamic coupling in OpenFOAM
- The DPM or Dense Particle Flows approach, which includes the effect of the particulate volume fraction on the continuous phase, suitable for dense particle flow simulation.
- The MPPIC or Multiphase Particle-in-Cell method* for collisional exchange. In this approach, particle-particle interactions are represented by models which utilize mean values calculated on the Eulerian mesh.
- The MPPIC is suitable for dense to dilute regimes.

* P. J. O'Rourke et al., Chemical Engineering Science 64:1784-1797, 2009

Governing equations and interfacial momentum transfer models

Eulerian-Lagrangian governing equations

$$m\frac{d\mathbf{U}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{pressure} + \mathbf{F}_{virtual\,mass} + \mathbf{F}_{other}$$

$$+$$

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

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla p + \nabla \cdot \tau + \rho g + \rho S$$

Governing equations and interfacial momentum transfer models

Eulerian-Lagrangian governing equations

$$m\frac{d\mathbf{U}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{pressure} + \mathbf{F}_{virtual\,mass} + \mathbf{F}_{other}$$

$$+$$

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

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla p + \nabla \cdot \tau + \rho g + f_\sigma + \rho S$$

$$\frac{\partial \gamma}{\partial t} + \nabla \cdot \mathbf{U}\gamma + \nabla \cdot (\mathbf{U}_r \gamma (1-\gamma)) = 0$$

Governing equations and interfacial momentum transfer models

Eulerian-Lagrangian governing equations

$$m\frac{d\mathbf{U}}{dt} = \mathbf{F}_{drag} + \mathbf{F}_{pressure} + \mathbf{F}_{virtual\,mass} + \mathbf{F}_{other}$$

$$+$$

$$\frac{\partial \alpha_k \rho_k}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k) = 0$$

$$\frac{\partial (\alpha_k \rho_k \mathbf{U}_k)}{\partial t} + \nabla \cdot (\alpha_k \rho_k \mathbf{U}_k \mathbf{U}_k) = -\nabla \cdot (\alpha_k \tau_k) - \alpha_k \nabla p + \alpha_k \rho_k \mathbf{g} + \mathbf{M}_k + f_\sigma + \mathbf{S}_k$$

$$\frac{\partial \alpha_k \rho_k}{\partial t} + \nabla \cdot \mathbf{U}_k \alpha_k \rho_k + \nabla \cdot (\mathbf{U}_r \alpha_k \rho_k (1 - \alpha_k)) = 0$$

Eulerian-Lagrangian models in OpenFOAM

- In OpenFOAM, there are many models for Eulerian-Lagrangian methods.
- We want to remind you that there is no universal model, it is up to you to choose the model that best fit the problem you are solving.
- Depending on the physics involved and the solver you are using, you will find different models and formulations
- You need to know the applicability and limitations of each model (refer to the literature).

Eulerian-Lagrangian models in OpenFOAM

- Dispersion models in OpenFOAM (not all of them are listed):
    - GradientDispersionRAS
    - none
    - StochasticDispersionRAS

- You can find the source code of the models in the following directories:

`OpenFOAM-9/src/lagrangian/parcelTurbulence/submodels/Kinematic/DispersionModel/`

Eulerian-Lagrangian models in OpenFOAM

- Particle injection models in OpenFOAM (not all of them are listed):
  - CellZoneInjection
  - ConeInjection
  - FieldActivatedInjection
  - KinematicLookupTableInjection
  - ManualInjection
  - NoInjection
  - PatchFlowRateInjection
  - PatchInjection

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/src/lagrangian/parcel/submodels/Momentum/InjectionModel/
OpenFOAM-9/src/lagrangian/parcel/submodels/Reacting/InjectionModel/
OpenFOAM-9/src/lagrangian/parcel/submodels/Spray/AtomizationModel/
```

Eulerian-Lagrangian models in OpenFOAM

- Particle force models in OpenFOAM (not all of them are listed):
  - Drag
  - Gravity
  - Lift
  - NonInertialFrame
  - Paramagnetic
  - PressureGradient
  - SRF
  - virtualMass
  - BrownianMotion
  - Source terms (forceSuSp)

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/src/lagrangian/parcel/submodels/Momentum/ParticleForces
OpenFOAM-9/src/lagrangian/parcelTurbulence/submodels/Thermodynamic/ParticleForces
```

Eulerian-Lagrangian models in OpenFOAM

- Patch interaction models in OpenFOAM (not all of them are listed):
  - LocalInteraction
  - NoInteraction
  - PatchInteractionModel
  - Rebound
  - StandardWallInteraction

- Patch interaction type in OpenFOAM (particles fate):
  - rebound
  - stick
  - escape

- You can find the source code of the models in the following directories:

```
OpenFOAM-9/src/lagrangian/parcel/submodels/Momentum/PatchInteractionModel
```

Eulerian-Lagrangian models in OpenFOAM

- Particle distribution models in OpenFOAM (not all of them are listed):
  - exponential
  - fixedValue
  - general
  - massRosinRammler
  - multiNormal
  - normal
  - RosinRammler
  - uniform

- You can find the source code of the models in the following directories:

`OpenFOAM-9/src/lagrangian/distributionModels`

Eulerian-Lagrangian models in OpenFOAM

- Other Lagrangian particles models:
  - MPPIC models (packing, collisions, damping, isotropy):
    - `OpenFOAM-9/src/lagrangian/parcel/submodels/MPPIC/`
  - Reacting and phase change models:
    - `OpenFOAM-9/src/lagrangian/parcel/submodels/Reacting/`
  - Spray models:
    - `OpenFOAM-9/src/lagrangian/parcel/submodels/Spray`
  - Function objects for particle clouds (cloudFunctions):
    - `OpenFOAM-9/src/lagrangian/parcel/submodels/CloudFunctionObjects/`

Final remarks

- In all the previous formulations we wrote the governing equations in their laminar form.
- To add turbulence, you only need to do the proper averaging (Reynolds or Favre) or use a filtering technique for LES simulations.
- In the Eulerian-Eulerian formulation for dispersed systems, the stress term in the new equations will become:

$$\tau_k = -\mu_{eff,k}(\nabla \mathbf{U}_k + \nabla \mathbf{U}_k^T - \frac{2}{3}\mathbf{I}(\nabla \cdot \mathbf{U}_k))$$

- You will need to use the proper closure relation for finding $\mu_{eff,k}$
- In the Eulerian-Eulerian formulation for separated systems (VOF), only one phasic set of closure relations for turbulence models are solved (the Reynolds or Leonard stress tensor is the same as for single-phase flows).

Final remarks

- In all the previous formulations we wrote the incompressible, isothermal governing equations.
- You can add the energy equation, which is written as follows

$$\frac{\partial \rho c_p T}{\partial t} + \nabla \cdot (\rho c_p \mathbf{U} T) = -\nabla \cdot q + \tau : \nabla \mathbf{U} + S$$

- By adding the energy equation, you can model mass transfer between phases.
- By adding thermal effects, you can model boiling, melting, freezing, sublimation, condensation, evaporation.
- You can also model cavitation and flashing. Have in mind that they are driven by local pressure effects.

Final remarks

Eulerian-Eulerian (VOF)	Eulerian-Eulerian (Dispersed systems)	Eulerian-Lagrangian
• Non-interpenetrating continua. • Continuous phases: Eulerian. • Fluid properties are written on either side of the interface (no averaging). • Solves one single set of PDEs: mass, momentum, energy.	• Interpenetrating continua. • Continuous phase: Eulerian. • Dispersed phase: Eulerian. • Phase-weighted averages. • Solves PDEs for all phases (including interphase transfer terms): mass, momentum, energy.	• Continuous phase: Eulerian. • Dispersed phase: Lagrangian. • Solves ODEs for particle tracking (for every single particle). • Solves a set of PDEs for the continuous phase: mass, momentum, energy. • Phase interaction terms (including interphase transfer terms).

Final remarks

Eulerian-Lagrangian approach advantages	Eulerian-Lagrangian approach drawbacks
• Complete and detailed information about behavior and residence time of individual particles. • Relative cheaper than the Eulerian-Eulerian approach for a wide range of particle sizes. • Better detail for drag, heat and mass transfer.	• For large volume fractions the model is not very accurate. • Can be very expensive if it is necessary to track a large number of particles • Difficult to get smooth information about local values of volume fractions, velocities, forces on walls, and so on.

Roadmap

1. ~~Introduction to multiphase flows~~
2. ~~Modeling approaches for multiphase flows~~
3. ~~Governing equations and interfacial momentum transfer models~~
4. **Multiphase solvers in OpenFOAM**
5. Selecting physical properties, phase interaction, and advanced models
6. Final remarks – Tips and tricks
7. Additional tutorials

Multiphase solvers in OpenFOAM

- We will only address the most general solvers and closure models.
- OpenFOAM comes with many solvers and models that can address a wide physics.
- When dealing with multiphase flows in OpenFOAM, you can use VOF, Eulerian-Eulerian, Eulerian-Eulerian with VOF, and Eulerian-Lagrangian methods.
- The solution methods can account for turbulence models, interface momentum transfer models, mass transfer models, particle interaction models and so on.
- It is also possible to add source terms, deal with moving bodies or use adaptive mesh refinement.
- As turbulence modeling is generic in many solvers (laminar, RANS, LES), we will not address it. We will only address turbulence modeling in those solvers that use hardwired models.

Multiphase solvers in OpenFOAM

- You will find the source code of all the multiphase solvers in the directory:
    - **`OpenFOAM-9/applications/solvers/multiphase`**

- You will find the source code all the particle tracking solvers in the directory:
    - **`OpenFOAM-9/applications/solvers/lagrangian`**

- You will find the source code all the combustion solvers in the directory:
    - **`OpenFOAM-9/applications/solvers/combustion`**

Multiphase solvers in OpenFOAM

- Inside each sub-directory you will find the source code for each solver.
- If you open the main file in the solver directory (the *.C file with the same name of the directory, e.g., *interFoam.C*), you will find a short description of the solver.
- For example, if you go to the sub-directory **multiphase/interFoam** and open the file *interFoam.C*, you will find this description of the solver in the header of the file:

```
Description

    Solver for 2 incompressible, isothermal immiscible fluids using a VOF
    (volume of fluid) phase-fraction based interface capturing approach,
    with optional mesh motion and mesh topology changes including adaptive
    re-meshing.
```

Multiphase solvers in OpenFOAM

- In OpenFOAM, you will find the following multiphase solvers:

  - cavitatingFoam
  - compressibleInterFoam
    - compressibleInterFilmFoam
  - compressibleMultiphaseInterFoam
  - driftFluxFoam
  - interFoam
    - interMixingFoam
  - interPhaseChangeFoam
  - interPhaseChangeFoam
  - multiphaseEulerFoam
  - multiphaseInterFoam
  - potentialFreeSurfaceFoam
  - twoLiquidMixingFoam

Multiphase solvers in OpenFOAM

- In OpenFOAM, you will find the following particle-tracking solvers or general solvers with support to particle clouds:

  - reactingFoam
  - denseParticleFoam
  - particleFoam
  - rhoParticleFoam
  - buoyantReactingFoam
  - engineFoam

Note:

Starting form OpenFOAM 9,

- MPPICFoam and DPMFoam have been replaced by denseParticleFoam.
- reactingParcelFoam have ben replaced by buoyantReactingFoam.
- coalChemistryFoam, simpleReactingParcelFoam, simpleReactingParticleFoam, and sprayFoam have ben replaced by reactingFoam.

Multiphase solvers in OpenFOAM

- In OpenFOAM, you will find the following combustion solvers:
    - chemFoam
    - coldEngineFoam
    - PDRFoam
    - reactingFoam
        - buoyantReactingFoam
        - engineFoam
    - XiFoam
        - XiEngineFoam

Closure models for multiphase solvers in OpenFOAM

- The source code of the interfacial momentum transfer models of the multiphaseEulerFoam solver is located in the directory,
  - `OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialModels`
  - `OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/interfacialCompositionModels/`

- In this directory, you will find the following sub-directories containing the implemented models (not all of them are listed),
  - aspectRatioModels
  - dragModels
  - heatTransferModels
  - liftModels
  - swarmCorrections
  - diffusiveMassTransferModels
  - surfaceTensionModels
  - turbulentDispersionModels
  - virtualMassModels
  - wallDependentModel
  - wallLubricationModels
  - phaseTransferModels
  - interfaceCompositionModels
  - saturationModels

Closure models for multiphase solvers in OpenFOAM

- The source code of the kinetic theory of granular flows of the multiphaseEulerFoam solver is located in the directory,
  - **`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/multiphaseCompressibleMomentumTransportModels/`**

- In this directory, you will find the following sub-directories containing the implemented models,
  - kineticTheoryModels
  - phasePressureModels
  - derivedFvPatchFields (boundary conditions)

Closure models for multiphase solvers in OpenFOAM

- Additionally, in the directory,
  - **`OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/phaseSystems`**

- You will find the following sub-directories containing the source code of the following models,
  - alphaContactAngle
  - BlendedInterfacialModel
  - diameterModels
  - phaseModel
  - phasePair
  - PhaseSystems
  - populationBalanceModel

Closure models for multiphase solvers in OpenFOAM

- **A short note regarding the solver multiphaseEulerFoam in OpenFOAM 8 (and newer).**
  - In OpenFOAM 7 (and earlier versions), there were two solvers to deal with dispersed flows, namely, multiphaseEulerFoam and twoPhaseEulerFoam.
  - In OpenFOAM 8 (and newer), these two solvers were merged into one single solver, namely, multiphaseEulerFoam.
  - The new solver, is more complex with very complicated input dictionaries.
  - At the same time, it is more general.
  - The description of the solver multiphaseEulerFoam in the source code of OpenFOAM 9 is the following one,

> Solver for a system of any number of compressible fluid phases with a common pressure, but otherwise separate properties. The type of phase model is run time selectable and can optionally represent multiple species and in-phase reactions. The phase system is also run time selectable and can optionally represent different types of momentum, heat and mass transfer.

Closure models for multiphase solvers in OpenFOAM

- **A short note regarding population balance modeling (PBM).**
    - Population balance modeling (PBM) is an approach used to describe time-dependent changes of the multidimensional particle property distribution due to various effects occurring in a system.
    - PBM, more generally, define how populations of separate entities develop in specific properties over time.
    - The behavior of the population of particles or bubbles can be due to breakage, nucleation, agglomeration, continuous growth, and attrition, among many mechanisms.
    - PBM is governed by the population balance equations (PBEs).
    - PBEs are a set of Integro-partial differential equations which gives the mean-field behavior of a population of particles from the analysis of behavior of single particle in local conditions

Closure models for multiphase solvers in OpenFOAM

- **A short note regarding population balance modeling (PBM).**
  - OpenFOAM comes with its own PBM library, which is part of the solver multiphaseEulerFoam.
  - The source code of the PBM library is located in the directory,
    - `OpenFOAM-9/applications/solvers/multiphase/multiphaseEulerFoam/phaseSystems/`
  - The PBM functionalities in OpenFOAM can be explored by taking a look at the following tutorials,
    - `$FOAM_TUTORIALS/multiphase/multiphaseEulerFoam/RAS/bubblePipe`
    - `$FOAM_TUTORIALS/multiphase/multiphaseEulerFoam/laminar/titaniaSynthesis`
    - `$FOAM_TUTORIALS/multiphase/multiphaseEulerFoam/RAS/wallBoilingPolydisperse`
  - More information about PBM in OpenFOAM can be found at the following links,

    https://openfoam.org/guides/population-balance-modelling/

    https://openfoam.org/guides/population-balance-openfoam/

Closure models for multiphase solvers in OpenFOAM

- The source code of the thermo-physical models, transport models and turbulence models for the multiphase solvers is located in the following directories:

  - **`OpenFOAM-9/src/thermophysicalModels`**
  - **`OpenFOAM-9/src/transportModels`**
  - **`OpenFOAM-9/src/MomentumTransportModels`**

- In the sub-directories **`MomentumTransportModels/phaseCompressible`** and **`MomentumTransportModels/phaseInompressible`**, you will find the turbulence models for dispersed systems.

Closure models for multiphase solvers in OpenFOAM

- The source code of the surface tension models for the multiphase solvers is located in the following directory:
  - **`OpenFOAM-9/src/twoPhaseModels/interfaceProperties/surfaceTensionModels/`**

- The source code of the wall contact models (wall contact angle) for the multiphase solvers is located in the following directory:
  - **`OpenFOAM-9/src/twoPhaseModels/twoPhaseProperties/alphaContactAngle/`**

Closure models for multiphase solvers in OpenFOAM

- The source code of the closure models for the Eulerian-Lagrangian multiphase solvers is located in the following directory,
  - **`OpenFOAM-9/src/lagrangian`**
- In this directory, you will find many sub-directories containing the models used by the Eulerian-Lagrangian solvers.
- Among many models, you will find injection models, particle diameter distribution models, particle interaction models, particle tracking, spray models, turbulence models.
- The source code of the different cloud types implemented is located in the following directory,
  - **`OpenFOAM-9/src/lagrangian/parcel/clouds/Templates`**

Closure models for multiphase solvers in OpenFOAM

- The source code of the thermo-physical models is located in the following directory:

  - **`OpenFOAM-9/src/thermophysicalModels`**

- If you are dealing with combustion, you will find the source code of the implemented models in the following directory,

  - **`OpenFOAM-9/src/combustionModels`**

- And if you are dealing with pyrolysis or surface films, you will find the models in the following directories

  - **`OpenFOAM-9/src/regionModels/surfaceFilmModels`**
  - **`OpenFOAM-9/src/regionModels/thermalBaffleModels`**

Additional information related to the multiphase solvers in OpenFOAM

- The source code of the control options of the VOF solvers is located in the following directory:
  - **`OpenFOAM-9/src/twoPhaseModels/twoPhaseMixture/VoF`**

- The source code of the specialized interpolation schemes for the multiphase solvers is located in the following directory:
  - **`OpenFOAM-9/src/twoPhaseModels/twoPhaseMixture/`**
    - **`./interfaceCompression`**
    - **`./MPLIC`**
    - **`./PLIC`**

Looking for information in the source code

- If you want to find a string inside a file, you can use the command `grep` as follows:
  - `$> grep -r -n Tomiyama $FOAM_SRC`
- This command will look for the string Tomiyama in all the files inside the directory **`$FOAM_SRC`**. The argument `-r` means recursive and `-n` will output the line number.
- To locate a file that contains a given string in its name, you can use the `find` command as follows,
  - `$> find $FOAM_SRC -name *alphaContact*`
- This command will locate the files containing the string `*alphaContact*` in the directory **`$FOAM_SRC`**. The argument `-name` means case sensitive.
- Remember, you can use wildcards.

Final remarks

- The multiphase solvers available in OpenFOAM (VOF, eulerian-eulerian, eulerian-lagrangian, and combustion), will give you extensive multiphase modeling capabilities.
- Among the modeling capabilities:
    - Interface capturing using VOF approach.
    - Modeling of dispersed phases in a continuum phase.
        - Gas-liquid, gas-solid, liquid-solid, liquid-liquid interactions.
    - Population balance modeling.
    - Interfacial interaction models.
        - Virtual mass, drag force, lift force, wall lubrication, turbulence dispersion, heat transfer, among many models.
    - Kinetic theory of granular flows for particulate flows.
    - Mass transfer.
        - Due to temperature: boiling, melting, freezing, sublimation, condensation, evaporation,
        - Due to pressure: cavitation, flashing.

Final remarks

- The multiphase solvers available in OpenFOAM (VOF, eulerian-eulerian, eulerian-lagrangian, and combustion), will give you extensive multiphase modeling capabilities.
- Among the modeling capabilities:
    - Thermodynamics, turbulence, and porosity modeling for multiphase.
    - Particle tracking using euler-lagrangian approaches (DEM, MPPIC).
    - Particle forces modeling:
        - Drag, lift, virtual mass, brownian, magnetic, gravitational, among many models.
        - Particle collision handling.
    - Spray and surface film modeling.
    - Pyrolysis and combustion modeling.
    - Moving bodies, source terms and adaptive mesh refinement.

Roadmap

1. ~~Introduction to multiphase flows~~
2. ~~Modeling approaches for multiphase flows~~
3. ~~Governing equations and interfacial momentum transfer models~~
4. ~~Multiphase solvers in OpenFOAM~~
5. **Selecting physical properties, phase interaction, and advanced models**
6. Final remarks – Tips and tricks
7. Additional tutorials

On the discretization schemes and linear solvers

- Remember, you should always conduct production runs using a second order discretization scheme.
- Most of multiphase flows are transient, so you need to use transient methods.
- When using the multiphase family of solvers, you may need to add new entries in the *fvScheme* and *fvSolution* dictionaries.
- The new entries, correspond to the new terms and equations used by the solver.
- You can use the provided tutorials as templates for setting your case.
- Be careful, by no means use the tutorials as standard practice, specially when dealing with models.
- We highly advise to conduct a benchmarking/validation case a calibrate your models according to the application.

interFoam family solvers

Selecting physical properties and advanced physics

- In the directory **`constant`** you will find the dictionaries used to select physical properties and advanced physics.
- The following dictionaries are compulsory:
  - *`g`*: in this dictionary you set the gravity field.
  - *`transportProperties`*: in this dictionary you set the transport properties for each phase.
  - *`momentumTransport`*: in this dictionary you set the turbulence model.
- The dictionary *`transportProperties`* requires special attention, as it is in this dictionary where we set the transport properties for each phase.
- We also give a name to the phases in the dictionary *`transportProperties`*.
- If you are using interFoam, you only need to define 2 phases.
- If you are using multiphaseInterFoam you can define n phases.

Selecting physical properties and advanced physics

- In you are using the solver interFoam, the `transportProperties` dictionary should look like this one:

The first phase is always considered the primary phase

Phases naming convention. The name of the phases is chosen by the user.

phase1 properties

phase2 properties

Surface tension between phase1 and phase2

```
phases (phase1 phase2);

phase1
{
        transportModel   Newtonian;
        nu               nu [ 0 2 -1 0 0 0 0 ] 1e-06;
        rho              rho [ 1 -3 0 0 0 0 0 ] 1000;
}

phase2
{
        transportModel   Newtonian;
        nu               nu [ 0 2 -1 0 0 0 0 ] 1.48e-05;
        rho              rho [ 1 -3 0 0 0 0 0 ] 1;
}

sigma sigma [ 1 0 -2 0 0 0 0 ] 0.07;
```

Selecting physical properties, phase interaction, and advanced models

Selecting physical properties and advanced physics

- In you are using the solver multiphaseInterFoam, the *transportProperties* dictionary should look like this one (three phases in this case):

phase1 properties.
The name of the phases is chosen by the user.

phase2 properties.
The name of the phases is chosen by the user.

phase3 properties.
The name of the phases is chosen by the user.

```
phases
(

    phase1
    {
        transportModel  Newtonian;
        nu              nu [ 0 2 -1 0 0 0 0 ] 1e-06;
        rho             rho [ 1 -3 0 0 0 0 0 ] 1000;
    }

    phase2
    {
        transportModel  Newtonian;
        nu              nu [ 0 2 -1 0 0 0 0 ] 1e-06;
        rho             rho [ 1 -3 0 0 0 0 0 ] 500;
    }

    phase3
    {
        transportModel  Newtonian;
        nu              nu [ 0 2 -1 0 0 0 0 ] 1.5e-56;
        rho             rho [ 1 -3 0 0 0 0 0 ] 1;
    }

);
...
```

Continues in the next slide (1/2)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseInterFoam, the *transportProperties* dictionary should look like this one (three phases in this case):

Surface tension between phases

Surface tension combinations for all phases

```
sigmas
(

          (phase1 phase2)          0.07
          (phase1 phase3)          0.07
          (phase2 phase3)          0.07

);
```

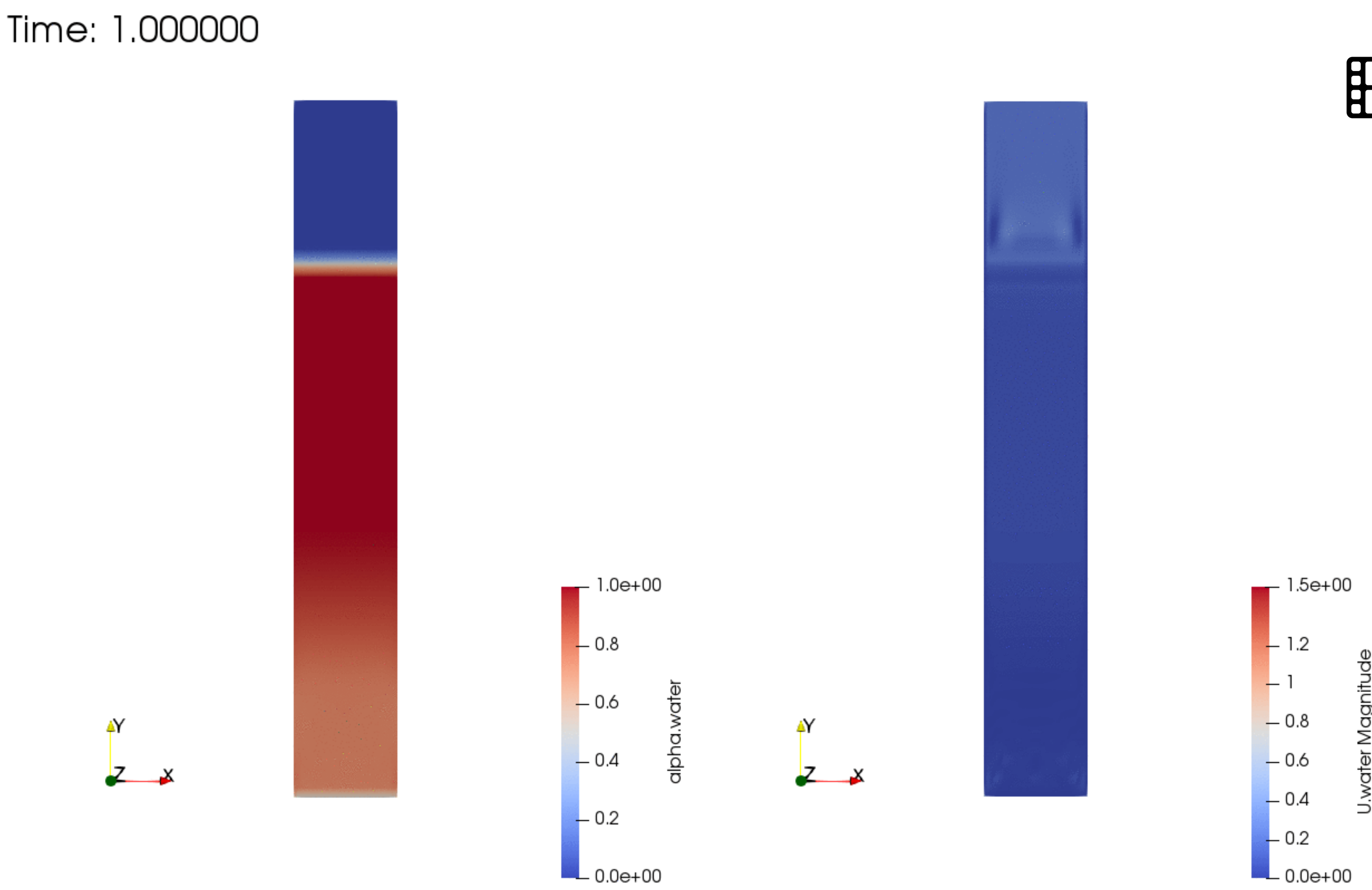
(2/2)

Selecting phasic boundary conditions

- In the directory **0** you will find the dictionaries used to define phasic boundary conditions and initial conditions.
- When you create the dictionaries for the boundary and initials conditions for the volume fraction or alpha, you use the same naming convention as in the dictionary *transportProperties*,
  - If you are using interFoam, you will need to create the following dictionary for the primary phase:
    - *alpha.phase1*
  - If you are using multiphaseInterFoam, you will need to create the following dictionaries for all the phases defined in *transportProperties* (three phases in this case):
    - *alpha.phase1*
    - *alpha.phase2*
    - *alpha.phase3*
    - *alphas* (It contains all the phases volume fraction. In this dictionary all walls are set to zeroGradient)

multiphaseEulerFoam family solvers (for gas-liquid applications)

Selecting physical properties, phase interaction, and advanced models



multiphaseEulerFoam solution
Bubble column (gas-liquid)

http://www.wolfdynamics.com/training/mphase/image23.gif

Selecting physical properties and advanced physics

- In the directory **`constant`** you will find the dictionaries used to select physical properties and advanced physics.
- The following dictionaries are compulsory (assuming that you named the phases air and water):
  - *`g`*: in this dictionary you set the gravity field.
  - *`phaseProperties`*: in this dictionary you set how phases interacts and the physical and interfacial models. In this dictionary, you also set the names of the phases, *e.g.*, water and air.
  - *`thermophysicalProperties.air`*: in this dictionary you set the thermo physical properties for air.
  - *`thermophysicalProperties.water`*: in this dictionary you set the thermo physical properties for water.
  - *`momentumTransport.air`*: in this dictionary you set the turbulence model for air.
  - *`momentumTransport.water`*: in this dictionary you set the turbulence model for water.

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam, the `phaseProperties` dictionary should look like this one (assuming that you named the phases as air and water):

```
type    basicMultiphaseSystem;

phases (air water);

air
{
        diameterModel    ...
}

water
{
        diameterModel    ...
}

blending
{
        ...
}
```

Phases naming convention. The name of the phases is chosen by the user.

Bubbles/droplets diameter model for air.

Bubbles/droplets diameter model for water.

Phases blending factor.

- Type of phase system to solve, several options are available.
- If you are interested in population balance modeling, here is where you can select it.
- Depending on the model type selected, different entries are required.
- We will only address the basic setup,
  - basicMultiphaseSystem
- The other models availble are the following,
  - interfaceCompositionPhaseChangeMultiphaseSystem
  - interfaceCompositionPhaseChangePopulationBalanceMultiphaseSystem
  - populationBalanceMultiphaseSystem
  - thermalPhaseChangeMultiphaseSystem
  - thermalPhaseChangePopulationBalanceMultiphaseSystem

Continues in the next slide (1/4)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam, the *phaseProperties* dictionary should look like this one (assuming that you named the phases as air and water):

Definition of surface tension.

Surface tension of air and water.

Bubbles/droplets aspect ratio model.

Drag force model.

```
...

surfaceTension
(
        (air and water)    0.07
);

aspectRatio
(
        ...
);

drag
(
        ...
);

...
```

Continues in the next slide (2/4)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam, the *phaseProperties* dictionary should look like this one (assuming that you named the phases as air and water):

Virtual mass model.

Heat transfer model.

Lift model.

```
...

virtualMass
(
        ...
);

heatTransfer
(
        ...
);

phaseTransfer
(

);

...
```

Continues in the next slide (3/4)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam, the *phaseProperties* dictionary should look like this one (assuming that you named the phases as air and water):

Lift model.

Wall lubrication model.

Turbulent dispersion model.

```
...

lift
(
        ...
);

wallLubrication
(
        ...
);

turbulentDispersion
(
        ...
);
```

(4/4)

Phase interaction modeling

- To define which phase is the continuous one and which one is the dispersed phase, we use the following convection
    - **(air in water)** //air bubbles dispersed in water
    - **(water in air)** //water droplets dispersed in air
    - **(air and water)** //no obvious dispersed phase
- This convention can be used with any model.
- Additionally, blending methods can be used to mix the effect of the three models.
- They are specified per model type, and a default can also be set. The methods available are available:
    - **noBlending**, where the continuous phase is known throughout the flow and can be specified;
    - **linear**, uses a linear basis between specified maximum dispersed phase fractions;
    - **hyperbolic** uses a hyperbolic basis, which is smoother than the linear method, but more expensive to evaluate.

Phase interaction modeling

- To account for phase interaction, the *phaseProperties* dictionary should look like this one.
- This formulation allows phase inversion, where the continuous and dispersed phases can switch.
- In the following example we are only considering phase interaction for virtual mass:

```
virtualMass
(
        (air in water)
        {
                type            constantCoefficient;
                Cvm             0.5;
        }

        (water in air)
        {
                type            constantCoefficient;
                Cvm             0.5;
        }
        (air and water)
         {
                type    none;
        }
);
```

Virtual mass model definition block.

Air bubbles dispersed in water.

Virtual mass model.

Water droplets dispersed in air.

Virtual mass model.

No obvious dispersed phase.

Phase interaction modeling

- To account for blending interaction between all phases, the *phaseProperties* dictionary should look like this one :

Phases blending factor.

Definition of default blending method. This blending model will be used for all phases interaction.

Blending method

Specification of blending coefficients for each phase. It follows the phases naming convention

```
blending
{
        default
        {
                type linear;

                minFullyDispersedAlpha.air          0.3;
                minPartlyDispersedAlpha.air         0.5;
                minFullyDispersedAlpha.water        0.3;
                minPartlyDispersedAlpha.water       0.5;
        }
}
```

Selecting phasic boundary conditions

- In the directory **0** you will find the dictionaries used to define phasic boundary conditions and initial conditions.
- When using the solver multiphaseEulerFoam for dispersed flows (gas-liquid), you need to create the dictionaries for the boundary and initials conditions for every single phasic field variable with the exception of granular temperature (*Tetha*).
- Assuming that you named the phases water and air, these are the dictionaries that you must create in the directory **0**:

  - *alpha.air*
  - *alpha.water*
  - *p* (calculated from p_rgh)
  - *p_rgh*
  - *T.air*
  - *T.water*
  - *U.air*
  - *U.water*

Selecting phasic boundary conditions

- If you are using a turbulence model, you should also create the dictionaries for the boundary and initials conditions for the turbulence field variables.
- For instance, if you are using the $\kappa - \epsilon$ turbulence model, you should create the following dictionaries in the directory **0**:

  - *k.air*
  - *k.water*
  - *epsilon.air*
  - *epsilon.water*
  - *nut.air*
  - *nut.water*
  - *alphat.air*
  - *alphat.water*

- Some turbulence models for multiphase flows (gas-liquid, gas-solid, liquid-solid interactions), are different from those for single-phase flows and liquid-liquid interactions.

Selecting phasic equation of state

- Finally, remember that when defining the thermo physical properties you need to use the right model for gases and the right model for liquids.
- You set the thermo physical properties in the *thermophysicalProperties* dictionary located in the directory **constant**.
- For gases, you have the following options for defining the equation of state,
  - equationOfState incompressiblePerfectGas;
  - equationOfState perfectGas;
  - equationOfState icoPolynomial;
  - equationOfState PengRobinsonGas;
  - equationOfState Boussinesq;
  - equationOfState rPolynomial;
- For liquids, you have the following options for defining the equation of state
  - equationOfState adiabaticPerfectFluid;
  - equationOfState rhoConst;
  - equationOfState rPolynomial;
- You will also need to define the right physical properties.

Naming convention

- Multiphase solvers in OpenFOAM use the following naming convention:

  - **alpha.<phase_name>** denotes “phase fraction of phase <phase_name>”, e.g., alpha.water is phase fraction of phase water

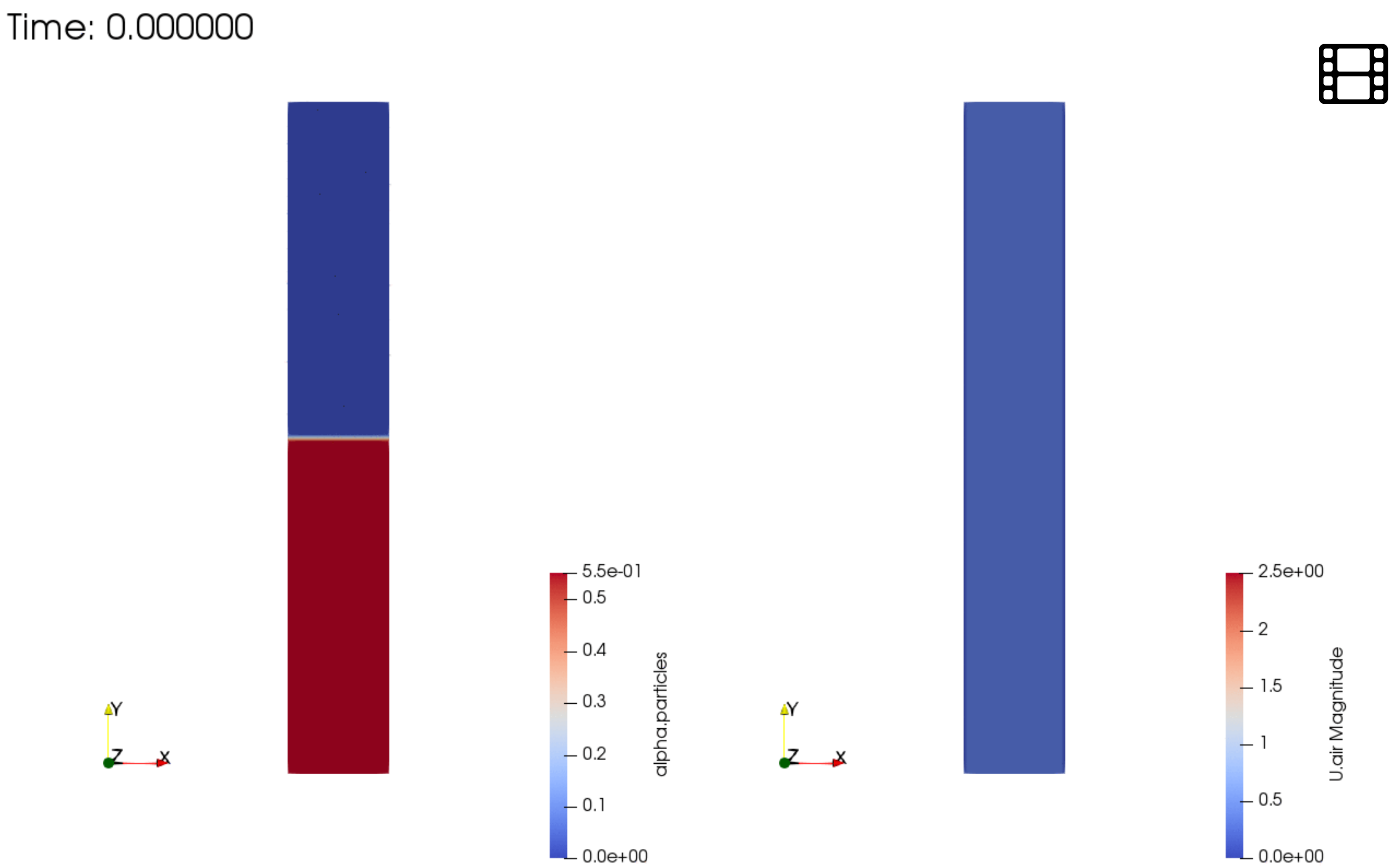
  - **<property>.<phase name>** denotes “property <property> of phase <phase_name>”, e.g., turbulenceProperties.water are the turbulence properties of phase water

- This phase naming convention supports arbitrary numbers of phases.

- This phase naming convention is used for all multiphase solvers.

multiphaseEulerFoam family solvers (for kinetic theory of granular flows)

Selecting physical properties, phase interaction, and advanced models



multiphaseEulerFoam solution
Fluidized bed (gas-solid or granular flow)

http://www.wolfdynamics.com/training/mphase/image24.gif

Selecting physical properties, phase interaction, and advanced models

Selecting physical properties and advanced physics

- In the directory **`constant`** you will find the dictionaries used to select physical properties and advanced physics.
- The following dictionaries are compulsory (assuming that you named the phases fluid and particles):
  - *g*: in this dictionary you set the gravity field.
  - *phaseProperties*: in this dictionary you set how phases interacts and the physical and interfacial models. In this dictionary, you also set the names of the phases, *e.g.*, fluid and particles.
  - *thermophysicalProperties.fluid*: in this dictionary you set the thermo physical properties of the fluid phase (gas or liquid).
  - *thermophysicalProperties.particles*: in this dictionary you set the thermo physical properties of the solid/granular phase.
  - *momentumTransport.fluid*: in this dictionary you set the turbulence model of the fluid phase (gas or liquid).
  - *momentumTransport.particles*: in this dictionary you set the turbulence model properties of the solid/granular phase.

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam for granular flows, the *phaseProperties* dictionary should look like this one (assuming that you named the phases fluid and particles):

```
type    basicMultiphaseSystem;

phases (particles air);

referencePhase air;

particles
{
        type        purePhaseModel;

        diameterModel       ...
}

air
{
        type        purePhaseModel;

        diameterModel       ...
}
...
```

Type of phase system to solve. Several options available.

Phases naming convention. The name of the phases is chosen by the user.

Reference phase for averaging.

Particle diameter model for solid/granular phase.

Bubbles/droplets diameter model for air.

Continues in the next slide (1/5)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam for granular flows, the *phaseProperties* dictionary should look like this one (assuming that you named the phases fluid and particles):

Phases blending factor.
In granular flows you do not need to use blending. Usually you will need to define the continuous phase using the keyword continuousPhase

Definition of surface tension.

Bubbles/droplets aspect ratio model.

```
...

blending
{
        type                    none;
        continuousPhase         air;
}

sigma
(
        (particles and air)   0
);

aspectRatio
(
        ...
);

...
```

Continues in the next slide (2/5)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam for granular flows, the *phaseProperties* dictionary should look like this one (assuming that you named the phases fluid and particles):

Interface compression

Drag force model.

Type of drag force model.

Swarm corrections.

```
...
interfaceCompression
(

);

drag
(
    (particles in air)
    {
        type GidaspowErgunWenYu;
        residualAlpha       1e-6;
        residualRe          1e-3;

        swarmCorrection
        {
            type none;
        }
    }
);
...
```

Continues in the next slide (3/5)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam for granular flows, the *phaseProperties* dictionary should look like this one (assuming that you named the phases fluid and particles):

Virtual mass model.

Heat transfer model.

```
...
virtualMass
(
    (particles in air)
    {
        type        constantCoefficient;
        Cvm         0.5
    }
);

heatTransfer
(
    (particles in air)
    {
        type            RanzMarshall;
        residualAlpha   1e-4
    }
);
...
```

Continues in the next slide (4/5)

Selecting physical properties and advanced physics

- In you are using the solver multiphaseEulerFoam for granular flows, the *phaseProperties* dictionary should look like this one (assuming that you named the phases fluid and particles):

```
...

lift
(
        ...
);

wallLubrication
(
        ...
);

turbulentDispersion
(
        ...
);
```

Lift model.

Wall lubrication model.

Turbulent dispersion model.

(5/5)

Selecting physical properties and advanced physics

- When using the solver multiphaseEulerFoam for granular flows, the dictionary *momentumTransport.particles* requires special attention.
- In this dictionary you define the kinetic theory for granular flows.

Turbulence model for granular phase.

Model to use.

```
simulationType RAS;

RAS
{
        RASModel kineticTheory;

        turbulence on;
        printCoeffs on;

...
```

Kinetic theory reference:
Derivation, implementation, and validation of computer simulation models for gas-solid fluidized beds, van Wachem, B.G.M., Ph.D. Thesis, Delft University of Technology, Amsterdam, 2000.

Continues in the next slide (1/3)

Selecting physical properties and advanced physics

- When using the solver multiphaseEulerFoam for granular flows, the dictionary *turbulenceProperties.particles* requires special attention.
- In this dictionary you define the kinetic theory for granular flows.

```
...

kineticTheoryCoeffs
{
    equilibrium             off;
    e                       0.8;
    alphaMax                0.65;
    alphaMinFriction        0.5;
    residualAlpha           1e-4;

    viscosityModel              Gidaspow;
    conductivityModel           Gidaspow;
    granularPressureModel       Lun;
    frictionalStressModel       JohnsonJackson;
    radialModel                 SinclairJackson;

...
```

Kinetic theory model coefficients and sub-models.

Maximum packing phase-fraction.

Value of the solid volume fraction when the frictional stresses become important (minimum value for which the frictional stresses are zero)

Kinetic theory model sub-models.

Set it to on to solve granular temperature equation instead of balance equation. This approach is valid is the velocity of the particles is low and the solid phase packing is high

Particles restitution coefficient. Ratio of the final to initial relative velocity between two particles after they collide. If equal to 1 perfect elastic collision (no kinetic energy is dissipated). If equal to 0 perfect inelastic collision (all kinetic energy is dissipated)

Kinetic theory reference:
Derivation, implementation, and validation of computer simulation models for gas-solid fluidized beds, van Wachem, B.G.M., Ph.D. Thesis, Delft University of Technology, Amsterdam, 2000.

Continues in the next slide (2/3)

Selecting physical properties and advanced physics

- When using the solver multiphaseEulerFoam for granular flows, the dictionary *turbulenceProperties.particles* requires special attention.
- In this dictionary you define the kinetic theory for granular flows.

Kinetic theory model sub-models. Fr, eta, and p are material dependent coefficient used for the calculation of normal frictional stresses. p (frictional normal stress) and phi (angle of internal friction) are coefficients used to compute the shear viscosity

Phase pressure model coefficients and sub-models.

```
...
            JohnsonJacksonCoeffs
            {
                Fr              0.05;
                eta             2;
                p               5;
                phi             28.5;
                alphaDeltaMin   0.05;
            }
        }
        phasePressureCoeffs
        {
            preAlphaExp     500;
            expMax          1000;
            alphaMax        0.62;
            g0              1000;
        }
}
```

Kinetic theory reference:
Derivation, implementation, and validation of computer simulation models for gas-solid fluidized beds, van Wachem, B.G.M., Ph.D. Thesis, Delft University of Technology, Amsterdam, 2000.

(3/3)

Selecting physical properties and advanced physics

- When using the solver multiphaseEulerFoam for granular flows, the dictionary *thermophysicalProperties.particles* requires special attention.
- This dictionary is different from the dictionaries you use for fluids (gas or liquid).

Equation of state for solids.
When working with fluids, you usually use perfectGas or perfectFluid.

```
thermoType
{
    type                heRhoThermo;
    mixture             pureMixture;
    transport           const;
    thermo              hConst;
    equationOfState     rhoConst;
    specie              specie;
    energy              sensibleInternalEnergy;
}

...
```

Continues in the next slide (1/2)

Selecting physical properties and advanced physics

- When using the solver multiphaseEulerFoam for granular flows, the dictionary *thermophysicalProperties.particles* requires special attention.
- This dictionary is different from the dictionaries you use for fluids (gas or liquid).

Physical properties of the solid.

```
...

mixture
{
        specie
        {
                molWeight       100;
        }
        equationOfState
        {
                rho             2500;
        }
        thermodynamics
        {
                Cp              6000;
                Hf              0;
        }
        transport
        {
                mu              0;
                Pr              1;
        }
}
```

(2/2)

Selecting phasic boundary conditions

- In the directory **0** you will find the dictionaries used to define phasic boundary conditions and initial conditions.
- When using the solver multiphaseEulerFoam for granular flows, you need to create the dictionaries for the boundary and initials conditions for every single phasic field variable, including the granular temperature (*Tetha*).
- Assuming that you named the phases particles and air, these are the dictionaries that you must create in the directory **0**:

  - *alpha.air*
  - *alpha.particles*
  - *p* (calculated from p_rgh)
  - *p_rgh*
  - *T.air*
  - *T.particles*
  - *Theta.particles*
  - *U.air*
  - *U.particles*

Selecting phasic boundary conditions

- If you are using a turbulence model, you should also create the dictionaries for the boundary and initials conditions for the turbulence field variables.
- For instance, if you are using the $\kappa - \epsilon$ turbulence model for the fluid phase, you should create the following dictionaries in the directory **0**:

  - *epsilon.air*
  - *k.air*
  - *nut.air*
  - *nut.particles*
  - *alphat.air*
  - *alphat.particles*

- Some turbulence models for multiphase flows (gas-liquid, gas-solid, liquid-solid interactions), are different from those for single-phase flows and liquid-liquid interactions.

Selecting phasic equation of state

- Finally, remember that when defining the thermo physical properties, you need to use the right model for gases, the right model for liquids and the right models for solids.
- You set the thermo physical properties in the *`thermophysicalProperties`* dictionary located in the directory **`constant`**.
    - For gases, you have the following options for defining the equation of state
        - equationOfState incompressiblePerfectGas;
        - equationOfState perfectGas;
        - equationOfState icoPolynomial;
        - equationOfState PengRobinsonGas;
        - equationOfState Boussinesq;

Selecting phasic equation of state

- Finally, remember that when defining the thermo physical properties, you need to use the right model for gases, the right model for liquids and the right models for solids.
- You set the thermo physical properties in the *thermophysicalProperties* dictionary located in the directory **constant**.
    - For liquids, you have the following options for defining the equation of state
        - equationOfState adiabaticPerfectFluid;
        - equationOfState perfectFluid;
    - For solids, you have the following options for defining the equation of state
        - equationOfState rhoConst;
    - You will also need to define the right physical properties.

Selecting physical properties, phase interaction, and advanced models

Naming convention

- Multiphase solvers in OpenFOAM use the following naming convention:

  - **alpha.<phase_name>** denotes “phase fraction of phase <phase_name>”, e.g., alpha.water is phase fraction of phase water

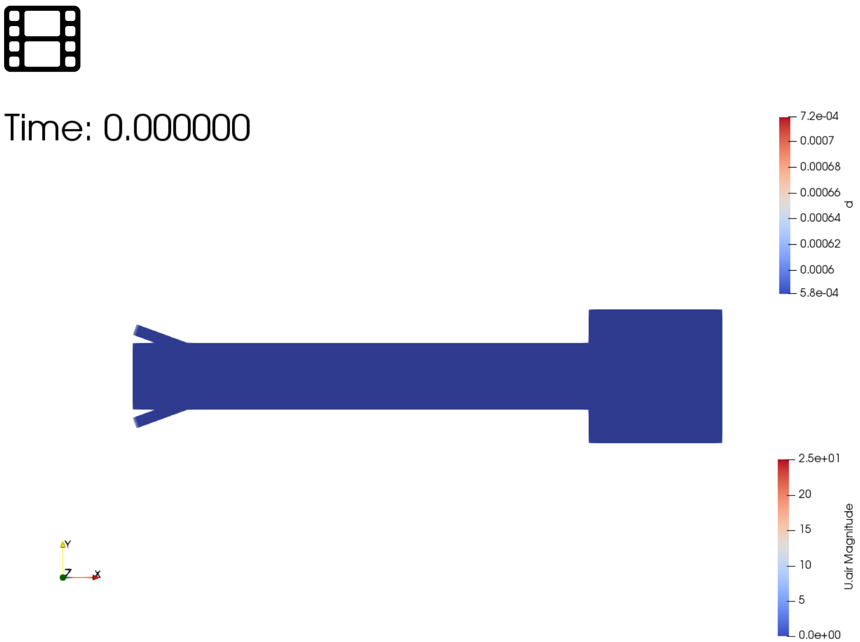
  - **<property>.<phase name>** denotes “property <property> of phase <phase_name>”, e.g., turbulenceProperties.water are the turbulence properties of phase water

- This phase naming convention supports arbitrary numbers of phases.

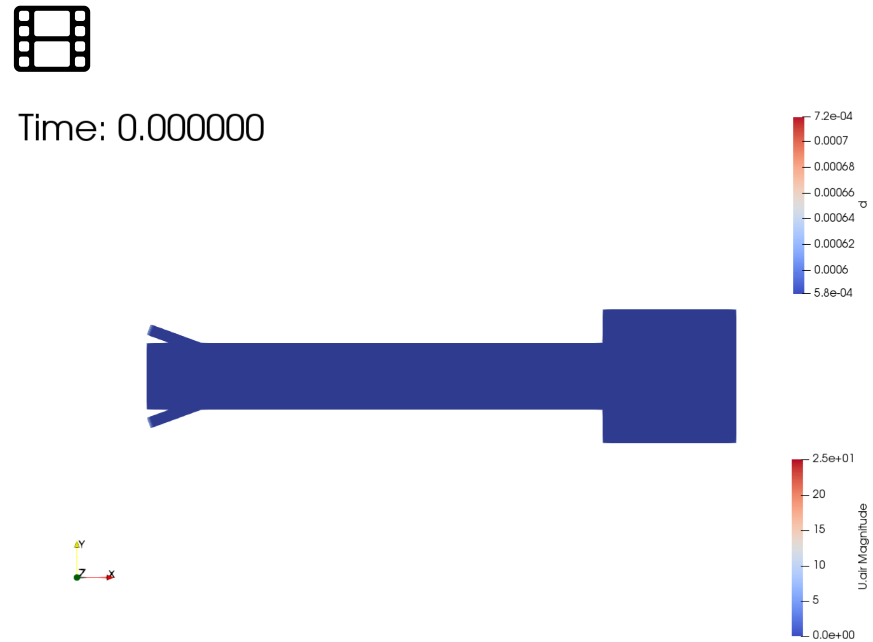
- This phase naming convention is used for all multiphase solvers.

Lagrangian family solvers (DPMFoam and MPPICFoam)

Selecting physical properties, phase interaction, and advanced models



MPPIC cloud solution
Particle tracking with coupled hydrodynamic
CPU Time = 240 seconds



Dense cloud solution
Particle tracking with coupled hydrodynamic
CPU Time = 2520 seconds

http://www.wolfdynamics.com/training/mphase/image26.gif

http://www.wolfdynamics.com/training/mphase/image27.gif

Lagrangian solvers in OpenFOAM

- To model dense particle flows or flows where the motion is driven by particle interactions and collisions (such as fluidized beds), you can use the following solver:
    - **denseParticleFoam**

- The standard set of Lagrangian clouds are now selectable at run-time.
- This means that a solver that supports Lagrangian modelling can now use any type of cloud (with some restrictions).
    - https://github.com/OpenFOAM/OpenFOAM-9/commit/43d66b5e7c566e087db187df023f134422ee8a4b

- The particle-particle interaction can be treated using the MPPIC approach or the dense approach (or any other approach compatible with the solver).
- In the dense approach the particle-particle interactions of every single particle is computed (computationally expensive).
- In the MPPIC approach the computation complexity is reduced by using parcels of particles and pack models (this approach is more affordable than the dense approach).

Lagrangian solvers in OpenFOAM

- In previous OpenFOAM versions (8 and below), the particle-particle treatment was selected via the available solvers, namely,
  - **DPMFoam**
  - **MPPICFoam**

- Starting form OpenFOAM 9, these solvers have been replaced by the following more general solver,
  - **MPPICFoam** and **DPMFoam** have been replaced by **denseParticleFoam**.

- In addition, the reacting parcel solvers have been replaced by the following more general solvers,
  - **reactingParcelFoam** have ben replaced by **buoyantReactingFoam**.
  - **coalChemistryFoam**, **simpleReactingParcelFoam**, **simpleReactingParticleFoam**, and **sprayFoam** have ben replaced by **reactingFoam**.

Lagrangian solvers in OpenFOAM

- **The dense approach (or DPMFoam in older OpenFOAM versions) ***: this approach uses the particle-in-cell (PIC) method, where particles positions and interactions are tracked in a Lagrangian frame, and the fluid flow is solved in a Eulerian frame. The fluid and the particles solvers can be coupled or uncoupled. This solver is suited for dense flows. For very large number of particles, it can become really expensive, as it needs to track the interactions between every single particle.

- **The MPPIC approach (or MPPICFoam in older OpenFOAM versions) ****: this approach uses the MPPIC method or multiphase particle-in-cell method, where particles are treated as discrete elements and as a continuum, and the fluid flow is solved in a Eulerian frame. This solver model particles collisions without the need of resolving particle-particle interactions, by mapping the Lagrangian coordinates to the Eulerian grid, solving the continuum derivative terms in the Eulerian grid and mapping back the interactions to the Lagrangian particles. This solver can be used for dense and dilute flows. Contrary to the dense approach, for large number of particles this solver has a low computational cost.

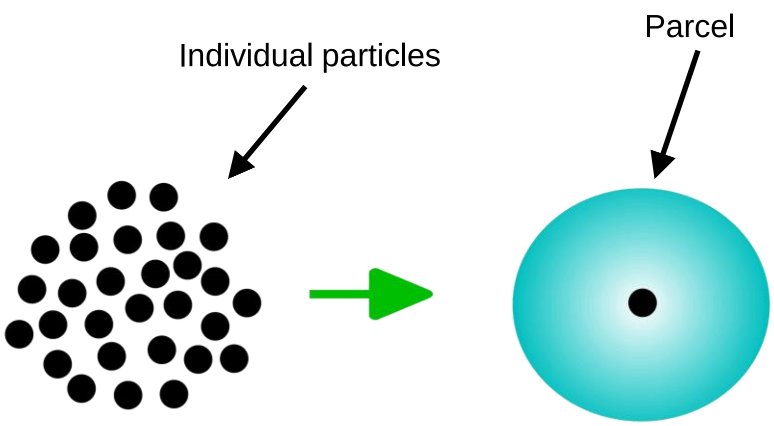
*D.M. Snider, An Incompressible Three-Dimensional Multiphase Particle-in-Cell Model for Dense Particle Flows, Journal of Computational Physics, Volume 170, Issue 2, 1 July 2001, Pages 523–549
**P. J. O'Rourke et al., A model for collisional exchange in gas/liquid/solid fluidized beds, Chemical Engineering Science, Volume 64, 2009, pages 1784-1797

Selecting physical properties, phase interaction, and advanced models

Lagrangian solvers in OpenFOAM

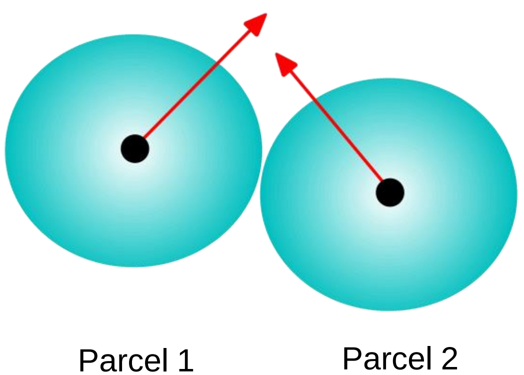
What are parcels in the context of Lagrangian solvers?

- For real applications, tracking all particles interactions and positions is too expensive.
- It is better to track a parcel of particles.
- When we use parcels, we put several particles that share the same properties into one parcel.
- Instead of tracking all particles, we track a representative particle on the parcel.
- The mass in collisions is that of the entire parcel.
- Radius of parcel is obtained from the mass of the marcel and the density of the parcel is considered to be the same as particle density.



What is a kinematic cloud in the context of Lagrangian solvers?

- A kinematic cloud is equivalent to the eulerian mesh.
- The kinematic cloud is the assembly of all particles in the system.



Selecting physical properties and advanced physics

- In the directory **`constant`** you will find the dictionaries used to select physical properties and advanced physics.
- The following dictionaries are compulsory (assuming that you named the continuous phase air):
  - *`g`*: in this dictionary you set the gravity field.
  - *`cloudPropertites`*: in this dictionary you set the properties of the kinematic cloud (particles).
  - *`transportProperties`*: in this dictionary you set the transport properties of the continuous phase. In this dictionary, you also set the names of the continuous phase, *e.g.*, air.
  - *`momentumTransport.air`*: in this dictionary you set the turbulence model of the continuous phase.

Selecting physical properties and advanced physics

- The dictionary *transportProperties* requires special attention because in this dictionary we set the name of the continuous phase and its transport properties.
- The dictionary *transportProperties* should look like this one:

Continuous phase name. The name of the phases is chosen by the user.

Density of the continuous phase

Transport model and viscosity of the continuous phase

```
continuousPhaseName     air;

rho.air                 1.2;

transportModel          Newtonian;
nu                      1e-05;
```

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

- The standard set of Lagrangian clouds are now selectable at run-time.
- This means that a solver that supports Lagrangian modelling can now use any type of cloud (with some restrictions).
- Previously, solvers were hard-coded to use specific cloud modelling. In addition, a cloud-list structure has been added so that solvers may select multiple clouds, rather than just one.
- More information at the following link, https://github.com/OpenFOAM/OpenFOAM-9/commit/43d66b5e7c566e087db187df023f134422ee8a4b

```
type        MPPICCloud;
...

/*
Possible types of clouds that can be used with denseParticleFoam:

type    cloud;
A basic cloud of solid particles. Includes forces,patch interaction, injection,
dispersion and stochastic collisions. Same as the cloud previously used by
rhoParticleFoam (uncoupledKinematicParticleFoam)


type    collidingCloud;
As "cloud" but with resolved collision modelling. Same as the cloud previously
used by DPMFoam and particleFoam (icoUncoupledKinematicParticleFoam).


type    MPPICCloud;
As "cloud" but with MPPIC collision modelling. Same as the cloud previously
used by MPPICFoam.

In the directory OpenFOAM-9/src/lagrangian/parcel/clouds/Templates you
will find the source code of the different cloud types.
*/
```

Continues in the next slide (1/9)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

```
solution
{

        coupled         true;
        transient       yes;

        maxCo           0.7;

        cellValueSourceCorrection on;

        interpolationSchemes
        {
                rho.air         cell;
                U.air           cellPoint;
                mu.air          cell;
                alpha.air       cell;

        }

        averagingMethod     dual;
...
```

- Activation switches.
- The coupled switch turns on/off coupled hydrodynamics.
- The transient switch turns on/off transient coupling.

Maximum currant number of the Lagrangian mesh.

Flag to correct cell values with latest transfer information during the lagrangian timestep

Interpolation method of cell centered quantities to particle solver.

Interpolation of the average of the lagrangian phase. Important only if you are using a packing model.

Continues in the next slide (2/9)

Selecting physical properties and advanced physics

- The dictionary `cloudProperties` requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

Time integration schemes of particles.

Additional source terms

Physical properties of the particles.

```
...

    integrationSchemes
    {
        U           Euler;
    }

    sourceTerms
    {
        schemes
        {
            U       semiImplicit 1;
        }
    }
}

constantProperties
{
    rho0                1000;
}
...
```

Continues in the next slide (3/9)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

Selection of sub-models.

Particle forces models.

```
...

subModels
{

    particleForces
    {
        ErgunWenYuDrag
        {
            alphac alpha.air;
        }

        gravity;
    }

...
```

Continues in the next slide (4/9)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

```
...

    injectionModels                                   <- Injection models.
    {

        model1                                        <- User defined name of injection model. Can add more than one.
        {

            type                patchInjection;       <- Type of injection model.

            parcelBasisType     fixed;

            patchName           mixing_elbow_inlet1;  <- Inject particles from this patch with given boundary values.
            U0                  (3 0 0);

            nParticle           1;                    <- Quantity of particles inside each parcel.

            parcelsPerSecond    10000;                <- Quantity of parcels to inject per second

...
```

Continues in the next slide (5/9)

Selecting physical properties and advanced physics

- The dictionary `cloudProperties` requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

A valid Function1 type to scale the flow rate profile. In this case, we are using a constant function.

Injection parameters. Total mass to inject into the system, start time of injection and duration of injection.

Model for diameter distribution of particles.

Diameter distribution values.

```
...
            flowRateProfile constant    1;

            massTotal               0;
            SOI                     0;
            duration                2;

            sizeDistribution
            {
                type        normal;
                normalDistribution
                {
                    expectation   100e-6;
                    variance      25e-6;
                    minValue      20e-6;
                    maxValue      180e-6;
                }
            }
        }
...
```

Continues in the next slide (6/9)

Selecting physical properties and advanced physics

- The dictionary `cloudProperties` requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

Turbulence dispersion model.

Particles-patch interaction model.

Name of the patch.
You must define all patches and the interaction type.

Interaction type.

```
...

    dispersionModel             none;

    patchInteractionModel       localInteraction;

    localInteractionCoeffs
    {
            patches
            (
                    patch_name
                    {
                            type    rebound;
                            e       0.97;
                            mu      0.09;
                    }
            );
    }

...
```

Continues in the next slide (7/9)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

```
...

        //packingModel none;
        //packingModel explicit;
        packingModel implicit;

        implicitCoeffs
        {
                alphaMin            0.0001;
                rhoMin              1.0;
                applyGravity        false;
                particleStressModel

                {
                        HarrisCrighton;
                        alphaPacked         0.6;
                        pSolid              5.0;
                        beta                2.0;
                        eps                 1.0e-2;
                }
        }
```

Particles packaging model used to apply non-linear stresses to particle parcels.

Coefficients related to the packing method selected

Particle stresses sub-model.

Continues in the next slide (8/9)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **MPPIC** approach, the dictionary should look like this one:

```
...
        heatTransferModel none;
        surfaceFilmModel none;
        dampingModel none;

        isotropyModel stochastic;
        stochasticCoeffs
        {
                timeScaleModel
                {
                        type isotropic;
                        alphaPacked 0.6;
                        e 0.9;
                }
        }

        stochasticCollisionModel none;
        radiation off;

    cloudFunctions{}
}
```

Heat transfer model.

Surface film model.

Collision damping model.

Particle scattering as a result of collisions.

Stochastic collision model.

Radiation model.

Specialized function objects

Continues in the next slide (9/9)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

- The standard set of Lagrangian clouds are now selectable at run-time.
- This means that a solver that supports Lagrangian modelling can now use any type of cloud (with some restrictions).
- Previously, solvers were hard-coded to use specific cloud modelling. In addition, a cloud-list structure has been added so that solvers may select multiple clouds, rather than just one.
- More information at the following link, https://github.com/OpenFOAM/OpenFOAM-9/commit/43d66b5e7c566e087db187df023f134422ee8a4b

```
type        collidingCloud;
...

/*
Possible types of clouds that can be used with denseParticleFoam:

type    cloud;
A basic cloud of solid particles. Includes forces,patch interaction, injection,
dispersion and stochastic collisions. Same as the cloud previously used by
rhoParticleFoam (uncoupledKinematicParticleFoam)


type    collidingCloud;
As "cloud" but with resolved collision modelling. Same as the cloud previously
used by DPMFoam and particleFoam (icoUncoupledKinematicParticleFoam).


type    MPPICCloud;
As "cloud" but with MPPIC collision modelling. Same as the cloud previously
used by MPPICFoam.

In the directory OpenFOAM-9/src/lagrangian/parcel/clouds/Templates you
will find the source code of the different cloud types.
*/
```


Continues in the next slide (1/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

```
solution
{

    coupled        true;
    transient      yes;

    maxCo          0.7;

    cellValueSourceCorrection on;

    interpolationSchemes
    {
        rho.air        cell;
        U.air          cellPoint;
        mu.air         cell;
        alpha.air      cell;
    }

    //averagingMethod   dual;
...
```

- Activation switches.
- The coupled switch turns on/off coupled hydrodynamics.
- The transient switch turns on/off transient coupling.

Maximum currant number of the Lagrangian mesh.

Flag to correct cell values with latest transfer information during the lagrangian timestep

Interpolation method of cell centered quantities to particle solver.

Not needed if you are using DPMFoam.

Continues in the next slide (2/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Time integration schemes of particles.

```
...

        integrationSchemes
        {
                U               Euler;
        }

        sourceTerms
        {
                schemes
                {
                        U           semiImplicit 1;
                }
        }
}

...
```

Continues in the next slide (3/11)

Selecting physical properties and advanced physics

- The dictionary `cloudProperties` requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Physical properties of the particles.

```
...

constantProperties
{
        parcelTypeId        1;

        rhoMin              1e-15;
        minParcelMass       1e-15;

        rho0                2526;
        youngsModulus       1e8;
        poissonsRatio       0.35;

        constantVolume      false;

        alphaMax            0.9;
}

...
```

Continues in the next slide (4/11)

Selecting physical properties and advanced physics

- The dictionary *`cloudProperties`* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Selection of sub-models.

Particle forces models.

```
...

subModels
{

    particleForces
    {
        ErgunWenYuDrag
        {
            alphac alpha.air;
        }

        gravity;
    }

...
```

Continues in the next slide (5/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

```
...

    injectionModels
    {

        model1
        {

            type                patchInjection;

            parcelBasisType     fixed;

            patchName           mixing_elbow_inlet1;
            U0                  (3 0 0);

            nParticle           1;

            parcelsPerSecond    10000;

...
```

Annotations:
- Injection models. → injectionModels
- User defined name of injection model. Can add more than one. → model1
- Type of injection model. → type
- Inject particles from this patch with given boundary values. → patchName / U0
- Quantity of particles inside each parcel. → nParticle
- Quantity of parcels to inject per second → parcelsPerSecond

Continues in the next slide (6/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Injection parameters. Total mass to inject into the system, start time of injection and duration of injection.

Model for diameter distribution of particles.

Diameter distribution values.

```
...
            flowRateProfile constant    1;

            massTotal          0;
            SOI                0;
            duration           2;

            sizeDistribution
            {
                type       normal;
                normalDistribution
                {
                    expectation   100e-6;
                    variance      25e-6;
                    minValue      20e-6;
                    maxValue      180e-6;
                }
            }
        }
...
```

Continues in the next slide (7/11)

Selecting physical properties and advanced physics

- The dictionary `cloudProperties` requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Turbulence dispersion model.

Particles-patch interaction model.

Name of the patch.
You must define all patches and the interaction type.

Interaction type.

```
...

    dispersionModel          none;

    patchInteractionModel    localInteraction;

    localInteractionCoeffs
    {
        patches
        (
            patch_name
            {
                type    rebound;
                e       0.97;
                mu      0.09;
            }
        );
    }

...
```

Continues in the next slide (8/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Particles-particles collision model.

Particles-particles collision model parameters and sub-models.

Pair forces model between particles colliding with a spring, slider, damper model

pairModel coefficients

```
...

        collisionModel pairCollision;

        pairCollisionCoeffs
        {
                maxInteractionDistance          0.0025;
                writeReferredParticleCloud      no;
                pairModel pairSpringSliderDashpot;

                pairSpringSliderDashpotCoeffs
                {
                        useEquivalentSize       no;
                        alpha                   0.02;
                        b                       1.5;
                        mu                      0.10;
                        cohesionEnergyDensity   0;
                        collisionResolutionSteps 12;
                };

...
```

Continues in the next slide (9/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

Particles-walls collision model. Forces between particles and walls, interacting with a spring, slider, damper model

Particles-walls collision model parameters and sub-models.

```
...

        wallModel wallSpringSliderDashpot;

        wallSpringSliderDashpotCoeffs
        {
            useEquivalentSize           no;
            collisionResolutionSteps    12;
            youngsModulus               1e8;
            poissonsRatio               0.23;
            alpha                       0.01;
            b                           1.5;
            mu                          0.09;
            cohesionEnergyDensity       0;
        };

        U       U.air;
    }

...
```

Remember to use the name of the continuous phase set in the dictionary transportProperties

Continues in the next slide (10/11)

Selecting physical properties and advanced physics

- The dictionary *cloudProperties* requires special attention, as it is in this dictionary where we set the interaction models and injection models of the particles. If you are using the **DPM** approach, the dictionary should look like this one:

```
...

        StandardWallInteractionCoeffs
        {
                type rebound;
                e    0.97;
                mu   0.09;
        }

        heatTransferModel none;

        surfaceFilmModel none;

        stochasticCollisionModel none;

        radiation off;

        cloudFunctions{}

}
```

Heat transfer model. → heatTransferModel

Surface film model. → surfaceFilmModel

Stochastic collision model. → stochasticCollisionModel

Radiation model. → radiation

cloudFunctions ← Specialized function objects

Continues in the next slide (11/11)

Selecting physical properties and advanced physics

- You can initialize the initial position of the particles manually. To do so, you will need to use a manual injection model in the dictionary `cloudProperties`, as follows,

Manual injection model.

File with the initial position of the particles.

Particle distribution model.

```
injectionModels
{
        injection_model_name
        {
                type                    manualInjection;
                massTotal               0;
                parcelBasisType         fixed;
                nParticle               1;
                SOI                     0;
                positionsFile   "cloudPositions";
                U0              ( 0 0 0 );
                sizeDistribution
                {
                        type        fixedValue;
                        fixedValueDistribution
                        {
                                value   0.0025;
                        }
                }
        }
}
```

Selecting physical properties and advanced physics

- The file `cloudPositions` should be in ascii format and formatted as follows,

```
(

( x1    y1    z1 )
( x2    y2    z2 )
( x3    y3    z3 )
( x4    y4    z4 )
( x5    y5    z5 )

…
…
…

( xn    yn    zn )

);
```

- Where xn, yn and zn are the cartesian coordinates of each particle.
- You can add as many particles as you like.
- Remember, you need to create this file and must be located in the directory constant.

Selecting phasic boundary conditions

- In the directory **0**, you will find the dictionaries used to define boundary conditions and initial conditions of the continuous phase.
- When you create the dictionaries for the boundary and initials conditions of the continuous phase, you use the same naming convention as in the dictionary *transportProperties*.
- If in the dictionary *transportProperties* you named the continuous phase as air (continuousPhaseName air), you will need to create the following dictionaries for the continuous phase:
    - *U.air*
    - *p*

Selecting phasic boundary conditions

- If you are using a turbulence model, you should also create the dictionaries for the boundary and initials conditions for the turbulence field variables of the continuous phase.
- For instance, if you are using the $\kappa - \epsilon$ turbulence model for the continuous phase, you should create the following dictionaries in the directory **0**:
    - *epsilon.air*
    - *k.air*
    - *nut.air*

Roadmap

1. ~~Introduction to multiphase flows~~
2. ~~Modeling approaches for multiphase flows~~
3. ~~Governing equations and interfacial momentum transfer models~~
4. ~~Multiphase solvers in OpenFOAM~~
5. ~~Selecting physical properties, phase interaction, and advanced models~~
6. **Final remarks – Tips and tricks**
7. Additional tutorials

Final remarks – Tips and tricks

Challenges of simulating multiphase flow

- Simulating multiphase flows is not at easy task.
    - More than one working fluid.
    - Unsteady nature of the system.
    - Accurate interface tracking.
    - Different spatial-temporal scales.
    - Arbitrary particle shape.
    - Particle-particle interactions.
    - Mass transfer and chemical reactions.
    - Turbulence.
    - Many models involved (drag, lift, heat transfer, turbulence dispersion, frictional stresses, collisions, kinetic theory, and so on).

Your goal, model and predict detailed behavior of these flows using the less wrong models.

Final remarks – Tips and tricks

Guidelines and remarks

- Most of multiphase flows are transient in nature, so you need to use transient methods, and this makes multiphase simulations expensive.
- Most of the time is not possible to find a solution using a steady approach.
- Mesh quality is of paramount importance, so try to always get a good quality mesh.
- Also, a finer mesh will resolve better the spatial scales.
- If you are working with separated systems try to use hexahedral meshes, they resolve better the discontinuity between phases and approximate better the gradients.
- Control the growth rate factor of the mesh, a factor of 1.2 is a good choice.
- Choose a time-step to get a CFL less than 1.0 and if you can afford it, try to use a CFL less than 0.5
- In general, you need to use a small time-step and a fine mesh to resolve well the spatial-temporal scales.

Final remarks – Tips and tricks

Guidelines and remarks

- Generally speaking, working with separated systems using the VOF method is easier than working with dispersed system using Eulerian-Eulerian methods.
- In dispersed systems there are too many models that need to be calibrated.
- In the VOF method (and every solution method), higher order discretization schemes capture better the interface, as do finer grids.
- Sharper interface can be maintained by using explicit schemes for the phasic volume-of-fluid, the drawback is that explicit schemes requires small time-steps due to stability constrains.
- If surface tension is important, use explicit schemes for the phasic volume-of-fluid.
- Calculation of curvature based on volume-of-fraction can be inaccurate and cause convergence issues in problems dominated by surface tension, use smooth slope limiters, tight convergence criterion and high numerical precision (double precision).

Final remarks – Tips and tricks

Guidelines and remarks

- Eulerian-Lagrangian methods are an alternative to Eulerian-Eulerian multiphase modeling.
- The same physics is essentially modeled but using different closure relations.
- Depending on the number of particles you are trying to simulate, Eulerian-Lagrangian solution methods can be computationally expensive, so the Eulerian-Granular approach is a good alternative.
- The MPPIC Eulerian-Lagrangian is less computational expensive than the DPM method.
- Eulerian-Granular and Eulerian-Lagrangian methods are applicable from dense to dilute particulate flows.
- For volume fraction less than 60%, the Eulerian-Granular approach will work fine.
- Eulerian-Lagrangian methods will perform well (solver speed and accuracy), if the domain volume fraction is less than 20%.

Final remarks – Tips and tricks

Guidelines and remarks

- If you are interested in using turbulence models in dispersed systems, use the standard $\kappa - \epsilon$ model.
- Always tune the closure relations (models) for the specific applications.
- For dispersed systems (gas-liquid, gas-solid and gas-solid mixtures), usually it is enough to use drag and virtual-mass models.
- The Schiller and Naumann model for drag forces and the constant model for virtual mass are good choices for gas-liquid mixtures.
- The Gidaspow model for drag forces and the constant model for virtual mass are good choices for gas-solid mixtures using the Eulerian-Granular kinetic theory of granular flows (KTGF) approach.
- The Ergun model for drag forces is a good choice for gas-solid mixtures using Eulerian-Lagrangian methods.
- Higher order discretization schemes and fine meshes give more realistic bubble shapes.

Final remarks – Tips and tricks

Guidelines and remarks

- Typical under relaxation factors are (if you need to use them): 0.3 for pressure, 0.7 for momentum, 0.7 for turbulence, and 0.7 for volume fraction.
- Do not forget to turn on gravity.
- Also, do not forget to use a bubble, droplet or particle diameter model.
- For Eulerian-Lagrangian methods you will need to initialize the initial position of the particles or define an injection model.
- For Eulerian-Eulerian methods use an interface compression velocity of 1.0
- If you are interested in modeling wall adhesion, capillary flows, and contact angle, you will need to impose the contact angle as a boundary condition.
- Also, you might need to set walls boundary conditions to slip and add a frictional force.
- Remember to always compute the mean values of the field variables, that is the best way to compare with experimental results.

Guidelines and remarks

- Tricks such as increasing the viscosity or switching to 1^{st} order discretization methods are not recommended when dealing with multiphase flows.
- If you are using custom initialization of field variables, remember to always keep a backup of the original files.
- When setting boundary and initial conditions for turbulent blows, use the primary phase properties to compute the turbulent quantities.
- Always remember to check the scales of the domain.
- Adaptive time-step can raise stability problems. It is better to set a fix time-step.
- If you have outlets, from time to time it may happens that the volume fraction accumulates at the outlet (for instance take a look at the flush case). To avoid this, you will need to use a fine mesh, force the flow at the outlet and use turbulence models.
- Free surface flows or separated flows can be simulated using steady solvers (however we do not recommend it).

Final remarks – Tips and tricks

Guidelines and remarks

- Choose wisely your solution strategy:
  - If you are working with separated systems, use the VOF solution method. Remember, the VOF is not appropriate if the interface length is small compared to a mesh cell.
  - If you are working with dispersed systems, use the Eulerian-Eulerian or Eulerian-Lagrangian approach.
  - If you are working with gas-liquid mixtures use the Eulerian-Eulerian approach.
  - If you are working with gas-liquid mixtures and you are interested in interface tracking use the Eulerian-Eulerian-VOF approach.
  - If you are working with gas-solid or liquid-solid mixtures you can use the Eulerian-Granular kinetic theory of granular flows (KTGF) approach or the Eulerian-Lagrangian solution method.

Final remarks – Tips and tricks

Main takeaways

- Most of multiphase flows are transient, therefore multiphase simulations are time consuming.
- Use discretization schemes second order accurate in space and time.
- Try to keep the CFL number below 1.
- Preferably use a constant time-step.
- Use TVD schemes and gradient limiters for bounded quantities (e.g., temperature, concentration).
- Do not use aggressive slope limiter to compute local curvature.
- Mesh quality is of paramount importance.
- It is wise to always compute mean quantities.
- Choose wisely your solution approach.
- Calibrate your models.
- When interpreting the results use common sense and be skeptical.

Final remarks – Tips and tricks

My simulation is always exploding

- If after launching the simulation it mysteriously crash, the first thing you need to do is to read the screen.
- If the screen information is not enough, try to use a 1st order method, as follows:
    - Set the discretization scheme of the convective terms to upwind.
    - Set the discretization scheme of the diffusive terms to Gauss linear limited 0.5
    - Set the discretization scheme of the gradient terms to cellLimited Gauss linear 1.0
    - Set the nHat gradient to Gauss linear or leastSquares.
    - Set the temporal discretization scheme to euler.
    - Set the number of correctors to nCorrectors 5, and nNonOrthogonalCorrectors 3.
    - Use Newton-Krylov type linear solvers with a tight convergence criterion (in the order of 1e-6 for the absolute tolerance and 0 for the relative tolerance).
    - Use a CFL number of the order of 0.3 and set deltaT to a low value in order to avoid jumps during the first iterations.

Final remarks – Tips and tricks

My simulation is always exploding

  - If you are using under relaxation, use low values during the first 50 iterations (in the order of 0.05) and then increase the values gradually one at a time and monitor your solution (up to 0.2).
  - Usually, the most critical URF are density and energy.
  - You can increase pressure URF up to 0.2, and momentum URF up to 0.7, but always monitor the solution.
- This is one of a hell stable numerical scheme.
- However, it is first order accurate. Use it only to get the simulation started.
- If the simulation keeps crashing after using this numerical scheme, you should check your boundary conditions, initial conditions, physical properties and model properties.
- You should also know the limitations of the solver and models you are trying to use.
- Use this method only to identify problems.
- For production runs use an accurate and bounded numerical scheme (2nd order).

Final remarks – Tips and tricks

Why multiphase flow simulations are not easy?

Reactor design

The objective of reactor design is to create the right conditions for reactions. The temperature and reactant species distribution, appropriate residence time and removal of products must be considered. Including the effect of a catalyst may be necessary. A comprehensive understanding of all the competing and interacting mechanisms is required to arrive at better designs and improved processes. **In particular, gas-solids reacting flows involve, not only complex interactions of granular materials with gas flow, but also phase-change, heterogeneous and homogeneous reactions, heat and mass transfer. Moreover, the spatial and temporal scales may vary over many orders of magnitude.** Thus modeling gas-solid reacting flows requires the integration of the best physics and chemistry models from various science and engineering fields with the most advanced computational algorithms. These algorithms must be scalable to large high-performance computers in order to bear on this important topic.

Excerpt from the preface of *“Computational Gas-Solids Flows and Reacting Systems: Theory, Methods and Practice”*
***May, 2010, Eds.* S. Pannala, M. Syamlal and T. O’Brien,**

Some useful bibliographical references

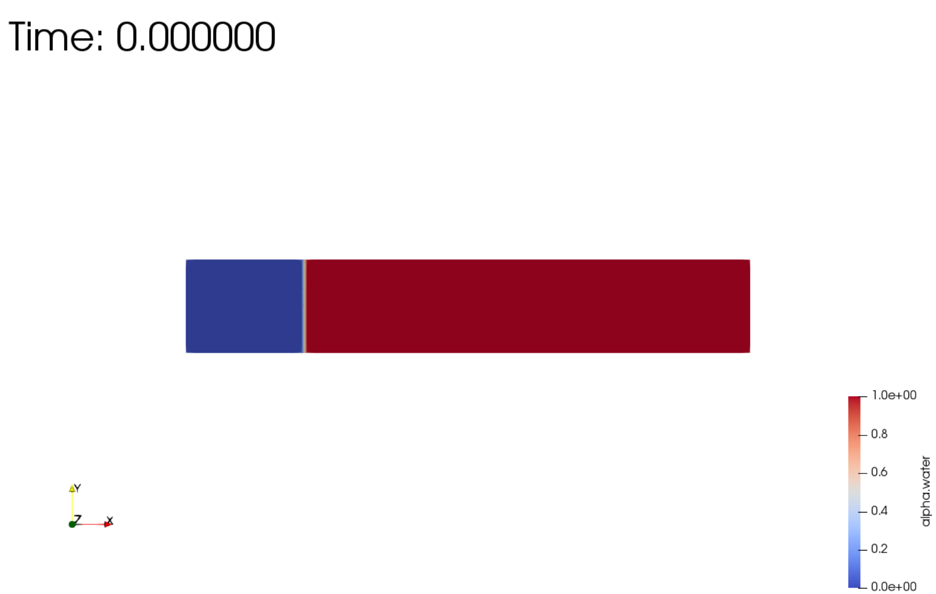
- **Computational Techniques for Multiphase Flows**
  G. H. Yeoh, J. Tu. 2009, Butterworth-Heinemann
- **Multiphase Flow Analysis Using Population Balance Modeling: Bubbles, Drops and Particles**
  G. H. Yeoh, C. P Cheung, J. Tu. 2013, Butterworth-Heinemann
- **Theory and Modeling of Dispersed Multiphase Turbulent Reacting Flows**
  L. Zhou. 2018, Butterworth-Heinemann
- **Computational fluid dynamics of dispersed two-phase flows at high phase fractions**
  H. Rusche. PhD Thesis. 2002. Imperial College, London.
- **Towards the numerical simulation of multi-scale two-phase flows**
  H. Marschall. PhD Thesis. 2011. Technische Universität München.
- **Derivation, Implementation, and Validation of Computer Simulation Models for Gas-Solid Fluidized Bed**
  B. van Wachem. PhD Thesis. 2000, TUDelft.
- **Gas-Particle flow in a vertical pipe with particle-particle intractions**
  J. L. Sinclair, R. Jackson AIChE Journal. Volume 35, Issue 9, 1473-1486, September 1989
- **Error analysis and estimation in the Finite Volume method with applications to fluid flows.**
  H. Jasak. PhD Thesis. 1996. Imperial College, London.
- **Turbulence Modeling for CFD**
  D. Wilcox. 2006, DCW Industries.

Roadmap

1. ~~Introduction to multiphase flows~~
2. ~~Modeling approaches for multiphase flows~~
3. ~~Governing equations and interfacial momentum transfer models~~
4. ~~Multiphase solvers in OpenFOAM~~
5. ~~Selecting physical properties, phase interaction, and advanced models~~
6. ~~Final remarks – Tips and tricks~~
7. **Additional tutorials**

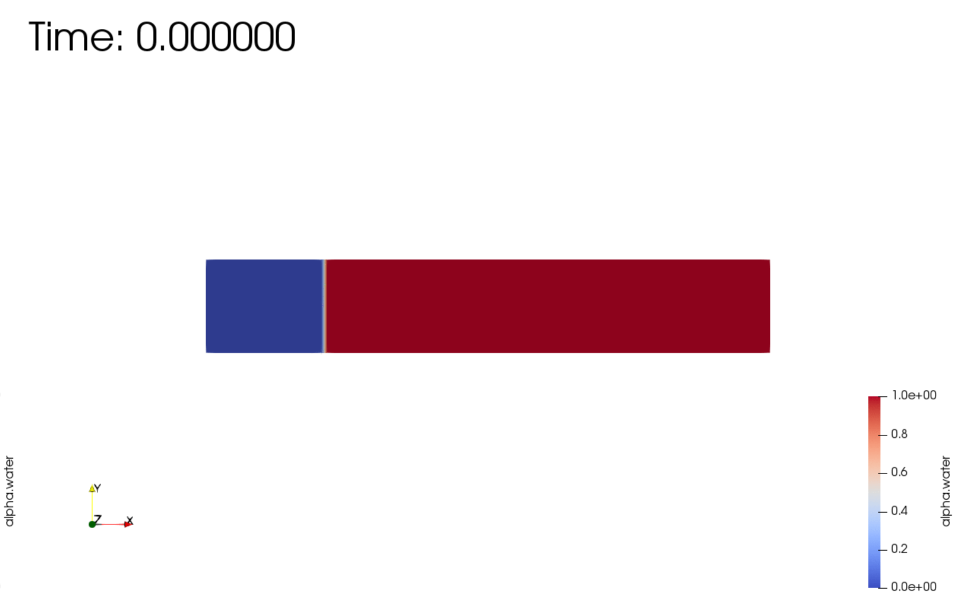
Additional tutorials

Capillary effect – VOF with contact angle



interFoam with contact angle 10 (hydrophilic surface)

http://www.wolfdynamics.com/training/mphase/image28.gif



interFoam with contact angle 135 (hydrophobic surface)

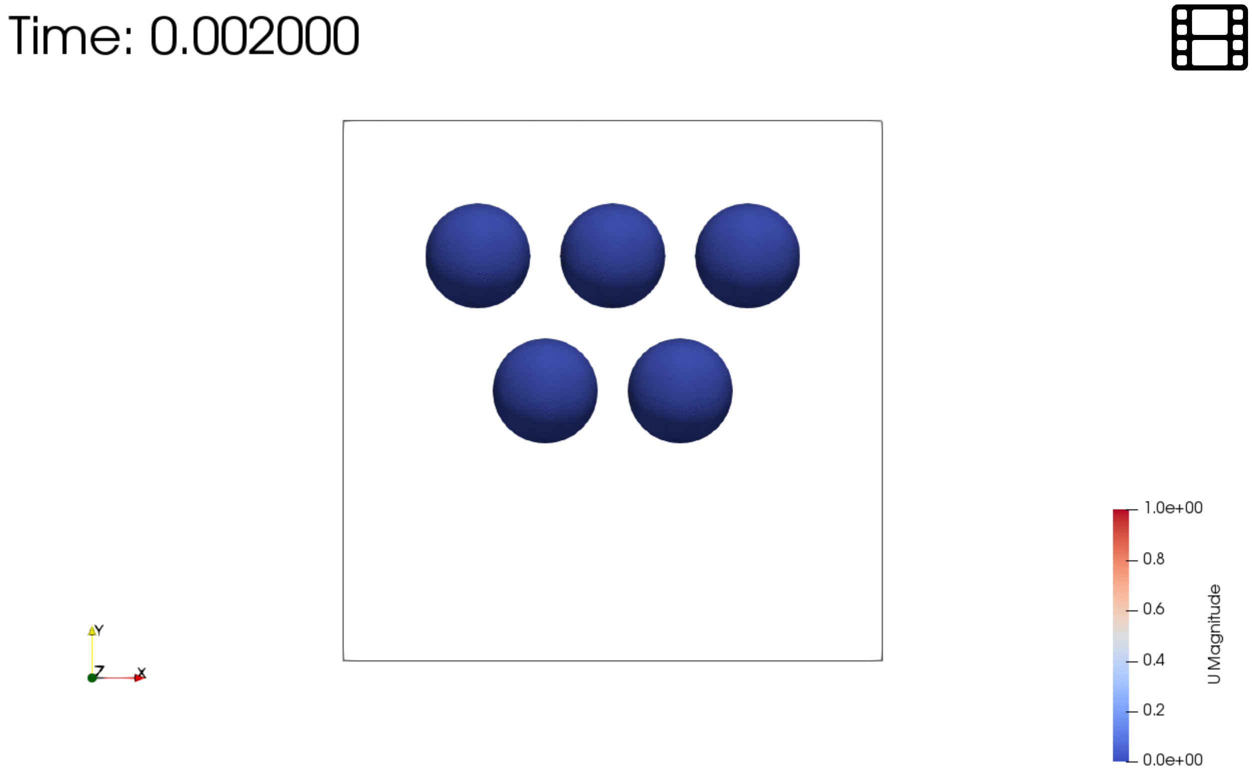
http://www.wolfdynamics.com/training/mphase/image29.gif

- This case is located in the directory:

`$TM/multiphase/additional_tutorials/capillary`

Additional tutorials

Particle-particle interactions with no hydrodynamic coupling



icoUncoupledKinematicParcelFoam – Particles to scale

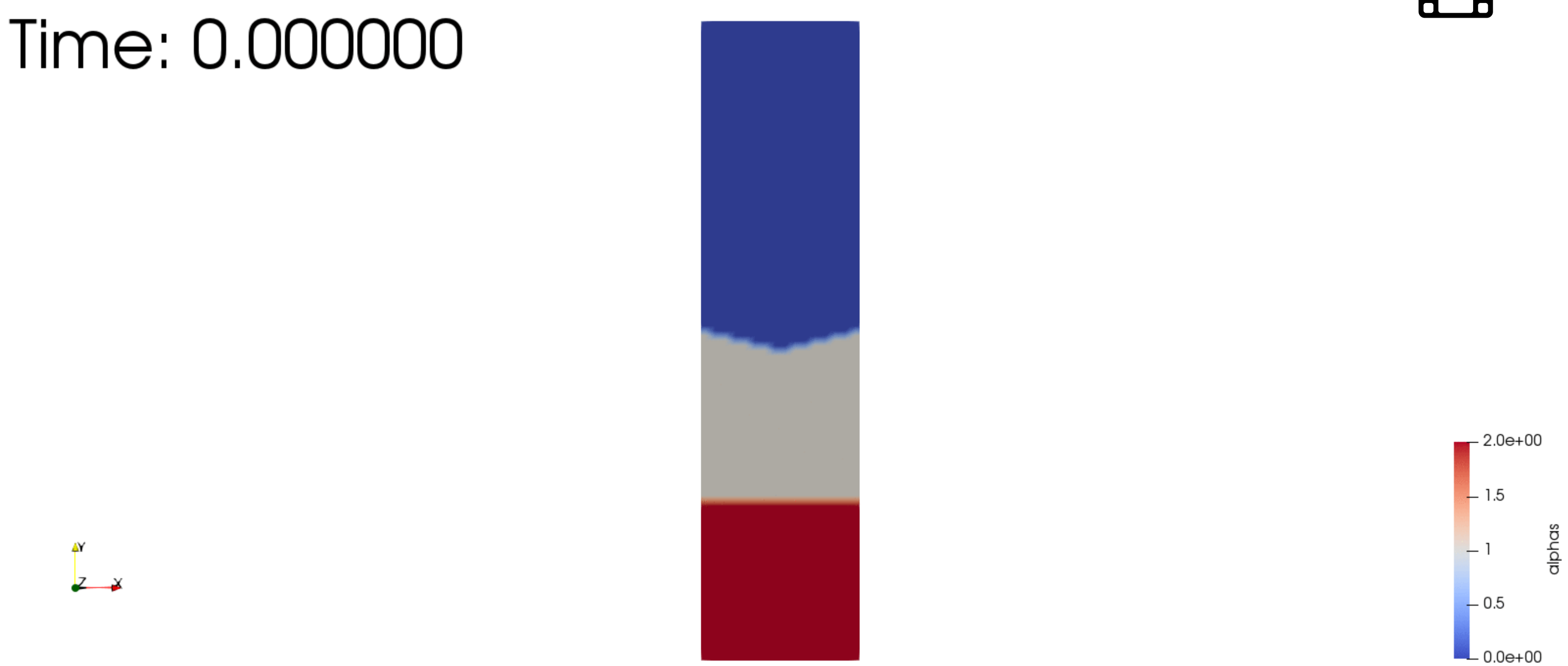
http://www.wolfdynamics.com/training/mphase/image21.gif

- This case is located in the directory:

$TM/multiphase/additional_tutorials/DEM

Additional tutorials

Rayleigh-Taylor instability – VOF with 3 phases



multiphaseInterFoam

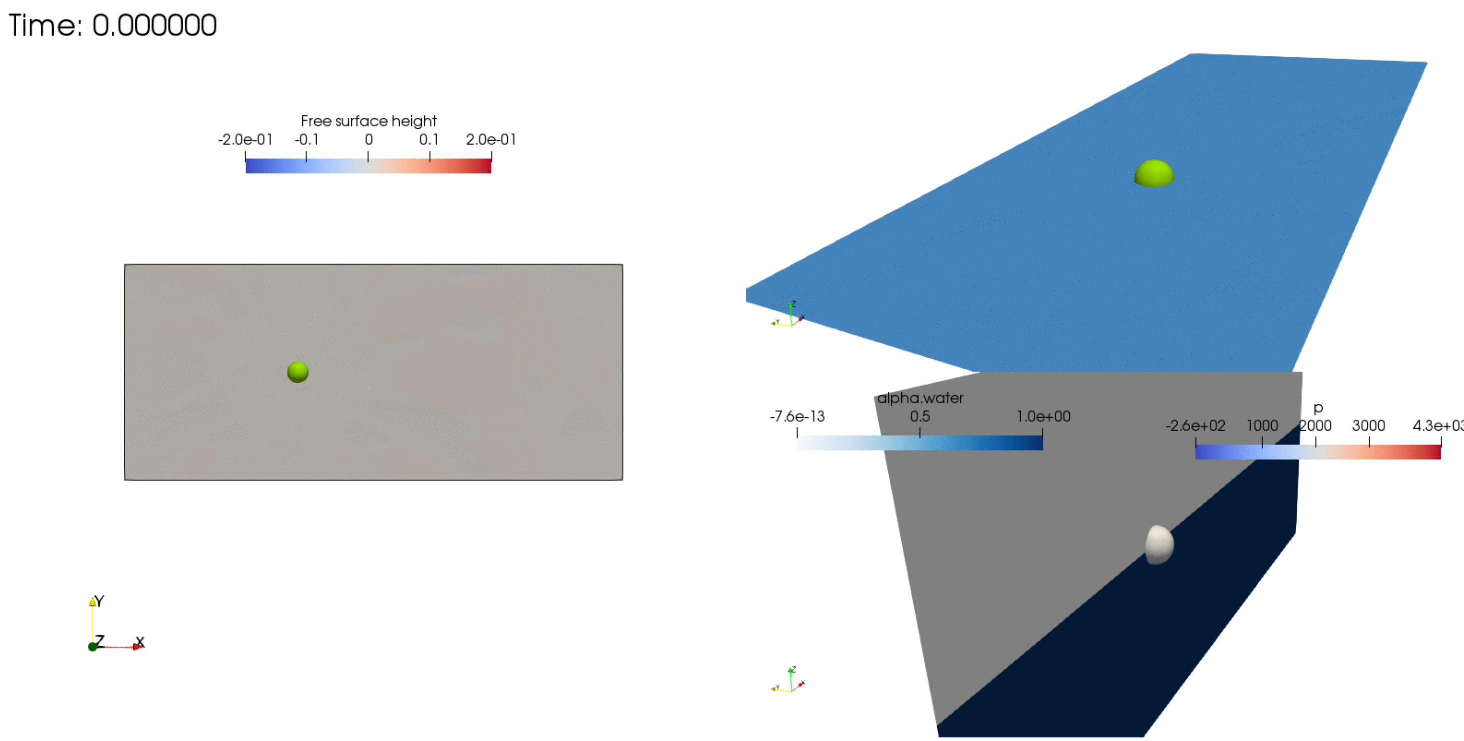
http://www.wolfdynamics.com/training/mphase/image4.gif

- This case is located in the directory:

`$TM/multiphase/additional_tutorials/rayleigh_taylor`

Additional tutorials

Sphere in a towing tank – VOF with Free surface



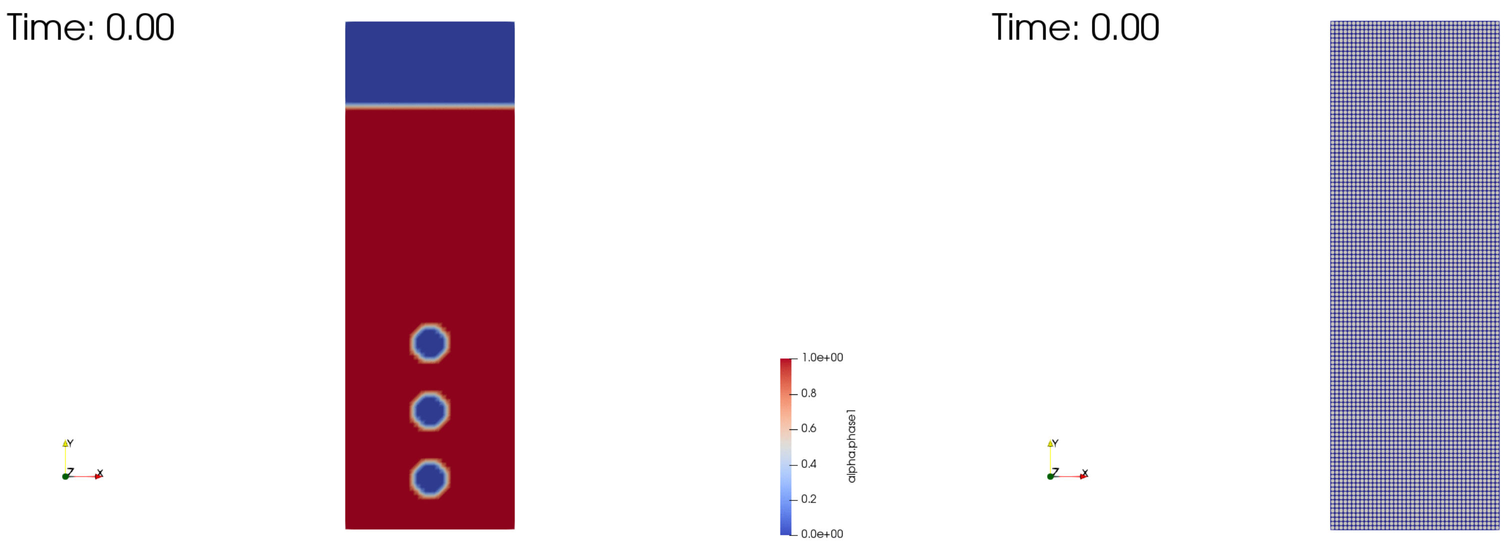
http://www.wolfdynamics.com/training/mphase/image15.gif

- This case is located in the directory:

`$TM/multiphase/additional_tutorials/sphere_towingTank`

Additional tutorials

Three rising bubbles – VOF with AMR



interDyMFoam with AMR

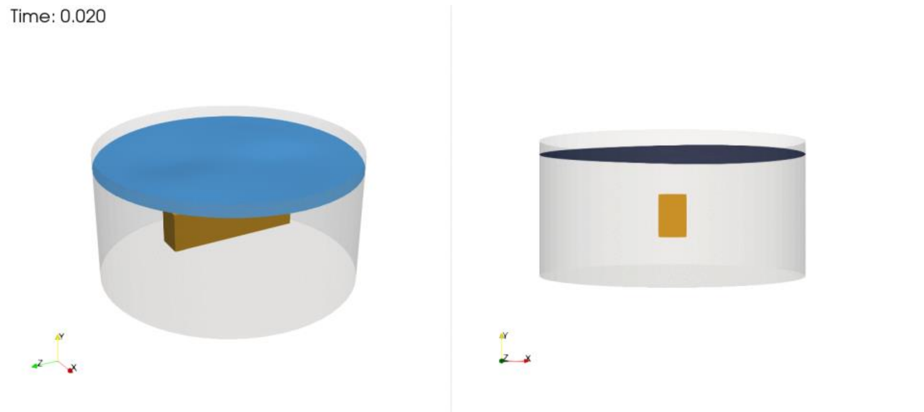
http://www.wolfdynamics.com/training/mphase/image2.gif http://www.wolfdynamics.com/training/mphase/image3.gif

- This case is located in the directory:

`$TM/multiphase/additional_tutorials/three_rising_bubbles`

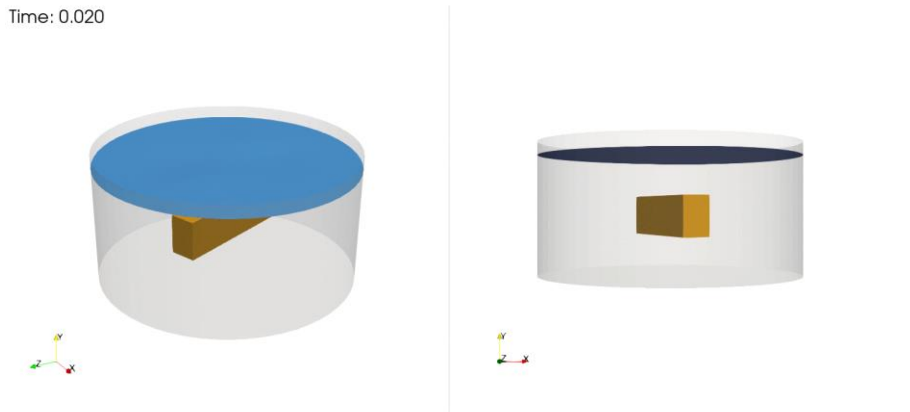
Additional tutorials

MRF vs. Sliding grids – VOF



interFoam with MRF

http://www.wolfdynamics.com/training/mphase/MRF1.gif



interFoam with dynamic meshes

http://www.wolfdynamics.com/training/mphase/sliding1.gif

- This case is located in the directory:

`$TM/multiphase/additional_tutorials/VOF_MRF_sliding`

Thank you for your attention

- We hope you have found this training useful, and we hope to see you in one of our advanced training sessions:
    - OpenFOAM® – Multiphase flows
    - OpenFOAM® – Naval applications
    - OpenFOAM® – Turbulence Modeling
    - OpenFOAM® – Compressible flows, heat transfer, and conjugate heat transfer
    - OpenFOAM® – Advanced meshing
    - DAKOTA – Optimization methods and code coupling
    - Python – Programming, data visualization, and exploratory data analysis
    - Python and R – Data science and big data
    - ParaView – Advanced scientific visualization and python scripting
    - And many more available on request
- Besides consulting services, we also offer '**Mentoring Days**' which are days of one-on-one coaching and mentoring on your specific problem.
- For more information, ask your trainer, or visit our website http://www.wolfdynamics.com/

Be collaborative, be innovative, be cloud

wolf dynamics

multiphysics simulations, optimization & data analytics

Let’s connect

guerrero@wolfdynamics.com

www.wolfdynamics.com