Towards numerical modeling of the drop collision with the solid surface

WARSAW UNIVERSITY OF TECHNOLOGY

S. Rek, T. Wacławczyk, J. Rokicki

Institute of Aeronautics and Applied Mechanics,
Warsaw University of Technology

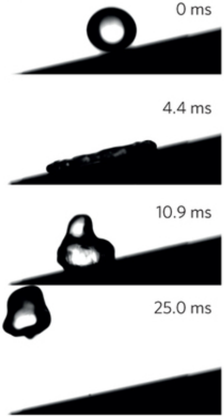
Warsaw, 01.27.2017

srek@meil.pw.edu.pl

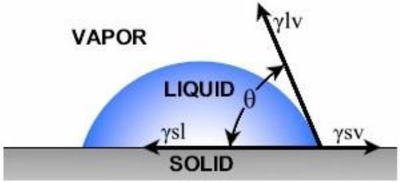
Outline

- Introduction
- Motivation - Huh's and Scrivens Paradox
- FVM solver comparison
- Volume of Fluid
- OpenFOAM - interFoam
- Plans and conclusion

Introduction



T = 513 K, tilt angle 20 deg
source: Directional transport of high-temperature Janus droplets mediated by structural topography J. Li,Y. Hou,Y. Liu, C. Hao, M. Li, M. K. Chaudhury, S. Yao, Z. Wang



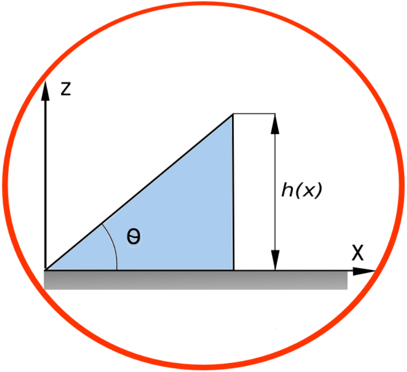
source: reme-heart instrument co.

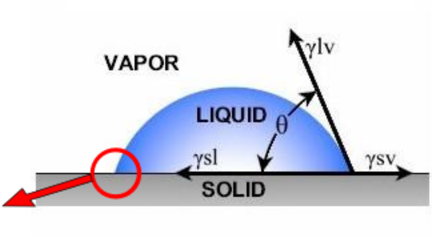
Young's equation

$$\gamma^{SV} = \gamma^{Sl} + \gamma^{lV} cos\theta$$

θ - the contact angle
γ^{SV} - the solid - liquid interfacial free energy
γ^{SV} - the solid - vapour interfacial free energy
γ^{SV} - the liquid - surface interfacial free energy

Introduction





source: reme-heart instrument co.

$$tan(\theta) = \frac{h(x)}{x}$$

$$h(x) = \theta x$$

Young's equation

$$\gamma^{SV} = \gamma^{Sl} + \gamma^{lV} cos\theta$$

θ - the contact angle

γ^{SV} - the solid - liquid interfacial free energy

γ^{SV} - the solid - vapour interfacial free energy

γ^{SV} - the liquid - surface interfacial free energy

Motivation - Huh's and Scrivens Paradox

WARSAW UNIVERSITY OF TECHNOLOGY

Assuming a non-slip condition at the surface of the solid surface gives rise to the - Huh and Scriven's paradox.

The rate of viscous dissipation:

$$\epsilon = \mu \left(\frac{du_x}{dz} \right)^2$$

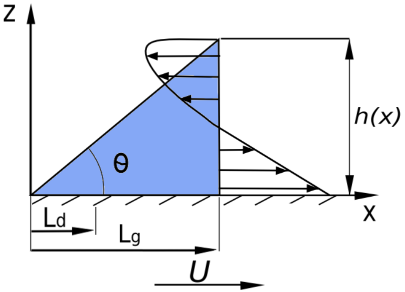
Typical vertical velocity gradient:

$$\frac{du_x}{dz} \approx \frac{U}{h(x)}$$

$$\epsilon = \mu \left(\frac{du_x}{dz} \right)^2 = \mu \left(\frac{U}{h(x)} \right)^2$$

$$D_{visc} = \int_{L_d}^{L_g} \int_0^h \epsilon \, dzdx$$

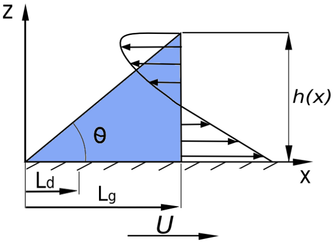
Coordinate system related to the surface of the droplet.



$$h(x) = \theta x$$

Motivation - Huh's and Scrivens Paradox

$$D_{visc} = \int_{L_d}^{L_g} \int_0^h \epsilon \, dzdx$$



$$\epsilon = \mu \left(\frac{du_x}{dz} \right)^2 = \mu \left(\frac{U}{h(x)} \right)^2$$

$$\int_0^h \epsilon \, dz = \int_0^h \mu \left(\frac{dU_x}{dz} \right)^2 dz = \mu \int_0^h \left(\frac{U}{h(x)} \right)^2 dz = \mu \frac{U^2}{h(x)}$$

$$D_{visc} = \mu \int_{L_d}^{L_g} \frac{U}{h(x)}^2 dx = \mu \int_{L_d}^{L_g} \frac{U}{\theta x}^2 dx = \mu \frac{U^2}{\theta} [\ln(x)]_{Ld}^{Lg}$$

$$D_{visc} = \mu \frac{U^2}{\theta} (\ln(L_g) - \ln(L_d)) = \frac{\mu U^2}{\theta} \left(\ln \frac{L_g}{L_d} \right)$$

$$D_{visc} \xrightarrow{L_d \to 0} \infty$$

The dissipation of energy is logarithmically diverging thus the rate of energy dissipation becomes infinite. No motion of the solid in contact with the liquid is possible.

"not even Herakles could sink a solid"

FVM advection - diffusion solver

WARSAW UNIVERSITY OF TECHNOLOGY

- The finite volume method (FVM) is a method for discretization and evaluation of partial differential equations in the form of algebraic equations

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla \cdot (\rho \mathbf{v} \phi) = \nabla \cdot (\Gamma^{\phi} \nabla \phi) + Q^{\phi}$$

- Solver was written in Python programming language.
  Structured 2D mesh grid is generated.
  Fallowing discretization methods were used:

for time derivative $\frac{\partial \phi}{\partial t}$ $\Rightarrow$ Euler first order implicit
for divergence term $\Rightarrow$ Gauss linear, upwind and central difference
for divergence diffusion term $\Rightarrow$ Gauss linear
grad scheme $\Rightarrow$ Gauss linear

To improve computational time LDU preconditioned was used
Recieved set of linear algebraic equations were solved using
BIConjugate Gradient STABilized method

Finite Volume Method

- Advection diffusion PDE

$$\frac{\partial(\rho\phi)}{\partial t} + \nabla\cdot(\rho\mathbf{v}\phi) = \nabla\cdot(\Gamma^{\phi}\nabla\phi) + Q^{\phi}$$

- After discretization in time and space we get equation for ϕ:

$$\phi^{n+1} + \frac{\Delta t}{V_c}\sum_{faces}\mathbf{v}\phi^{(n+1)}\mathbf{n}A - \frac{\Delta t}{V_c}\sum_{faces}\Gamma^{\phi}\nabla\phi^{(n+1)}\mathbf{n}A = \frac{Q^{\phi}}{\rho}\Delta t + \phi^{n}$$

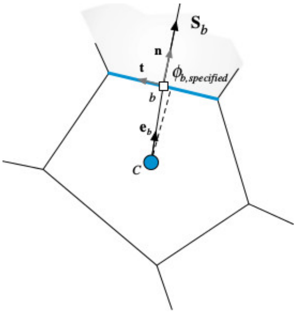
- For each finite volume we get a linear equation, the set of equations can be writen in matrix form.

$$Mx = B$$

And can be solved using one of many available methods like BICG

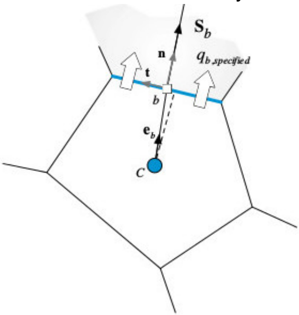
FVM Python advection diffusion solver

Two boundary conditions were implemented. Dirichlet and the Neumann boundary condition.



Value Specified (Dirichlet Boundary Condition)

$$\phi_b = \phi_{b,specified}$$



Flux Specified (Neumann Boundary Condition)

$$\mathbf{J}_b^{\phi} \cdot \mathbf{n}_b S_b = q_{b,specified} S_b$$

image source: F. Moukalled L. Mangani M. Darwish - The Finite Volume Method in Computational Fluid Dynamics, Springer International Publishing Switzerland 2016

OpenFOAM - Scalar Transport Foam

WARSAW UNIVERSITY OF TECHNOLOGY

- The scalarTransportFoam is a basic solver which resolves a transport equation for a passive scalar, using a user-specified stationary velocity field.

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\mathbf{u}\phi) - D_\phi \nabla^2(\phi) = S$$

- FOAM (Field Operation and Manipulation): Represent equations in their natural language

```
solve
(
fvm::ddt(φ)
+ fvm::div(phi, φ)
- fvm::laplacian(Dφ, φ)
== Source
);
```

- C++ toolbox
- object oriented:
  mesh
  fvmesh
  boundary conditions
  gemetricField
- fields <Type>:
  vector
  scalar
  tensor

Pyton FVM / OpenFOAM FVM Convection Diffusion

WARSAW UNIVERSITY OF TECHNOLOGY

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\mathbf{u}\phi) - D_\phi \nabla^2(\phi) = S$$

fvm::ddt(ϕ) + fvm::div(phi,ϕ) - fvm::laplacian(D_ϕ, ϕ) = Source

OpenFOAM ScalarTransportFoam:

ddt $\Rightarrow$ Euler implicit
div $\Rightarrow$ Gauss linear, upwind
laplace $\Rightarrow$ Gauss linear
grad scheme $\Rightarrow$ Gauss linear

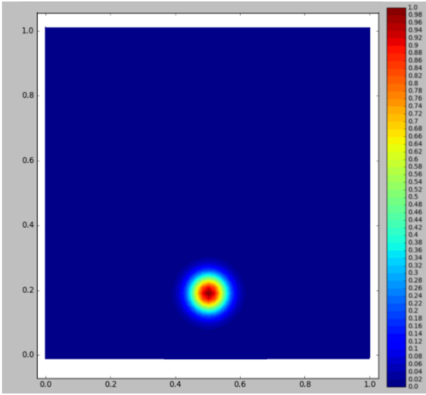
FVM Python Convection diffusion Solver:

ddt $\Rightarrow$ Euler implicit
div $\Rightarrow$ Gauss linear, upwind
laplace $\Rightarrow$ Gauss linear
grad scheme $\Rightarrow$ Gauss linear

Pyton FVM / OpenFOAM FVM Advection

WARSAW UNIVERSITY OF TECHNOLOGY

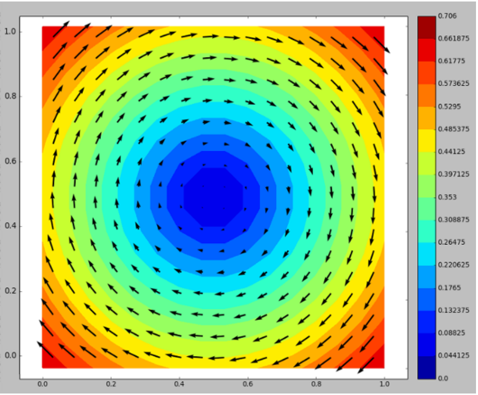
Scalar field:



for initial time t = 0 s

$$T_i = E^{(-400d^2)}$$

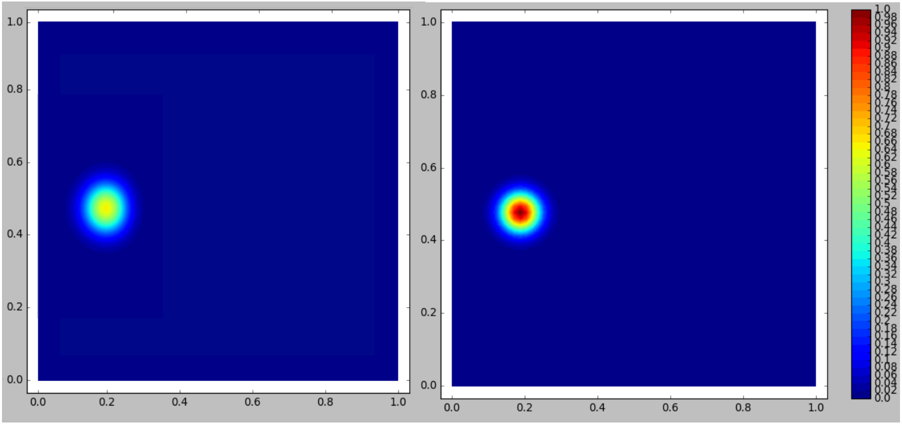
Stationary velocity field:



velocity field given by

$$\mathbf{v} = [y - 0.5, -x + 0.5]$$

Pyton FVM / OpenFOAM FVM advection

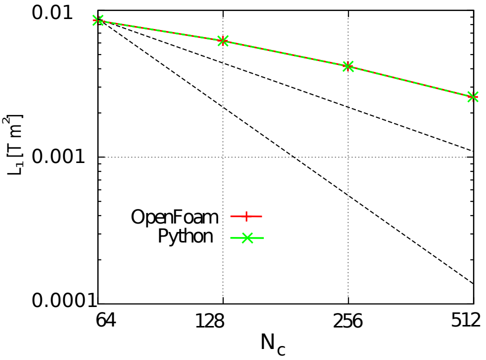


mesh 512 x 512 - 262144 cells

After rotation by $\pi/2$

Pyton FVM / OpenFOAM FVM Advection

WARSAW UNIVERSITY OF TECHNOLOGY

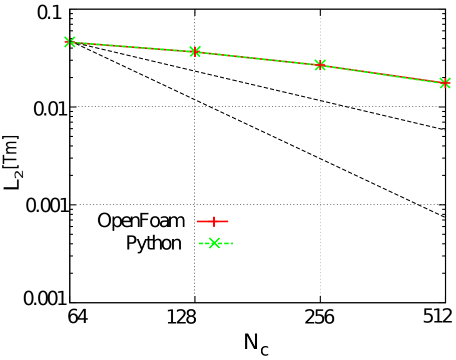


$$L_1 = \sum_{i=1}^{n} |(y_i - f(x_i))| A_i$$

$$L_2 = \sqrt{\sum_{i=1}^{n} (y_i - f(x_i))^2 A_i}$$

Dev	Norm 1 OF	Norm 1 PY
512	0.002578	0.002551
256	0.004188	0.004182
128	0.006242	0.006236
64	0.008558	0.008562

Pyton FVM / OpenFOAM FVM Advection



$$L_1 = \sum_{i=1}^{n} |(y_i - f(x_i))| A_i$$

$$L_2 = \sqrt{\sum_{i=1}^{n} (y_i - f(x_i))^2 A_i}$$

Dev	Norm 2 OF	Norm 2 PY
512	0.017741	0.017731
256	0.026933	0.026889
128	0.036865	0.036827
64	0.046313	0.046303

One Fluid Model and Volume of Fluid Method

$$\frac{\partial(\rho\mathbf{u})}{\partial t} + \nabla\cdot(\rho\mathbf{u}\mathbf{u}) = -\nabla p + \rho\mathbf{f} + \nabla\cdot\tau + \sigma\kappa\nabla\, H_m$$

VOF is a free-surface modelling technique, i.e. a numerical technique for tracking and locating the free surface

$$\alpha = \frac{\int H_m dV}{\int dV} = \frac{V^1}{V}\ , \quad \frac{\partial H_m}{\partial t} + \mathbf{u}\nabla H_m = 0$$

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

with the following constraint

$$\sum_m^2 \alpha_m = 1$$

$$\rho = \sum \rho_m\phi_m\ ,\ \mu = \sum \mu_m\phi_m$$

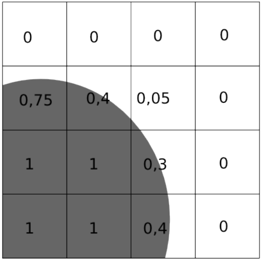


image source: Wikipedia

These properties are then used to solve a single momentum equation through the domain

Volume of Fluid - surface tension

- Young -Laplace equation - equation that describes the capillary pressure difference sustained across the interface between two static fluids

$$p_2 - p_1 = \sigma \left(\frac{1}{R_1} + \frac{1}{R_2} \right)$$

$$\nabla \alpha = \delta \mathbf{n}$$

where δ - Dirac delta function

- curvature

$$\kappa = -\nabla \cdot (\mathbf{n})$$

- the surface tension can be written in terms of the pressure jump across the surface (for two phases)

$$F_{vol} = \int_V \sigma \nabla \alpha \kappa \mathbf{n} dV \approx \sigma \delta \kappa \Delta V$$

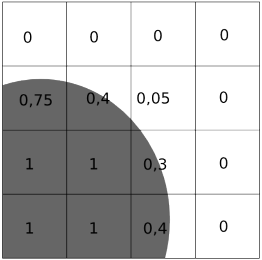


image source: Wikipedia

OpenFOAM solver: interFoam

- VOF
- incompressible
- transient
- multiphase (two fluids)
- immiscible
- isothermal
- interface capturing approach

Solved equations :
Continuity equation:

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

Momentum equations:

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \nabla \cdot \tau - \nabla p + \mathbf{F} + \sigma\kappa\nabla\alpha$$

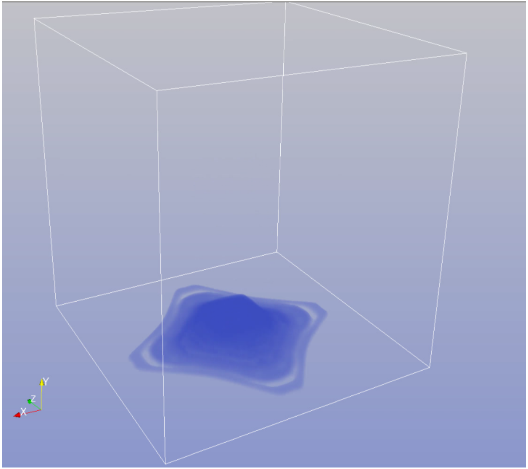
Volume of fluid:

$$\rho = \rho_l \alpha + (1 - \alpha)\rho_g$$

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\mathbf{u}\alpha) - \nabla(\alpha(1 - \alpha)\mathbf{u}_r) = 0$$

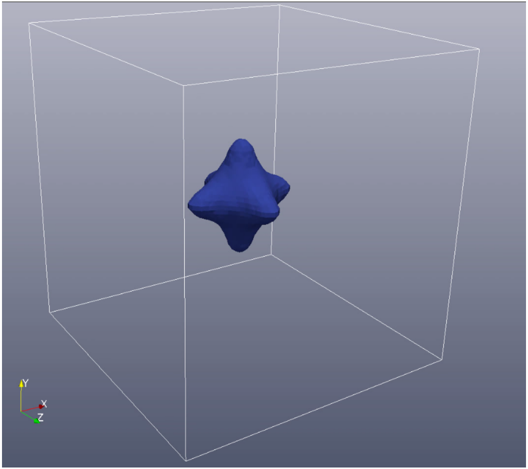
OpenFOAM solver: interFoam

WARSAW UNIVERSITY OF TECHNOLOGY



OpenFOAM solver: interFoam
turbulence $\Rightarrow$ laminar
deltaT $\Rightarrow$ 0.001 s
gravity $\Rightarrow$ 9.81 m/s^2
surface tension $\Rightarrow$ 0.07 Nm^{-1}
For liquid:
kinetic viscosity $\Rightarrow$ $10x10^{-6}$
density $\Rightarrow$ 1000 kgm^{-3}
For gas:
kinetic viscosity $\Rightarrow$ $1.48x10^{-5}$
density $\Rightarrow$ 1 kgm^{-3}
Discretization schemes:
ddt $\Rightarrow$ Euler implicit
div($\rho\phi$, U) $\Rightarrow$ Gauss linear, upwind
div(ϕ, H) $\Rightarrow$ Gauss vanLeer
div(ϕrb, H) $\Rightarrow$ Gauss linear
laplace $\Rightarrow$ Gauss linear
grad scheme $\Rightarrow$ Gauss linear
snGrad scheme $\Rightarrow$ corrected

OpenFOAM solver: interFoam



OpenFOAM solver: interFoam
turbulence ⇒ laminar
deltaT ⇒ 0.001 s
gravity ⇒ 9.81 m/s^2
surface tension ⇒ 0.07 Nm^{-1}
For liquid:
kinetic viscosity ⇒ $10x10^{-6}$
density ⇒ 1000 kgm^{-3}
For gas:
kinetic viscosity ⇒ $1.48x10^{-5}$
density ⇒ 1 kgm^{-3}
Discretization schemes:
ddt ⇒ Euler implicit
div($\rho\phi$, U) ⇒ Gauss linear, upwind
div(ϕ, H) ⇒ Gauss vanLeer
div(ϕrb, H) ⇒ Gauss linear
laplace ⇒ Gauss linear
grad scheme ⇒ Gauss linear
snGrad scheme ⇒ corrected

Conclusions and perspectives

WARSAW UNIVERSITY OF TECHNOLOGY

- comparison of results and mesh convergence study of OpenFOAM SSF and Python solver was made.
- start to solve more complex multiphase flows cases in OpenFOAM
- investigating models of dynamic contact line and its applications
- investigation of parasitic currents phenomena
- wetting angle as boundary condition