Assessment of a hybrid VOF two-fluid CFD solver for simulation of gas-liquid flows in vertical pipelines in OpenFOAM

Relatore
Prof. Luciano Galfetti

Candidato
Francesco Tocci
Matr. 804717

Anno Accademico 2015–2016
© Copyright Dicembre 2016.

Politecnico di Milano:
[www.polimi.it](http://www.polimi.it)

Scuola di Ingegneria Industriale e dell’Informazione:
[www.ingindinf.polimi.it](http://www.ingindinf.polimi.it)
Abstract

In the framework of OpenFOAM, multiphaseEulerFoam is used for simulation of incompressible multiphase flows in vertical pipelines. This model combines the Eulerian-Eulerian (two-fluid) approach with the Volume of Fluid (VOF) interface tracking model. The results from the coupled model are compared to experimental data.

From the physical point of view the hybrid approach is not problematic since the VOF model uses an indicator function for tracking the interface between the phases which has the same meaning as a volume fraction variable in the two-fluid model. The hybrid model is obtained via the addition of interface capturing on top of an Eulerian two-fluid model. Averaged mass and momentum conservation equations are applied across the entire domain to describe the time-dependent motion of both phases and an interface sharpening algorithm is used to obtain sharp interface regions.

The results from 3D transient simulations are compared to both time-dependent and time-averaged experimental data such as phase distributions and pressure drop. Two case studies are presented in which the vertical risers differ in length, diameter, superficial gas and liquid velocity and flow patterns. In the first case, an annular film of liquid is introduced at the inlet. The coupled solver is able to correctly predict the liquid holdup fraction and pressure drop. For the second test case, slug flow is predicted and a reasonably good agreement in the form of void fraction time series is obtained. Taylor bubbles are detected, separated by slugs of continuous liquid which bridge the pipe and contain small gas bubbles. By means of velocity field views the three regions of Taylor bubble, falling film and the wake are captured. There is a dependency of the flow behaviour on the mesh employed and the most suitable grid topology is provided.

The aim of this work is to gain a deeper understanding of transitional regimes of multiphase flows. This thesis highlights the application of a hybrid CFD model in prediction of complex gas-liquid flow in vertical pipes, without a priori knowledge of the flow pattern and thus overcoming the issue of flow regime selection.

Keywords: CFD, OpenFOAM, hybrid multiphase solver, multiphaseEulerFoam, coupled VOF two-fluid model, vertical pipelines, slug flow, annular flow.
Nel contesto di OpenFoam, il solutore multiphaseEulerFoam è utilizzato per la simulazione di flussi multifase, gas-liquido, in condotti verticali circolari. Questo solutore accoppia una metodologia Eulerian-Eulerian (two-fluid) con il metodo Volume of Fluid (VOF) per la risoluzione dell’interfaccia tra le fasi.

Dal punto di vista fisico l’accoppiamento non presenta problemi poiché la funzione che distingue le diverse fasi nel modello VOF non è nient’altro che la variabile frazione volumetrica nell’approccio two-fluid. Le equazioni mediate della conservazione della massa, con l’ipotesi di incomprimibilità, e del bilancio di quantità di moto sono definite sull’intero dominio per descrivere il comportamento delle fasi. L’interfaccia è ricostruito attraverso l’aggiunta di un termine, chiamato compressione artificiale, nell’equazione di conservazione della massa.

Simulazioni instazionarie e tridimensionali sono confrontate con dati sperimentali mediati a loro volta nello spazio, su sezioni del condotto, e nel tempo. Sono trattati due casi studio, differenti per dimensioni del condotto circolare e condizioni al contorno. I due casi simulati richiedono particolare attenzione per la scelta della mesh. Da precedenti risultati trovati in letteratura, viene impiegata la tipologia di mesh più adatta. Per il primo caso, dove un anello di liquido viene introdotto all’inlet, annular flow, i risultati della simulazione sono in accordo con gli esperimenti, mostrando valori di caduta di pressione e liquid holdup del tutto simili. Il secondo caso riguarda il cosiddetto slug flow. Il modello è in grado di simulare la formazione di tale regime e la comparazione con i dati sperimentali è ancora positiva. Andamenti nel tempo di void fractions mediate su diverse sezioni, le loro funzioni di densità di probabilità e l’immagine qualitativa del campo di moto sono i termini di paragone impiegati. La bolla di Taylor viene correttamente riprodotta così come la regione di film di liquido che la circonda.

Lo scopo di questo lavoro è di approfondire la conoscenza per la modellazione di flussi gas-liquido. In particolar modo la tesi sottolinea le capacità del solutore impiegato a trattare regimi di natura diversa, grazie alla sua natura ibrida.
# Contents

1 Introduction                                                                 1  
   1.1 Background                                                                 1  
   1.2 Research objective                                                        2  
   1.3 Thesis outline                                                            3  

2 Literature Review                                                            5  
   2.1 Multiphase flow: basic definitions and dimensionless numbers            5  
      2.1.1 Velocities                                                            5  
      2.1.2 Slip effects                                                         6  
      2.1.3 Froude number                                                        6  
      2.1.4 Morton number                                                        7  
      2.1.5 Eotvos number                                                        7  
      2.1.6 Dimensionless inverse viscosity number                               7  
      2.1.7 Slippage number                                                      7  
   2.2 Upward gas-liquid flow in vertical pipes                                 7  
      2.2.1 Flow regimes                                                         8  
      2.2.2 Flow pattern maps                                                    12  
      2.2.3 Transition mechanism                                                 15  
   2.3 CFD methodologies for two-phase flow                                     16  
      2.3.1 Direct numerical simulation                                           16  
      2.3.2 Eulerian-Lagrangian method                                            18  
      2.3.3 Eulerian-Eulerian method                                              19  

3 Coupling Euler-Euler two-fluid with VOF                                      21  
   3.1 Hybrid approach                                                          21  
   3.2 multiphaseEulerFoam                                                       22  
      3.2.1 Multifluid momentum equations                                         22  
      3.2.2 Time step                                                             27  
      3.2.3 Turbulence                                                            27  
      3.2.4 Solution procedure                                                    28  

4 Validation                                                                  33  
   4.1 Case study 1                                                               33  
      4.1.1 Case description                                                       33  
      4.1.2 Case setup                                                            34  
      4.1.3 Results                                                               38  
   4.2 Case study 2                                                               44
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Sketches of flow regimes for two-phase flow in vertical pipes</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Test section photographs of upward air-water flow</td>
<td>10</td>
</tr>
<tr>
<td>2.3</td>
<td>Visualization of wire-mesh sensor data from upward air-silicone oil flow</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>An image of the Taylor bubble bottom, the near wake and the film region</td>
<td>12</td>
</tr>
<tr>
<td>2.5</td>
<td>Slug unit</td>
<td>13</td>
</tr>
<tr>
<td>2.6</td>
<td>Weisman flow pattern map for vertical tubes, air-water</td>
<td>14</td>
</tr>
<tr>
<td>2.7</td>
<td>Taitel flow pattern map for vertical tubes, air-water</td>
<td>15</td>
</tr>
<tr>
<td>2.8</td>
<td>Schematic representation of interface-tracking and interface-capturing methods</td>
<td>17</td>
</tr>
<tr>
<td>2.9</td>
<td>Representation of geometrical scales in a bubbly flow</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>Flow chart of multiphaseEulerFoam</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Procedure to solve the phase volume fraction</td>
<td>31</td>
</tr>
<tr>
<td>3.3</td>
<td>Solution procedure of the phase continuity equation</td>
<td>32</td>
</tr>
<tr>
<td>4.1</td>
<td>Model geometry for simulation of vertical upward flow through a 50.8 mm diameter pipe</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>Cross-Section grid of the computational domain</td>
<td>35</td>
</tr>
<tr>
<td>4.3</td>
<td>Boundary conditions at the inlet</td>
<td>36</td>
</tr>
<tr>
<td>4.4</td>
<td>CFD results with OpenFOAM for the flow pattern in the vertical pipe at t = 10 s. Cells having a water volume fraction larger than 0.5 are coloured blue.</td>
<td>38</td>
</tr>
<tr>
<td>4.5</td>
<td>Time series for the void fraction in the CFD simulations with OpenFOAM</td>
<td>40</td>
</tr>
<tr>
<td>4.6</td>
<td>CFD results for the last 30D of the vertical pipe at t = 10 s.</td>
<td>41</td>
</tr>
<tr>
<td>4.7</td>
<td>Liquid holdup at different cross sections at t = 10 s.</td>
<td>42</td>
</tr>
<tr>
<td>4.8</td>
<td>Mixture velocity at different cross sections at t = 10 s.</td>
<td>43</td>
</tr>
<tr>
<td>4.9</td>
<td>Total liquid holdup fraction in the 50D long vertical pipe</td>
<td>44</td>
</tr>
<tr>
<td>4.10</td>
<td>3D geometry of the computational flow domain showing the location of the experimental measurement</td>
<td>46</td>
</tr>
<tr>
<td>4.11</td>
<td>Cross-sectional view of different computational grid used for mesh independent study</td>
<td>47</td>
</tr>
<tr>
<td>4.12</td>
<td>Simulation with OpenFOAM for the Taylor bubble with different grid number</td>
<td>48</td>
</tr>
<tr>
<td>4.13</td>
<td>Void fraction (see Eq. (4.4)) at the measurement section (0.5 m)</td>
<td>48</td>
</tr>
<tr>
<td>4.14</td>
<td>Computational mesh used for simulations by Abdulkadir et al. in [5].</td>
<td>49</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
<td>------</td>
</tr>
<tr>
<td>4.15</td>
<td>Cross-Section grid of the computational domain in OpenFOAM.</td>
<td>49</td>
</tr>
<tr>
<td>4.16</td>
<td>Simulation with OpenFOAM for the for the leading Taylor bubble rising through a 0.067 m internal diameter vertical pipe: instantaneous velocity field for the liquid phase.</td>
<td>51</td>
</tr>
<tr>
<td>4.17</td>
<td>Simulation with OpenFOAM for the fully developed slug flow: instantaneous void fraction.</td>
<td>52</td>
</tr>
<tr>
<td>4.18</td>
<td>Simulation with OpenFOAM for the fully developed slug flow: instantaneous velocity.</td>
<td>53</td>
</tr>
<tr>
<td>4.19</td>
<td>Simulation with OpenFOAM for a large trailing Taylor bubble and leading train of Taylor bubbles.</td>
<td>54</td>
</tr>
<tr>
<td>4.20</td>
<td>Comparison between experimental data (red) and CFD simulation with Star-CCM+ (black) from Abdulkadir et al.</td>
<td>54</td>
</tr>
<tr>
<td>4.21</td>
<td>Time series for the void fraction in different planes; comparison between experiments and CFD simulations with OpenFOAM.</td>
<td>55</td>
</tr>
<tr>
<td>4.22</td>
<td>PDF of cross-sectional average void fraction for the case of slug flow obtained from the experiments, from the CFD simulations with Star-CCM+ (in Abdulkadir et al. and with OpenFOAM.</td>
<td>56</td>
</tr>
<tr>
<td>4.23</td>
<td>Comparison between experiments and CFD simulations with Star-CCM+ (from [5]) and with OpenFOAM.</td>
<td>57</td>
</tr>
<tr>
<td>4.24</td>
<td>Time delay of a Taylor bubble passing along two different measuring locations in the pipe.</td>
<td>58</td>
</tr>
<tr>
<td>4.25</td>
<td>Comparison between experiments and CFD simulations with Star-CCM+ (from [5]) and with OpenFOAM for the leading Taylor bubble (Start-up).</td>
<td>59</td>
</tr>
</tbody>
</table>
## List of Tables

4.1 Results comparison: data with Fluent, SFC, and OLGA from [115] and with OpenFOAM. ........................................... 44
4.2 Number of cells used in different grids. ........................................... 45
4.3 Dimensionless numbers. ......................................................... 51
4.4 Flow development along the vertical riser in the simulation with OpenFOAM. ........................................... 60
4.5 Comparison between experiments and CFD simulations with Star-CCM+ (from [5]) and with OpenFOAM for the leading Taylor bubble: void fraction and liquid film thickness. ........................................... 61
Listings

3.1 Drag specification in \texttt{transportProperties} dictionary ............. 25
3.2 MULES constructor in \texttt{multiphaseSystem.C} .......................... 30
A.1 m4 Script to generate the \texttt{blockMesh} dictionary for the Case study 1. 65
A.2 m4 Script to generate the \texttt{blockMesh} dictionary for the Case study 2. 68
B.1 Changes in \texttt{UEqns.H} ............................................. 73
B.2 Changes in \texttt{createFields.H} ..................................... 73
B.3 Changes in \texttt{multiphaseEulerFoam.H} .............................. 73
B.4 Changes in \texttt{yPlus\_mixture} ....................................... 74
B.5 Changes in \texttt{yPlus\_mixture} ....................................... 74
C.1 \texttt{fvSolution} ....................................................... 75
D.1 \texttt{fvSchemes} ....................................................... 77
E.1 Velocity reconstruction in \texttt{pEqn.H} .................................. 79
Chapter 1

Introduction

This thesis is part of a collaboration project between Delft University of Technology and Dynaflow Research Group, under the supervision of Dr. ir. Frank Bos (Dynaflow Research Group BV, Zoetermeer) and Prof. dr. ir. Ruud Henkes (Shell, TU Delft, 3ME).

1.1 Background

Multiphase flows remain an area where the prediction through CFD is yet out of reach for the majority of applications. Multiphase flows present a unique challenge due to the variety of flow regimes. They are characterized by a broad range of scales, from the dispersed droplets at the microscale up to macroscale free surface flows. A complication is the regime dependency limitation of the existing multiphase CFD methodology: specific simulation methods are applicable only to a specific class of flows. For example, free surface flows, in which the dynamics of the interface are important to the overall physics, are typically modeled with a shared momentum equation and a phase fraction field using a sharp interface capturing method. However, such methods are not suited for the simulation of dispersed flows characterized by many small fluid particles which cannot be fully resolved on a computational mesh. In such cases, the two-fluid or Euler-Euler approach is taken. Here the phases are treated as interpenetrating continua, each governed by its own momentum equation and having exchange terms to account for interphase momentum transfer. This methodological segregation of flow regimes is acceptable for many classes of flows but presents severe limitations for problems in which both segregated and dispersed flow features are present or where transitions between flow types are critical to the phenomena of interest.

Recently, a hybrid CFD solver has been developed which aims to overcome the issue of regime dependency by combining the Euler-Euler multifluid method with sharp interface capturing in a regime flexible framework [89], [109].

In this context the interest is focused on vertical upward gas-liquid flows. Gas-liquid flows occur in many applications. The motion of bubbles in a liquid as well as droplets in a conveying gas stream are examples of gas-liquid flows. Bubble columns
are commonly used in several process industries. Atomization to generate small droplets for combustion is important in power generation systems. Also droplet formation and impaction are important in spray forming for materials processing. Steam-water flows in pipes and heat exchangers are very common in power systems such as fossil fuel plants and nuclear reactors.

Other important gas-liquid flows occur in pipelines. Here free gas may exist because it is originally present at the inlet, as in many gas-condensate pipelines, but it may also be due to the formation of gases originally dissolved in the liquid as the pressure along the pipeline falls. Depending on the liquid and gas flow rates and on the slope of the pipeline, one may observe a whole variety of flow regimes such as bubbly, stratified, wavy, slug and annular. Each one of them brings its own pressure drop and liquid hold-up fraction.

For example, in slug flow, solid surfaces such as pumps and tube walls are often subjected to large fluctuating forces which may cause dangerous vibrations and fatigue. It is therefore of great practical importance to be able to predict which flow regime would occur in a given situation, the operational limits to remain in the desired regime, and how the system would react to transients such as start-ups and shut-downs. The experimental effort devoted to this subject has been very considerable and the computational method described in this thesis is a promising tool for a better understanding of these problems.

1.2 Research objective

For the design and improved operations of oil and gas production systems simulation tools are used extensively. Most of the simulations are carried out with steady and transient one-dimensional tools, such as PIPESIM, OLGA and LedaFlow. For some applications, however, such as flow in bends, flow in splitters, flow in headers to facilities etc., the one-dimensional assumption limits the prediction accuracy. As an alternative, Computational Fluid Dynamics (CFD) can be used, either for 2D and 3D configurations. Available CFD packages include Fluent, START-CCM+, and OpenFOAM. The present study has focused on the verification and validation of CFD results for multiphase flow of gas and liquid through vertical pipe sections.

The problem originates from the failure of the VOF method as implemented in the CFD model in [115] with Fluent to predict slug flow in the risers. The first objective is then investigate the abilities of OpenFOAM for industrial applications which have dispersed and segregated flow regimes, such as slug flow in the risers. The object is initially restricted by the following constraints: the use of the VOF model, for comparison, and the simulations are performed with the same setups in OpenFOAM and Fluent [115] for the case study 1 and with the same setups in OpenFOAM and Star-CCM+ [5] for the case study 2. The latter constraint includes the use of a rather coarse grid to limit computation times.

From a first analysis, reported in 4.1, the disagreement between experiment and CFD is attributed to the deficiencies of the VOF method to simulate flows in which both dispersed and segregated flows are present. Therefore the constraint of using the VOF model is removed and a hybrid solver is further developed as described
The aim of this thesis becomes to build up a knowledge base, through numerical simulations and validation against experimental data, of a hybrid solver developed in the framework of OpenFOAM, that can be used for defining multiphase CFD procedures for industrial applications.

1.3 Thesis outline

The outline for the thesis will be as follows:

- Chapter 2: Basic theory on multiphase flow in general, description of upward gas-liquid flows in risers and the most common CFD approaches for multiphase flow.
- Chapter 3: Theory of the hybrid approach coupling two-fluid model with VOF.
- Chapter 4: Results and validation of the model.
Chapter 2

Literature Review

2.1 Multiphase flow: basic definitions and dimensionless numbers

A phase refers to the liquid, vapour, or solid state of matter. A multiphase flow is the flow of a mixture of phases such as gases (bubbles) in a liquid, or liquid (droplets) in gases, and so on. Dispersed phase flows are flows in which one phase consists of discrete particulates, such as droplets in a gas or bubbles in a liquid. On the other hand, separated flows are flows where the two phases are separated by a line of contact \[67\]. In other words, in a separated flow one can pass from one point to another in the same phase while remaining in the same medium. Here the scope is limited to gas-liquid mixtures where gas and liquid simultaneously flow through a vertical pipe. This section covers definitions and dimensionless numbers used to characterize gas-liquid flows.

2.1.1 Velocities

The superficial gas velocity \( U_{s,gas} \) is the gas velocity as if the gas was flowing in the pipe without liquids, in other words the total gas throughput (\( Q_{gas} \) in \( m^3/s \) at operating temperature and pressure) divided by the total cross sectional area of the pipe (\( A \)). For the superficial liquid velocity the same can be derived, and the simple expressions are:

\[
U_{s,gas} = \frac{Q_{gas}}{A} \quad U_{s,liquid} = \frac{Q_{liquid}}{A} \tag{2.1}
\]

The sum of the \( U_{s,gas} \) and \( U_{s,liquid} \) is the multiphase mixture velocity, and the expression is given by:

\[
U_m = U_{s,gas} + U_{s,liquid} \tag{2.2}
\]

The actual phase velocities are defined as:

\[
U_{gas} = \frac{Q_{gas}}{A_{gas}} \quad U_{liquid} = \frac{Q_{liquid}}{A_{liquid}} \tag{2.3}
\]
The difference and ratio for phase velocities is the slip velocity and the slip ratio, respectively:

\[ U_s = |U_{\text{gas}} - U_{\text{liquid}}| \quad S = \frac{U_{\text{gas}}}{U_{\text{liquid}}} \]  

(2.4)

### 2.1.2 Slip effects

The liquid fraction \( (\alpha_{\text{liquid}}) \) and the gas fraction \( (\alpha_{\text{gas}}) \) of the pipe cross sectional area \( (A) \) as measured under two-phase flow conditions are known as the liquid hold-up fraction \( (\lambda_{\text{liquid}}) \) and gas void fraction \( (\lambda_{\text{gas}}) \), respectively. The liquid hold-up fraction is the ratio of the pipe cross-section occupied by liquid \( (A_{\text{liquid}}) \) and the total pipe cross section \( (A) \). In a similar way the local gas void fraction is the ratio of the pipe cross-section occupied by gas \( (A_{\text{liquid}}) \) and the pipe cross section \( (A) \):

\[ \lambda_{\text{liquid}} + \lambda_{\text{gas}} = 1 \quad \alpha_{\text{liquid}} + \alpha_{\text{gas}} = 1 \]  

(2.5)  

(2.6)

\[ \lambda_{\text{liquid}} = \frac{A_{\text{liquid}}}{A} \quad \text{Liquid hold-up fraction} \]  

(2.7)

\[ \lambda_{\text{gas}} = \frac{A_{\text{gas}}}{A} \quad \text{Gas void fraction} \]  

(2.8)

In pipe flow the liquid volume fraction is usually defined as the fraction of the total volumetric flow rate that consists of liquid. The liquid hold-up is equal to the liquid volume fraction only under conditions of no-slip, when the flow is homogeneous and the two phases travel at equal velocities. Only in no-slip conditions the gas volume fraction is equal to the gas void fraction. Owing to the slip, the liquid hold-up will be larger than the liquid volume fraction in the majority of the conditions in the flow regimes and the gas void fraction will be smaller than the gas volume fraction. The lighter gas phase will normally move much faster than the liquid phase. The superficial velocity and the phase velocity are related by the hold-up fraction:

\[ U_{s,\text{gas}} = \alpha_{\text{gas}} U_{\text{gas}} \]  

(2.9)

The same relations apply for the other phase.

It should be noted that in CFD the volume fraction is normally defined differently from the description above: in CFD the volume fraction is taken the same as the holdup. This is also how we will use the volume fraction in the present study.

### 2.1.3 Froude number

The Froude number \( (Fr) \) is the ratio of the inertial force and the gravitational force for a particular phase; in other words, the ratio of the kinetic to the potential energy of the gas or the liquid as given in Eq. (2.10)

\[ Fr = \frac{u^2}{gD} \]  

(2.10)
2.1.4 Morton number

\[ Mo = \frac{g \eta^4}{\rho \sigma^3} \]  

(2.11)

where \( g \) is the gravity constant, \( \eta \) the viscosity \([Kg/ms]\), \( \rho \) the density and \( \sigma \) the surface tension \([N/m]\).

2.1.5 Eotvos number

\[ Eo = \frac{\rho g D^2}{\sigma} \]  

(2.12)

where \( D \) is the pipe diameter \([m]\).

2.1.6 Dimensionless inverse viscosity number

From the Eotvos number and the Morton number the dimensionless inverse viscosity number \((N_f)\) can be derived:

\[ N_f = \left( \frac{Eo^3}{Mo} \right)^{1/4} \]  

(2.13)

2.1.7 Slippage number

Abdelsalam et al. [1] proposed a new dimensionless number, the Slippage number, for gas-liquid flow in pipes. The number is defined as the ratio of the difference in the gravitational forces between slip and no-slip conditions to the inertial force of the gas:

\[ SL = \frac{(p_{TP} - p_H)gD}{pU_{s,\text{gas}}^2} \]  

(2.14)

In gas-liquid flow, the average two-phase flow mixture density or the slip-density, \( \rho_{TP} \), is different from the homogenous or no-slip density of the mixture, \( \rho_H \), due to slippage between the phases. The no-slip mixture density, \( \rho_H \), is simply calculated based on the ratio of the volumetric flow rate of the phases assuming no slippage between the phases. The slip density, \( \rho_{TP} \), is calculated based on the actual or measured liquid holdup.

2.2 Upward gas-liquid flow in vertical pipes

From a practical engineering point of view one of the major design difficulties in dealing with multiphase flow is that the mass, momentum, and energy transfer rates and processes are sensitive to the topology of the components within the flow. There is a complicated two-way coupling between the flow in each of the phases and the geometry of the flow [24]. The way in which the phases are distributed is called flow pattern and its occurrence depends mainly on the combination of different gas and liquid flow rates. Other parameters, like pipe orientation, pipe
diameter and fluid properties, play a role as well. An appropriate starting point is a phenomenological description of the geometric distributions or flow patterns that are observed in common multiphase flows. This section describes the flow patterns observed in vertical pipes and identifies a number of the instabilities that lead to transition from one flow pattern to another. The material in this section is extracted from various sources: [3], [24], [48], [13].

2.2.1 Flow regimes

Depending on the combination of gas and liquid superficial velocities, a specific flow regime may occur in vertical upward gas-liquid flow. We are primarily interested in four clearly distinguishable flow regimes, namely, bubbly, slug, churn and annular flows, which will be referred to as distinct regimes. To these, we also add three other regimes that share some of the features of at least two of the distinct regimes: bubbly-slug, slug-churn and churn-annular flows [88]. In past literature [100] such regimes have sometimes been referred to as transitional regimes, because, if the liquid flow rate in a pipe was kept constant while the gas flow rate was increased, they would be observed at the transition between two distinct regimes. For vertical upflow, as the amount of gas is gradually increased, the following four distinct regimes evolve: bubbly flow, slug flow, churn flow, and annular flow (Figs. 2.1, 2.3).

**Bubbly flow:** In this configuration, there is a continuous liquid phase and the gas phase is dispersed as bubbles within the liquid. The bubbles travel with a complex motion within the flow, may be coalescing and are generally of non-uniform size.

**Slug flow:** Characteristic bullet-shaped bubbles, often called Taylor bubbles, flow up the pipe surrounded by a thin film of liquid. The liquid slug body between the Taylor bubbles often contains a dispersion of smaller bubbles.

**Churn flow:** At higher velocities, the Taylor bubbles in slug flow break down into an unstable pattern in which there is an oscillatory motion or churning of liquid in the tube.

**Annular flow:** This configuration is characterised by liquid travelling as a film on the channel walls. Part of the liquid can also be carried as drops in the central gas core.

In Fig. 2.2 a visual identification of each flow pattern, including the transitional regimes, is shown. Bubbly-slug flow is characterized by the presence of a relatively large cap-shaped bubble and the churn flow shows an unstable slug since the liquid slug might be able to bridge the pipe diameter but it is continually penetrated by the gas [81]. In the semi-annular flow one may notice the existence of a gas core and a relatively uniform annular liquid film on the pipe wall as well as liquid slugs.

We now confine our attention to the slug flow since this regime with its formation and transition will be the key topic of the CFD simulations of this thesis.
2.2. Upward gas-liquid flow in vertical pipes

Figure 2.1: Sketches of flow regimes for two-phase flow in vertical pipes. From left to right: Bubbly, Slug, Churn and Annular Flow. (Taitel and Bornea, 1980)

2.2.1.1 Slug flow

Slug flow is characterised by an alternating flow of gas pockets and liquid slug bodies. Most of the gas-phase is concentrated in large bullet-shaped gas pockets, defined as Taylor bubbles (after Geoffrey Taylor, notable for his pioneer work on slug flow [35]) surrounded by a thin falling film of liquid [16]. The Taylor bubbles are separated by intermediate liquid slug bodies, which may contain small entrained gas bubbles [8].

In large diameter pipes, there is significant entrainment of gas from the Taylor bubble into the liquid slug region caused by the wall film plunging into the upward flowing slug of liquid. In smaller diameter pipes the slug of liquid is bubble free. The gas content in the slug region increases systematically with the pipe diameter [71]. Isolated Taylor bubbles rise almost uniformly in vertical pipes, occupying nearly the entire cross-section of the tube. As it reaches the bottom of the bubble, the annular film enters the liquid slug behind it with the possibility of creating a recirculation region known as the bubble wake [68].

Both the shape of the bubble trailing edge and the wake flow pattern depend on the fluid properties and the tube geometry [17], besides the flow conditions. If the separation distance between two Taylor bubbles is small enough, the motion and shape of the trailing bubble get largely affected by the flow in the wake of the leading one: the nose becomes distorted and wavy; its velocity increases and coalescence between bubbles will occur. Extensive investigations of these bubbles and slugs have been carried out in the past and several physically based models have been proposed for fully developed flow [39], [40], [23].

Fig. 2.4 shows an image of the Taylor bubble bottom from an experimental investigation through PIV [53] while Figs. 2.5a and 2.5b present a numerical
image of the flow around a Taylor bubble and a schematic model of the slug unit respectively \[68\]. The bubble velocity \(U_{TB}\), the radius of curvature at the nose \(R_N\), the interaction distance above the bubble \(Z_A\), the length needed to have fully developed annular liquid film \(Z^*\), the film thickness \(\delta\), the wall shear stress in the stabilized film \(\tau_W\), and the wake dimensions, represented by its length \(l_W\), are depicted in Fig. 2.5b.

In \[101\] Taitel and Barnea presented a review on the modeling of gas-liquid flow in pipes. As in \[99\] the review follows an approach based on Fernandes et al. \[40\], which establishes macroscopic balances to the two phases and quantifies the various physical parameters comprising those equations. In Campos et al. \[68\] different physical mechanisms inside the flow of a Taylor bubble are extensively assessed through dimensional analysis, following the work of White and Beardmore \[113\].

To accurately model slug flow it is first essential to understand the motion of single Taylor bubbles rising through stagnant liquids. Throughout its rise, the bubble is influenced by gravitational, inertial, viscous and interfacial forces. Fluid density, surface tension and viscosity also have an important effect on the flow regime in a pipe \[25, 60, 2\]. Szalinski et al. \[97\] found that in the air-water flow there is the relation between coalescence and breakup shifted towards more coalescence than in air-silicone oil flow. Bubbles in the air-water flow tend to be larger than in air-silicone oil flow at similar superficial velocities. These differences in coalescence intensity are obviously due to the different viscosities of the liquid phases. Hernandez et al. \[47\] studied the effect of the Morton number on Taylor bubble behaviour: for a higher Morton number there is more distortion.

Neglecting gas expansion, several authors, \[113, 31\] and \[98\], reported \(Fr = f(Eo, Mo)\) to be sufficient for the characterization of the rise of single Taylor bubble

**Figure 2.2:** Test section photographs of upward air-water flow exhibiting the visual flow pattern features: (a) Bubbly; (b) Spherical cap; (c) Stable slug; (d) Unstable slug; (e) Semi-annular; (f) Annular. (Rosa, Flora and Souza, 2012)
2.2. Upward gas-liquid flow in vertical pipes

Figure 2.3: Visualization of wire-mesh sensor data from air-silicone oil two-phase flow at liquid superficial velocity = 0.25 m/s and different gas superficial velocities. The gas phase is represented by bright and the liquid phase by dark colour. Left: virtual side projection, right: axial slice images. (Szalinski et al., 2010)

in stagnant liquid conditions.

Taylor bubbles have significant variations in the shape of their rear ends [14]. In low viscosity liquids they have an approximately flat bottom and a recirculation region below it. Small bubbles are torn off the rear end of the Taylor bubble and either move down or are reabsorbed into the rear end. As the liquid viscosity increases the rear of the bubble becomes rounded and there is no recirculation in the wake as reported by [113], [108] and [65].

Numerous authors have confirmed the importance of the diameter effect on flow regime [94], [82]: slug flows cannot be sustained in pipes with diameters larger than some limit [50], [77]. The behaviour of Taylor bubbles can also depend on the geometry in which they flow, particularly the inlets and outlets for the gas and liquid.

The rise velocity of bullet-shaped bubbles was first studied analytically by Duimitrescu [38] and Davies and Taylor [35] who determined the rise velocity to be equal to \( Fr \sqrt{gD} \). They proposed values of \( Fr \) of 0.351 and 0.328 respectively. Viana et al. [108] presented an equation for \( Fr \) based on \( Eo \) and a dimensionless inverse viscosity which they term the Buoyancy Reynolds number (\( Re_b = \sqrt{gD^3(\rho_l - \rho_g)\rho_l/\eta_l} \)).

Nicklin et al. [69] proposed an empirical relationship to describe the rise velocity of single Taylor air bubble in a static water column. Nicklin’s empirical relationship, given by Eq. (2.15), describes the translational velocity of a Taylor bubble, \( U_b \), as the sum of drift velocity, which is the velocity of a Taylor bubble in a stagnant liquid, plus the contribution of the mixture superficial velocity in the preceding slug.

\[
U_b = C_0(U_{s,gas} + U_{s,liquid}) + 0.35\sqrt{gD} \tag{2.15}
\]
A value of 1.2 for $C_0$ was suggested by Nicklin et al. [69].

It is outside the scope of this work is to provide more details regarding empirical correlations. This has been recently covered by Campos et al. [68], who reviewed the literature on vertical gas-liquid slug flow with Newtonian fluids, from 1943 to 2015. A comprehensive study on the modelling of the rise behaviour of single Taylor bubbles is carried out by Ambrose in [12].

2.2.2 Flow pattern maps

Flow pattern data are often represented on a two dimensional diagram in terms of system variables. The most common dimensional variables used are the superficial velocities (Sternling 1965, Wallis 1969), superficial momentum fluxs (Hewitt and Roberts 1969) or volumetric flux (Fig. 2.6). Since variables other than the superficial velocities are known to affect the flow, pattern maps of this kind are specific to a particular combination of fluids and geometry. These maps have failed to gain universal acceptance because, although they apply to the experimental configuration for which they were developed, they may not be accurate for different configurations [88], [13]. However, they are simple to use, and unlike the case of single-phase Newtonian flow where the single parameter of Reynolds number brings all flows together, it is by no means clear exactly which other variables should be included. Hence, even for the simplest duct geometries, there exist no universal, dimensionless flow pattern maps that incorporate the full, parametric dependence of the boundaries on the fluid characteristics.

The commonest way of constructing a flow map is to identify the flow pattern
2.2. Upward gas-liquid flow in vertical pipes

(a) Numerical image of the flow around a Taylor bubble. (Morgado, 2016)

(b) Schematic representation of the main hydrodynamic features assessed in the slug unit. (Morgado, 2016)

**Figure 2.5:** Slug unit.
at a set of conditions covering the field, and then to sketch in boundary lines separating the different patterns. Because of problems in correctly identifying flow patterns, it often happens that a few experimental points lie on the wrong side of these lines, and the lines would be better regarded as transition zones. This should always be remembered when using maps on which only the boundary lines appear [13]. Besides these difficulties there are a number of other troublesome questions. In single phase flow it is well known that an entrance length of 30 to 50 diameters is necessary to establish fully developed turbulent pipe flow. The corresponding entrance lengths for multiphase flow patterns are less well established and it is quite possible that some of the reported experimental observations are for temporary or developing flow patterns. Moreover, the implicit assumption is often made that there exists a unique flow pattern for given fluids with given flow rates. It is by no means certain that this is the case [24]. Consequently, there may be several possible flow patterns whose occurrence may depend on the initial conditions, specifically on the manner in which the multiphase flow is generated. In vertical upward flows, the pressure decreases with increasing elevation, but also drops due to viscous friction; this implies a decrease in gas density and a commensurate increase in the gas superficial velocity with increasing elevation. Consequently, the flow pattern at a certain location in the pipe would not only depend on inlet conditions, but also on the coupled actions of bubble breakup/coalescence and pressure drop. For some conditions, gas-liquid flows may not reach full development even not for very long axial pipe lengths. Among the wide variety of flow pattern maps for vertical flow found in the literature, Taitel et al. [100] suggested physically based mechanisms which underlie each transition and modelled the transitions based on these mechanisms. Figs. 2.7a and 2.7b report their results.
2.2. Upward gas-liquid flow in vertical pipes

2.2.3 Transition mechanism

In this section we concentrate on the qualitative features and underlying instabilities, that give rise to transitions, of the boundaries between the following flow patterns: bubble-slug and slug-churn.

2.2.3.1 Bubble-slug transition

Transition from the condition of dispersed bubbles observed at low gas rates to slug flow requires a process of agglomeration or coalescence. As the gas rate is increased, the bubble density increases. This closer bubble spacing results in an increase in the coalescence rate. However, as the liquid rate increases, the turbulent fluctuations associated with the flow can cause breakup of larger bubbles formed as a result of agglomeration. If this breakup is sufficiently intense to prevent recoalescence, then the dispersed bubble pattern can be maintained [100]. Hence in any bubbly flow, two opposing processes are at work: bubble coalescence as a result of collisions between bubbles, and bubble break up as a result of turbulence in the liquid phase.

At low liquid velocities, turbulence is small, so coalescence dominates and the equilibrium bubble size is large; these larger bubbles have distorted, constantly varying shapes, and rise in zigzag or spiral motion. At higher velocities turbulence is increased, and the equilibrium bubble size is smaller, reducing collision. The work of Song et al. [93] indicates that the void fraction at transition depends on the bubble size. Kytomaa and Brennen [62] and Cheng et al. [29] have considered a wider range of pipe diameters, 25 mm to 100 mm, and show that the critical void fraction depends on the ratio of mean bubble size to pipe diameter.

With increases in gas flow rate, at low liquid rates, the rate of agglomeration to larger bubbles increases sharply. This results in a transition to slug flow. Recent work has shown that this flow pattern does not occur in larger diameter pipes (0.15 m and 0.2 m), where there is a direct transition from bubble to churn [70].
Experiments suggest that the bubble void fraction at which this happens is around 0.25 to 0.30 [43]. Published data agree in that the void fraction in bubbly flow rarely exceeds 0.35, whereas for void fractions less than 0.20 coalescence is rarely observed [44]. A review of models for this transition can be found in [13].

2.2.3.2 Slug-churn transition

It is difficult to accurately identify the slug-churn transition because there is confusion on the description of the churn flow itself [15]. We characterize the churn flow pattern as the condition where oscillatory motion of the liquid is observed. In slug flow, the liquid between two Taylor bubbles moves at a constant velocity and its front as well as its tail have constant speed. In churn flow, the liquid slug is too short to support a stable liquid bridge between two consecutive Taylor bubbles. The falling film around the bubble penetrates deeply into the liquid slug creating a highly agitated aerated mixture at which point the liquid slug is seen to disintegrate and to fall in a rather chaotic fashion [100]. The liquid reaccumulates at a lower level at the next slug where liquid continuity is restored and the slug then resumes its upward motion. Thus, an oscillatory motion of the liquid can be observed, which is a characteristic of churn flow. Nicklin and Davidson [69] suggested that churn flow occurred when the gas flow is sufficiently low to cause flooding in the film surrounding it. This mechanism of transition was also suggested by Jayanti and Hewitt [59] and Watson [110].

The mechanistic models that have been published in the identification of the occurrence of this transition are presented in [13].

2.3 CFD methodologies for two-phase flow

Several models have been designed to simulate multiphase phenomena: three levels of modeling can be identified based on the spatial and temporal resolution of the models.

2.3.1 Direct numerical simulation

The direct numerical simulation of gas-liquid flow is also called an interface resolving method [64]. The DNS techniques are based on the local instantaneous conservation equations. These methods focus on the finest level, e.g. individual bubbles, small vortices behind bubbles and bubble-bubble interactions. All the closure equations for the forces acting on a bubble can be directly computed. However, these approaches are restricted to a single bubble or to a few interacting bubbles due to very expensive computational requirements. In general, the numerical methods employed to solve interfaces are based on Finite Volume Methods (FVM) and can be classified into two categories: interface-tracking and interface-capturing (see Fig. 2.8).

Regarding the interface-tracking methods, they follow the motion of the interface with high precision [105], [104]. The conservation equations are solved for each phase and they are coupled across the interface, implementing the kinematic and dynamic...
conditions at the interface, which is defined as a sharp surface. Furthermore, each phase is solved on its own mesh, which deforms according to the interface movement. This method is limited to moderate interface deformations without topology changes, although this can be addressed through re-meshing the domain. As for the interface-capturing methods \[86\], \[92\], they solve not only the continuity and momentum equations in both phases, but also an additional equation for a scalar function, used as a phase indicator. Alternatively, marker-particles might be used to determine the interface over a fixed grid.

Moreover, different techniques can be employed in the interface-capturing methods, which can be classified into two main groups: surface and volume methods \[26\], \[54\].

Concerning surface methods, several techniques can be employed, such as Front Tracking (FT) or Level-Set (LS) \[96\]. In the FT technique, the interface is fitted using connected massless particles over the free-surface or a secondary moving grid to locate the interface in a Lagrangian way across an Eulerian mesh. Although in this algorithm the interface has a zero thickness, capturing interface deformations can be troublesome. In the case of the LS technique, the interface is defined as a zero level of a colour function to identify the interface position. Similarly as with FT, the interface is tracked accurately with a zero thickness, although mass is not always conserved.

Regarding volume methods, several techniques are also available, such as the Marker and Cell (MAC) and the Volume of Fluid (VOF) techniques \[52\], where the
latter is the most popular one. In the VOF technique, the interface is determined by using a scalar indicator which varies between one (fluid 1) and zero (fluid 2), corresponding to the volume fraction occupied in each cell. As a consequence, in the VOF method, the interface is characterized by a discontinuous change in the volume fractions and difficulties related to its advection are well documented in the literature [42].

There are several ways to implement this concept and in the framework of OpenFOAM the methodology is described in extenso by Ubbink [106], Rusche [83] and Berberovic et al. [21]. In OpenFOAM [66], the solver interFoam is developed based on the compressive volume-of-fluid method, which incorporates an interfacial compression flux term to mitigate the effects of numerical smearing of the interface [36], [85], [111]. The interface location, its normal and curvature are known only implicitly from the underlying indicator function.

The main features of VOF are the conservation of mass and the ability to include phenomena such as interface breakup and coalescence. It should also be noted that this technique is dependent on the grid resolution, which might be a source of errors in the advection and reconstruction algorithms. This drawback induces the so-called artificial or numerical coalescence of the disperse phase, preventing the real definition of the interface when the separation between two interfaces is smaller than the size of the computational cell.

Furthermore, the calculation of the surface tension force constitutes a key issue of these methods, since its effects at the interface play a decisive role in multiphase flows [18], [11]. In most cases, the surface tension term is accounted for in the momentum equation by using the Continuum Surface Force (CSF) model developed by Brackbill et al. [22].

Nevertheless, this model may introduce parasitic or spurious currents [45], [41]. The main sources of spurious currents have been identified as inaccurate interface curvature and lack of a discrete force balance [36]. Several solvers have been developed to reduce these parasitic currents [73], [78] although the problem has not been completely solved so far [57].

### 2.3.2 Eulerian-Lagrangian method

In the Eulerian-Lagrangian approach, the continuous phase is treated in the Eulerian framework and the averaged equations are solved, whereas the motion of individual discrete particle is simulated by solving Newton’s equation of motion. The trajectories of the particles are computed in the control volume. In contrast to the DNS methods, the Eulerian-Lagrangian approach requires closure relations to account for the interphase forces. The closure models can be obtained from empirical relations or from more sophisticated simulations with a fine resolution.

Most of the gas-solid Eulerian-Lagrangian methods are developed by coupling CFD with the discrete element method (DEM): also OpenFOAM is capable of doing the Eulerian-Lagrangian simulations for gas-solid flows [117].
2.3. CFD methodologies for two-phase flow

2.3.3 Eulerian-Eulerian method

In the Eulerian-Eulerian approach, each phase is treated as a continuous medium interpenetrating the other phase, and is represented by the macroscopic conservation equations, which are valid throughout the entire flow domain.

This method is commonly known as the two-fluid model or multifluid model \cite{55}. This approach requires less computational effort than the Eulerian-Lagrangian approach. However, the discrete character of the dispersed phase is lost due to the averaging procedure. The model employs averaged mass and momentum conservation equations to describe the time-dependent motion of both phases and requires additional models for the inter-phase momentum transfer and for the turbulence, due to the averaging process \cite{83}. In \cite{112} the derivation, using conditional averaging (\cite{37}), of the two-phase equations is outlined followed by the manipulation necessary to cast the equations in a form suitable for numerical solution.

The two-fluid model is widely employed to simulate gas-liquid flows. Most of the gas-liquid two-fluid simulations were carried out using a single mean bubble size \cite{116}. This assumption is usually reasonable in the homogeneous flow regime. To account for the bubble size distribution, many attempts have been made to couple computational fluid dynamics with a population balance model (CFD-PBM) to simulate gas-liquid flows \cite{28}, see \cite{19}, \cite{90}.

The applicability of each model and the quality of the results strongly relies on the nature of the problem. The VOF model is used in problems where surface capturing is crucial, such as the importance of surface tension and adhesion phenomena (jet break-up, drop formation) or where the free surface position prediction is essential (free-surface problems in hydraulics, liquids separation). In all of these cases the interfaces are covered by an appropriate mesh size, Fig. 2.9. In the case of two-fluid models the interest is in the capability of predicting the behaviour of flows with small-scale interfaces when is not possible or desirable to have a more complete modeling. This kind of interfaces is often found in sedimentation tanks, cyclone separators, annular flow in refineries and fine bubbles flow in heat exchangers.
The DNS is the only method that is capable to resolve general multiphase problems \cite{86}; however, the required huge computational resources make DNS out of reach for most applications. On the other hand, due to their lack of generality, the other models work only for particular cases that sufficiently satisfy the underlying model assumptions.

This master thesis deals with the validation of a two-phase CFD code in OpenFOAM based on the combination of an Eulerian two-fluid approach and sharp interface capturing using the Volume of Fluid approach. Details of the model are presented in Chapter 3.
Chapter 3

Coupling Euler-Euler two-fluid with VOF

3.1 Hybrid approach

In a dispersed flow, where chunks of the particular fluid are smaller than the grid cells, the interface tracking algorithm fails. Two-phase flows of practical importance, including the flow regimes described previously in 2.2, can be too dispersed to be resolved with the interface tracking algorithm. The two-fluid models are suitable for two-phase problems where the length scale of the interface shape is smaller than the grid size. However the basic assumption of those models is that each fluid behaves as a continuum that occupies the whole domain and the information that is lost due to the averaging is replaced by closure relations for the interfacial transfer of mass, momentum and energy. This approach is suitable for simulation of the dispersed flow. The equations of the two-fluid model are less accurate than the VOF model due to the empirical closures required in the averaged equations. However, in the case of sufficiently dispersed flow, the results of the two-fluid model are still much closer to the laboratory and field data than the results of the VOF method, which do not have any physical meaning when the grid becomes too coarse. Moreover VOF fails to take into account the slippage velocity between the two phases due to its shared momentum concept. Therefore at higher flow rates, where a large slippage exists, the functionality of this model is questionable.

In complex multiphase flows in which both dispersed flow and segregated flow regions are present one would like to couple VOF and the two-fluid model. From the physical point of view the coupling is not problematic since the VOF model uses an indicator function for tracking the interface between the phases, which has the same meaning as a volume fraction variable in the two-fluid model.

One of the first studies in this field is due to Cerne et. al. where a coupling between the VOF and two-fluid models is used to solve the Rayleigh-Taylor instability. They employed a switching routine based on the gradient of the volume fraction across neighbouring cells to flag cells as either VOF or two-fluid and solved the appropriate number of equations in each cell. Thus, there is a threshold value over which the interface is treated as having long scale and captured by VOF, and the opposite case with the two-fluid model. The challenge of such a
coupled model was to combine two mathematical models with different numbers of equations across the same domain.

Strubelj and Tiselj [95] give a good overview of methods that have been employed for this coupling along with details regarding difficulties in coupling the phase momentum equations at the sharp interface, where the phase velocities should be equal. This issue has particular importance in the treatment of the velocities, since the VOF model has only one velocity field whereas the two-fluid model has one velocity per phase. The transition from two velocities to one is managed by the definition of the velocity for the centre-of-volume, in the opposite case the same velocity is assigned to each phase, losing the interface friction.

Wardle and Weller [109] developed a hybrid multiphase method in OpenFOAM based on the combination of an Eulerian multifluid solution framework and sharp interface capturing. This solver, named multiphaseEulerFoam, has been included in the release of version 2.1 of OpenFOAM.

3.2 multiphaseEulerFoam

In multiphaseEulerFoam a numerical interface sharpening is implemented within the Eulerian framework. The advantage is that the governing equations to be solved are the same in the whole domain; on the other hand, a numerical sharpening on top of an Eulerian solver may not be as effective as a coupled model with interface capturing capabilities [103].

3.2.1 Multifluid momentum equations

In order to derive the conservation equations of the model, the individual phases should be distinguished. This is achieved by conditioning the local equations so that contributions to the averaged conservation equation of one phase come only from regions which contain that particular phase [51]. The idea of conditioning is based on the work of Dopazo [37]. For the conditional averaging (sometimes called phase-weighted averaging), the governing equations are multiplied by a phase indicator function before standard averaging techniques are applied.

The phase indicator function $I_k(x, t)$ is defined as:

$$I_k(x, t) = \begin{cases} 1 & \text{if point } (x, t) \text{ is in phase } k \\ 0 & \text{otherwise} \end{cases}$$

(3.1)

The phase volume fraction is calculated as the probability of point $(x, t)$ being in phase $k$:

$$\alpha_k = \overline{I_k(x, t)}$$

(3.2)

where the overbar represents the ensemble average. The ensemble average is more fundamental than time and volume averaging and it does not have the time
and space restrictions. As already pointed out the averaging process introduces new variables and terms and therefore closure laws are required.

In Rusche [83] and Weller [112] this ensemble averaging method is described in details, and this is not reproduced in the present report.

Assuming that there is no mass transfer across the phases, the conditionally averaged phase continuity and momentum equations for incompressible, isothermal flow are given by a set of mass and momentum equations for each of the phases $k$:

\[
\frac{\partial \alpha_k}{\partial t} + u_k \cdot \nabla \alpha_k = 0 \tag{3.3}
\]

\[
\frac{\partial (\rho_k \alpha_k u_k)}{\partial t} + (\rho_k \alpha_k u_k \cdot \nabla) u_k = -\alpha_k \nabla \rho + \nabla \cdot (\mu \alpha_k \nabla u_k) + \rho \alpha_k g + F_{D,k} + F_{s,k} \tag{3.4}
\]

where $\rho_k$, $\alpha_k$, $u_k$ are the density, phase fraction and velocity, respectively, for phase $k$ and $g$ is the gravity. The two interfacial forces are the drag force $F_{D,k}$ and the surface tension force $F_{s,k}$. Generally, in the gas-liquid flows, the interfacial forces (interphase momentum transfer) are divided into two categories: drag force and non-drag forces. The non-drag forces are the lift force, the virtual mass force, the turbulent dispersion force and the wall lubrication force [64]. In this work lift, virtual mass force, wall lubrication, and turbulent dispersion are neglected, and only the drag force, that is the dominant interfacial force, is included [74].

### 3.2.1.1 Interface capturing

The solver has a flexible algorithm that is able to give a sharp interface between the phases. The interface sharpening method of Weller [111] is employed, wherein an additional term is added to Eq. (3.3) in the following way:

\[
\frac{\partial \alpha_k}{\partial t} + u_k \cdot \nabla \alpha_k + \nabla \cdot (u_c \alpha_k (1 - \alpha_k)) = 0 \tag{3.5}
\]

The interface compression method for interface capturing is not as accurate as other interface reconstruction methods; however, it is much easier to implement and it is mass conserving [42]. The interface compression scheme of Weller adds an additional term to the LHS of the volume fraction transport equation. This additional convective term is referred to as the compression term keeping in mind its role to compress the free surface towards a sharper one (it should be noted that the wording compression represents just a denotation and does not relate to compressible flow). The value for the artificial interface compression velocity, $u_c$, is given by:

\[
u_c = C_\alpha |u| \frac{\nabla \alpha}{|\nabla \alpha|} \tag{3.6}
\]

$u_c$ is applied in the direction normal to the interface to compress the volume fraction field and to maintain a sharp interface. The term $\alpha_k (1 - \alpha_k)$ ensures that it is only active in the interface region [109].

Since the derivation of Eq. (3.5), through Eq. (3.6), relies on the weighted average velocity $u$, a strong coupling between the classical VOF and the two-fluid model is achieved.
The term $\frac{\nabla \alpha}{|\nabla \alpha|}$ denotes the interface unit normal vector for the direction of the applied compression velocity. The magnitude of the velocity $|u|$ is used since dispersion of the interface can occur, in the worst case, as fast as the magnitude of the local velocity $[109]$.

The coefficient $C_\alpha$ controls the interfacial compression which can be switched on ($C_\alpha = 1$) or off ($C_\alpha = 0$). With $C_\alpha$ set to 0 for a given phase pair, there is no imposed interface compression which will result in phase dispersion according to the two-fluid model. In contrast to this when it is set to 1, sharp interface capturing is applied and VOF-style phase fraction capturing occurs. The implementation of the solver is configured such that the interface compression coefficient $C_\alpha$ is defined and applied independently for all phase pairs.

It is necessary to devise a method to calculate the coefficient $C_\alpha$ such that interface capturing is used only in regions where meshing is sufficient to resolve the interface. However, in the release version of multiphaseEulerFoam, $C_\alpha$ is simply defined as a scalar that remains a constant value in both space and time. In this work $C_\alpha$ is equal to 1. Even if the coefficient $C_\alpha$ is always set to 1, we should keep in mind that the momentum equations for each phase (after conditional averaging) include the interfacial forces (see Eq.(3.4)) and therefore the model does not become a simple VOF model. This thesis aims to provide a preliminary validation of the solver. Therefore further details on the implementation of $C_\alpha$ as a spatially and temporally varying volumetric field variable, are not considered. Interesting studies in this direction are $[109]$, $[84]$ and $[89]$.

### 3.2.1.2 Surface tension

The surface tension at the gas-liquid interface generates an additional pressure gradient (or jump), which is evaluated using the continuum surface force (CSF) model (see $[22]$). This model interprets surface tension as a continuous, three dimensional effect across an interface, rather than as a boundary value condition on the interface:

$$ F_{s,k} = \sigma \kappa \nabla \alpha $$  \hspace{1cm} (3.7)

where $\sigma$ is the fluid surface tension coefficient and $\kappa$ is the local surface curvature determined from:

$$ \kappa = -\nabla \cdot \left( \frac{\nabla \alpha}{|\nabla \alpha|} \right) $$  \hspace{1cm} (3.8)

The accurate calculation of the phase fraction distribution is crucial for a proper evaluation of the surface curvature, which is required for the determination of the surface tension force and the corresponding pressure gradient acrosss the free surface. The interface region between two phases is typically smeared out over a few grid cells and is therefore highly sensitive to the grid resolution.

Version 2.2.x of OpenFOAM is used in this work. In this version, and also in the following versions, surface tension was omitted from the reconstruction of the velocity field, though it is included in the construction of the pressure itself. However it is straightforward to fix the velocity reconstruction, see E and E.1.
3.2. Interphase drag models

The interface drag represents the resistance opposed to the bubble motion in the fluid (or, more generally, the resistance due to the relative motion between two phases). The drag force clearly depends on the bubbles size (i.e. a larger bubble experiences a larger drag force) and on the relative velocity between the two phases. The drag term $F_{D,k}$ is given by [109]:

$$F_{D,k} = \alpha_c \alpha_d K (u_d - u_c)$$  \hspace{1cm} (3.9)

where the subscripts $c$ and $d$ denote the continuous and dispersed phase values and $K$ is given by:

$$K = \frac{3}{4} \rho_c C_D \frac{|u_d - u_c|}{d_d}$$  \hspace{1cm} (3.10)

The drag calculation implemented in the solver is generic, such that the model must simply return the value of $K$. The drag coefficient $C_D$ is usually deduced from experiments and many models were developed in order to fit different experimental data sets. The drag coefficient is related to many factors: the bubble shape, orientation with respect to the flow, flow parameters such as the bubble Reynolds number, the Eotvos number, turbulent level, and so on. An interesting and comprehensive literature study on the topic can be found in [83]. In the solver many different drag coefficient models are implemented [66].

The model of Schiller and Naumann [87] is used for the simulations in this thesis. According to this model the correlation for the drag coefficient is:

$$C_D = \begin{cases} \frac{24(1+0.15\text{Re}^{0.683})}{\text{Re}} & \text{Re} \leq 1000 \\ 0.44 & \text{Re} > 1000 \end{cases}$$  \hspace{1cm} (3.11)

$$Re = \frac{d_d |u_d - u_c|}{\nu_c}$$  \hspace{1cm} (3.12)

where $\nu_c$ is the kinematic viscosity of the continuous phase and $d_d$ is the diameter of the disperse phase.

In multiphaseEulerFoam the calculation of the drag coefficient can be done by specifying a dispersed phase or by an independent calculation with each phase as the dispersed phase. For the latter case the overall drag coefficient, applied to the momentum equations, is taken as the volume fraction weighted average of the two values. This is the blended scheme in in the code listing of Table 3.1. In this way both drag forces are calculated and the solver automatically weights the drag forces based on the phase fraction. Different models for each dispersed phase can be chosen. Therefore the parameters for the drag model need to be specified pair-wise.

The phases are read in the transportProperties dictionary from left to right, see line 3 in the code listing of Table 3.1. In this example the first phase, air, is the dispersed phase and its diameter is used for the calculation of Eq. (3.12).

Table 3.1: Drag specification in transportProperties dictionary

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>drag</td>
</tr>
<tr>
<td>2</td>
<td>(</td>
</tr>
<tr>
<td>3</td>
<td>(air siliconeOil)</td>
</tr>
</tbody>
</table>
In the listing given in Table 3.1 there are two extra parameters in the setup of transportProperties: residualPhaseFraction and residualSlip. They are not included in the calculation of the actual phase fractions or velocities, but only in the calculation of the drag force itself to help the coupled formulation stability. In the limit of a sharp interface, the velocities on either side of the interface must be equal to meet the so-called no-slip interface condition. This is an inherent feature of traditional VOF simulations as all phases share a single momentum equation and thus the phase velocities are the same everywhere. Imposing a sharp interface through the addition of interface compression on top of a two-fluid formulation requires that an additional artificial drag is used to equalize the velocities at the interface. That is why constants for a small residual drag and residual phase fraction are added for each phase pair \[109\].

### 3.2.1.4 Diameter models

The multiphaseEulerFoam module contains, since its introduction in version 2.1.0, two diameter models: constant and isothermal. Other models for variable droplet size, e.g. based on a reduced population balance method \[89\], are compatible with this flexible framework.

The isothermal model is used in the work reported here, assuming the change of state to be isothermal. Gas bubbles change their diameter as the ambient pressure changes. Generally, the ideal gas law governs the state of gas:

\[
pV = nRT
\]

under the assumption of an isothermal state:

\[
pV = \text{const}
\]

Next we introduce the bubble volume:

\[
V = \frac{d^3 \pi}{6}
\]
Thus we obtain the relation:

\[ \frac{p_1 d_1^3 \pi}{6} = \frac{p_2 d_2^3 \pi}{6} \]  

(3.16)

This leads to the isothermal diameter model:

\[ d_2 = \sqrt[3]{\frac{p_1}{d_1 p_2}} \]  

(3.17)

3.2.2 Time step

To ensure convergence for a time-marching solver a restriction on the Courant number will influence the maximum time step. multiphaseEulerFoam uses an adjustable time step which is based on the maximum Courant number in the domain. The definition of the Courant number for n degrees of freedom is given in Eq. (3.18).

\[ C_o = \Delta t \sum_{i=1}^{n} \frac{u_i}{\Delta x_i} \]  

(3.18)

The time step is directly proportional to the mesh spacing, that is, if the mesh spacing is cut in half, the timestep must essentially be decreased by a factor 2. Bounded discretization schemes for divergence terms and time step control are both used to improve the numerical stability. It is generally recommended to keep the maximum local Courant number much below unity. It is also beneficial to solve the phase fraction equation in several subcycles within a single time step. The time step to be used in a single time subcycle, \( \Delta t_{sc} \), is set by dividing the global time step, \( \Delta t \), by the preset number of subcycles (\( n_{\text{AlphaSubCycles}}, n_{sc} \)).

\[ \Delta t_{sc} = \frac{\Delta t}{n_{sc}} \]  

(3.19)

Besides the Courant number several other physical parameters will also have an influence on the stability and accuracy of the solver [20]. The surface tension can impose a restriction on the time step, related to:

\[ \Delta t \leq \max(10\tau_{\mu}, 0.1\tau_{p}) \]  

with \( \tau_{\mu} = \frac{\mu \Delta x}{\sigma} \) and \( \tau_{p} = \sqrt{\frac{\rho \Delta x^2}{\sigma}} \)  

(3.20)

3.2.3 Turbulence

In the release version 2.2.x of OpenFOAM, turbulence can be covered by using Large Eddy Simulation (LES) with the Smagorinsky sub-grid model for the mixture or with RANS (see B). These models have been originally developed for turbulence modeling in single-phase flows and thus solely accounts for turbulence within the continuous phase or within the effective fluid (i.e. the mixture). However, for the Euler-Euler formulation it is more rigorous to have a turbulence model per phase and inter-phase turbulence exchange terms [83], [32]. This is more general but also more complex. In this study turbulence is accounted through RANS models for the mixture, [74], [4], [49], [34].
3.2.4 Solution procedure

`multiphaseEulerFoam` handles the requirement to describe both the properties of the individual phases as well as the interaction terms between the phases by two distinct libraries `phaseModel` and `multiphaseSystem`. Fig. 3.1 shows the main parts of `multiphaseEulerFoam`.

The procedure starts with updating the timestep according to the Courant number limit and then solves the coupled set of volume fraction equations with interface sharpening for selected phase pairs (3.5).

The drag coefficients are computed and an equation for the phase velocities is constructed and solved for preliminary values.

The Pressure Implicit with Splitting of Operators (`PISO`) algorithm is used to solve the pressure-velocity coupling.

For the sake of generality in Fig. 3.1, the pressure-velocity coupling is handled by a `PIMPLE` loop rather than by `PISO`. The `PIMPLE` algorithm is a combination of `PISO` and `SIMPLE`. However, if in the `PIMPLE` control dictionary the number of `nOuterCorrectors` is set to 1 (default settings), the `PIMPLE` solver will operate in `PISO` mode.

The implementation of the `PISO` algorithm follows [56] and a description is given by Jasak in [58]. In a `PISO` loop the momentum equation is solved first (momentum predictor). Then the pressure equation can be formulated giving the first estimate of the new pressure field (pressure solution) and the velocity field is corrected as a consequence of the new pressure distribution. The correction is done in an explicit manner. In `PISO` there is no loop over the whole set of equations.

`PIMPLE` performs sub-iterations (outer corrector steps) to ensure convergence and to be able to use an acceptable time step. However the time step is already limited to small values by the nature of the problem (interface capturing) and the advantage of `PIMPLE` regarding using a larger time step can not be considered for this work.

`PISO` requires that the solution must converge at every time step. We assume that one solution step plus pressure corrections are enough to obtain convergence. Decreasing residuals indicate that we are converging to the right solution and they are checked at the end of every simulations.

3.2.4.1 Phase fraction

In order to ensure phase conservation for the coupled phase fractions with added interface sharpening, limiters on the phase fraction, as well as on the sum of the phase fractions, are incorporated prior to the explicit solution of the phase fraction equation system [109]. These additional limiters have been incorporated in a new multiphase implementation of the Multidimensional Universal Limiter with Explicit Solution (MULES).

The MULES algorithm handles the boundedness property by first limiting the flux transport and then solves for the phase fraction. Limiting the flux transport is important since large transport of fluxes from a cell may drive the volume fraction in a particular cell below zero during one time step. The MULES function is given in the listing of Table 3.2:
3.2. multiphaseEulerFoam

Figure 3.1: Flow chart of multiphaseEulerFoam.
Table 3.2: MULES constructor in multiphaseSystem.C

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MULES::explicitSolve</td>
</tr>
<tr>
<td>2</td>
<td>(</td>
</tr>
<tr>
<td>3</td>
<td>geometricOneField(),</td>
</tr>
<tr>
<td>4</td>
<td>phase1,</td>
</tr>
<tr>
<td>5</td>
<td>phiAlpha,</td>
</tr>
<tr>
<td>6</td>
<td>zeroField(),</td>
</tr>
<tr>
<td>7</td>
<td>zeroField()</td>
</tr>
<tr>
<td>8</td>
<td>) ;</td>
</tr>
</tbody>
</table>

The first argument passed to the constructor is the density; in case of incompressible
phases, it is passed as geometricOneField() which is a unit value field. The second
argument is the variable to be solved which is the phase fraction in our case. The
limited normal convective flux is the next argument which is earlier solved explicitly
by MULES:limit. The next two terms are the explicit and implicit source terms in
the continuity equation which arise when mass transfer across the phases or reaction
source terms exist. In our case we pass both the arguments as zeroFields() since
there is no mass transfer. The MULES algorithm solves for the phase fraction with the
explicit consideration of the convective flux of the phase fraction. The considered
transport mechanism is convection only [61]. The procedure is showed in Fig. 3.2.

The solution for the phase fraction is invoked by the fluidSolve() function.
The schematic of the function operation is shown in Fig. 3.3. If the number of
correctors are larger than one, then the phase fraction at the old time is stored and
fluxes for all phases are set to zero.
Figure 3.2: Procedure to solve the phase volume fraction.
Figure 3.3: Solution procedure of the phase continuity equation.
Chapter 4

Validation

Numerical simulations were conducted on the cluster of LINUX computer servers at the Laboratory for Aero & HydroDynamics of the TU Delft. This cluster uses the Slurm scheduler to efficiently manage workloads. One node with 12 processors has been used for the simulations presented in this thesis.

Before proceeding it is worth noting that the assumptions made in 1.2 are the basis for the following analysis. The starting objective of this thesis is the comparison of the results obtained with OpenFOAM against experiments and reference solutions that have been obtained previously with Fluent and Star-CCM+ for the industry.

4.1 Case study 1

This section will present results for the first case, which consists of annular flow at the inlet. Section 4.1.1 will introduce the problem and the flow regime considered. Section 4.1.2 discusses the computational details, such as the mesh and solver settings. Results will be presented in section 4.1.3.

4.1.1 Case description

In [115] Worthen and Henkes carried out CFD simulations with ANSYS Fluent 15.0 for the splitting of two-phase, gas-liquid flow from a horizontal flowline into two vertical risers. The same flow conditions that were used in the air-water experiments at the Shell Technology Centre Amsterdam were simulated. In addition to the splitter geometry, the flow through the vertical pipe alone with fixed inlet boundary condition was simulated, as is illustrated in Fig. 4.1. The latter is the case study that is considered in this work. The outlet is located 50D (2.54 m) downstream the inlet.

The liquid holdup fraction at the inlet was set to 0.18 which is the value predicted by the Shell Flow Explorer tool (SFE version 6.0). The inlet flow rates of gas and liquid were:

\[ Q_{\text{air}} = 31.1 \text{ m}^3/\text{h} \quad \text{and} \quad Q_{\text{water}} = 1 \text{ m}^3/\text{h} \]  \hspace{1cm} (4.1)

The nominal pipe diameter was 50.8 mm which gives the superficial velocities of:
The liquid entered the domain as an annular film while the gas entered through the core and then the two phases flow upwards and discharge through the outlet at atmospheric pressure. The same conditions are imposed in the simulation with OpenFOAM. In the Fluent simulations of [115] a total flow time of 9 s was simulated while in the present study the physical time used is 14 s. This time interval allows flow stabilization. The liquid holdup and pressure drop were only calculated over the last 30D of pipe length to ensure a more fully developed flow. The same quantities as considered with Fluent are calculated from the present OpenFOAM simulations in order to compare the results.

4.1.2 Case setup

4.1.2.1 Mesh

It has been reported by Hernandez-Perez et al. [49] and confirmed in [3] and [102] that different grid structures, applied in the cylindrical pipe geometry, can give a different numerical accuracy. In [49] the O-grid is highly recommended for the simulation of two-phase flow in a pipe. This grid allows refining the mesh close to the wall and prevents a singularity at the centre of the pipe. In this grid, a Cartesian mesh is used in the centre of the pipe combined with a cylindrical one around it. It requires multiple blocks but generally has the best grid quality in terms of orthogonality and mesh density. Therefore the O-grid is the type of mesh employed. The construction of the mesh is realized with blockMesh. A parameterization of the blockMesh dictionary is done by using m4-scripting (see A). Fig. 4.2 shows a section of the computational domain. The computational mesh contains 563200 hexahedral cells. The number of cells along the axis of the pipe (positive z axis) is

\[ U_{s,\text{air}} = 4.26 \text{ m/s} \quad \text{and} \quad U_{s,\text{water}} = 0.137 \text{ m/s} \] (4.2)
4.1. Case study 1

It was not possible to carry out a grid convergence study to check the mesh dependency because of the high computational costs required, however we can consider the work of Worthen & Henkes in [115] where the mesh dependency of the solution for the same case is checked. Moreover there are experimental results available for the comparison with the CFD results. About 55 days of real time were needed to simulate 14 seconds of physical time by performing a parallel computation on 12 processors with the time step limitations discussed in 4.1.2.6.

The mesh shown in Fig. 4.2 is a rather coarse grid for an interface capturing solver, where a mesh spacing of $\sim 10$ times smaller than the smallest droplet would be required. However the scope of this work is to provide a first validation of the solver using limited computational time.

The idea of the coupled solver is to deal with different scales: the VOF model is appropriate for flows with large interfacial length scales (larger than grid size), and the Eulerian-Eulerian model for the cases with small length scales. Neither of them seems to be applicable for this problem: VOF requires a finer mesh and two-fluid is to be discarded because we want to track the interface, at least for the large scales of the annular liquid film. With this problem we can test multiphaseEulerFoam capabilities.

The quality of this full hexahedral O-grid mesh is high. The maximum skewness is equal to 0.48 and the maximum mesh non-orthogonality is equal to 38.3. Non-orthogonality introduces misalignment between the face normal vector and the computed gradients and affects the Laplacian terms. High values of skewness can
affect the convective terms. It introduces a first order interpolation error due to the distance between the intersection of the line connecting two cell centres with their common face and the centre of that face. The volume ratio plays also a role, introducing artificial, non-physical boundaries and inconsistent interpolation.

4.1.2.2 Turbulence

Turbulence is modelled using the RANS equations with the \( SST \) \( K - \omega \) model for the mixture [115], with wall functions [76], [114] (see also B.2). The initialization of the turbulence quantities follows the expressions provided in [46].

4.1.2.3 Boundary and initial conditions

Flow rates are prescribed at the inlet through the superficial velocities (see (4.1) and (4.2)) and through the liquid hold up. Fig. 4.3 shows the boundary condition for the inlet. The liquid holdup is 1 for the annular film and 0 in the gas core. The velocity at the inlet is shown in Fig. 4.3b where the air and water velocity (not the superficial ones, see Eqs. (2.3)) are prescribed to ensure that the flow rates in the simulations and in the experiments are identical.

In order to have the non-uniform volume fraction and velocity conditions at the inlet the special utility provided by OpenFOAM, called \texttt{funkySetBoundaryField}, is used. At the wall the no slip condition is applied for the velocity. At the outlet the pressure is fixed at the atmosferic value through the type \texttt{fixedMean}. It is worth mentioning that the first choice was for the \texttt{totalPressure} boundary condition. This condition reverts to a fixed value condition for flow out of the domain. However this was causing an instability when a liquid/gas mixture starts to leave the domain. Therefore a \texttt{fixedMean} condition for the pressure is used at the outlet, which extrapolates the field to the patch using the near-cell values and adjusts the distribution to match the specified mean value.
The initial condition is that the riser contains gas and that all the velocity components are equal to 0 m/s.

4.1.2.4 Fluid properties

The relevant fluid properties are:

\[ \rho_{\text{air}} = 1.25 \text{ kg/m}^3 \quad \text{and} \quad \mu_{\text{air}} = 0.0178 \text{ cP} \]  \hspace{1cm} (4.3a)

\[ \rho_{\text{water}} = 999 \text{ kg/m}^3 \quad \text{and} \quad \mu_{\text{water}} = 1.3 \text{ cP} \]  \hspace{1cm} (4.3b)

The air-water surface tension is specified as 0.0742 N/m.

4.1.2.5 Solver Settings

The subdirectory system contains all the files related to the solution procedure itself. Governing equations have to be discretized in time and space to be solved numerically and the discretization schemes adopted can be specified in fvSchemes. For the time discretization the implicit Euler scheme is used, which is a first order scheme. For the discretization of gradient terms the leastSquares scheme is used with cellMDLimited as gradient limiters. It is a second-order scheme, with least squares distance calculation using all neighbour cells. Gradient limiters will avoid over and under shoots on the gradient computations. For laplacian terms the Gauss linear limited corrected scheme is adopted. The discretization of the divergence terms is specified with the Gauss scheme with Van Leer (strictly bounded between 0 and 1, vanLeer01) and with the linear interpolation scheme. The fvSchemes dictionary is reported in D. Using the implicit first order scheme in time reduces the overall accuracy to first order. However the rather coarse mesh used is already limiting the spatial discretisation accuracy. Equivalent temporal and spatial accuracy implies that the peak error magnitudes from temporal and spatial discretisation are the same. It would be useful to know how much of the total error is caused by spatial and how much by temporal discretisation but it is outside the scope of this work, keeping in mind the objective to simulate industrial flows.

In fvSolution the linear-solver control is specified. Each discretized equation needs to have a selected linear solver and the solution tolerances need to be specified to control the accuracy of the solver. Specifying a low tolerance might lead to long calculation times. It is therefore important to consider an appropriate compromise between accuracy and the number of iteration loops needed to achieve convergence. To solve the pressure equations the Geometric-algebraic multi-grid solver (GAMG) was chosen. The principle behind GAMG is to first calculate an intermediate solution on a coarse mesh, and then map it onto a finer grid using the first solution as an initial guess. This is considered a relatively quick method. The pressure and velocity fields were coupled by the PISO algorithm, since nOuterCorrectors is set to 1. The fvSolution dictionary is reported in C.

4.1.2.6 Simulation control

The time step is adjusted to the largest possible value, while still fulfilling the Courant number criterion, that is the maximum Courant number being specified to
Figure 4.4: CFD results with OpenFOAM for the flow pattern in the vertical pipe at $t = 10 \text{ s}$. Cells having a water volume fraction larger than 0.5 are coloured blue.

0.2. The upper limit on the time step is set to $5 \times 10^{-6}$ which decreases the maximum Courant number to $\sim 0.03$ during the simulation. This is done to increase the stability of the simulation and improve the convergence.

4.1.3 Results

The CFD results described in [115] were obtained with Fluent using a VOF method. An attempt to use the VOF method is made also in OpenFOAM with similar results. The disagreement between experiment and CFD (see Table 4.1 and Fig. 4.9) is attributed to the failure of the VOF method to predict slug flow in the risers. Therefore the hybrid model is used in the following simulations.

In the experiments the gas flow was too low to obtain annular flow in the risers and the flow condition observed is churn flow. The CFD results for the flow pattern in the vertical pipe at $t = 10 \text{ s}$ are presented in Fig. 4.4. The CFD simulations predict a churn-like flow in the riser.

Fig. 4.5 displays time series of cross-sectional averaged void fraction data (see Eq. (4.4)) which contain prime information concerning liquid structures within the flow. All time series exhibit cyclic fluctuations that are in the form of sudden drops in cross-sectional averaged void fraction. The presence of the drops in the time series is due to the passage of cyclic liquid structures. The amplitude of these drops is controlled by the liquid volume of the passing structure. In other words, the more the liquid content of the passing structure, the larger the amplitude of the drops in the time series. According to Parsi et al. [75], the cyclic liquid structures can be liquid continuous (liquid slug body) or non-continues liquid structures (huge wave, disturbance wave). In order to determine the time series of the void fraction, the following procedure is followed (which is similar to the one used by Hernandez-Perez. [48] and Abdulkadir [3]): a cross-sectional plane is defined across
the measurement location and an area-weighted average value of the void fraction is calculated. The area-weighted average of the void fraction is computed by:

\[
\frac{1}{A} \int \alpha A = \frac{1}{A} \sum_{i=1}^{n} \alpha_i A_i
\]  

(4.4)

where the sum is extended to all the faces of the cells that lie on the measurement section.

Fig. 4.6 demonstrates the ability of the CFD model in OpenFOAM to reproduce a liquid structure briefly bridging the pipe cross-section as well as large interfacial waves of different sizes compressing the gas core.

The instantaneous (at \( t = 10 \) s) liquid holdup and mixture velocities are shown in Fig. 4.7 and Fig. 4.8 for different sections.

For this case the experimental results available are the pressure drop and the liquid holdup fraction. The liquid holdup was not measured but it was estimated based on the measured pressure drop across the riser, assuming that the pressure drop was due solely to gravity (i.e., frictional pressure drop was assumed negligible).

The liquid holdup and pressure drop were time-averaged after a time at which the flow was considered to have been stabilized to obtain a single value for each and compare them with [115]. In the work of Worthen & Henkes the quantities were time-averaged over the time between \( 2 \) s and \( 9 \) s. From Fig. 4.9 we can see that for the case with multiphaseEulerFoam the time to reach fully developed flow is larger than with Fluent and therefore the average is computed over a flow time from \( 5 \) s to \( 14 \) s.

The value of the pressure drop per unit length is computed by defining two cross-sectional planes: at \( 20D \) and at the outlet. An area-weighted average value of the pressure is calculated for the two sections (\( p_{20D} \) and \( p_{Outlet} \) respectively). Finally the pressure drop is given by:

\[
\Delta p_l = \frac{p_{20D} - p_{Outlet}}{l}
\]  

(4.5)

where \( l \) is the distance between the two sections. The total liquid holdup fraction is simply defined as the volume average of the phase fraction of water over the entire pipe. Table 4.1 gives a summary of the predicted and measured pressure drop per unit length and for the liquid holdup fraction for upward flow through a 50.8 mm diameter vertical pipe. The results provided by this work with multiphaseEulerFoam are indicated as OpenFOAM.

The prediction by OpenFOAM using multiphaseEulerFoam agrees better than any other method with the measurements, indicating that multiphaseEulerFoam does properly capture the multiphase flow pattern. The large differences between the liquid holdup fraction and pressure drop as found in the CFD simulation with multiphaseEulerFoam and the CFD simulation with VOF, show the capability of this hybrid model to overcome the limitations of the VOF model that was used in Fluent and OpenFOAM.

Fluent and OpenFOAM with VOF overpredicts the dispersion between the phases, which gives a too low liquid holdup fraction, which in turn also gives a too low hydrostatic head and thus a too low pressure drop.
Figure 4.5: Time series for the void fraction in the CFD simulations with OpenFOAM.
4.1. Case study 1

(a) Liquid holdup

(b) Streamlines for the gas phase on the yz-plane

Figure 4.6: CFD results for the last 30D of the vertical pipe at $t = 10$ s.
Figure 4.7: Liquid holdup at different cross sections at $t = 10$ s.
Figure 4.8: Mixture velocity at different cross sections at $t = 10$ s.
4.2 Case study 2

This section will present results for the second case, being slug flow in a vertical pipe. Section 4.2.1 will introduce the problem and the flow regime considered. Section 4.2.2 discusses the computational details, such as the mesh and solver settings. Results will be presented in section 4.2.3.

4.2.1 Case description

This second test case is based on the work of Abdulkadir et al. [3], [5], [6], [8], which compares the results obtained from experiments and CFD studies of slug flow in a vertical riser. In [3] a series of two experimental investigations were carried out for a 6 m vertical pipe with a 0.067 m internal diameter using an air-silicone
oil mixture. The details of this experimental facility can be found in Azzopardi et al. [6], [9]. The experimental test section consists of a transparent acrylic pipe and the flow patterns were recorded using electrical capacitance tomography (ECT) and wire mesh sensors (WMS). A detailed description of the ECT and WMS technology can be found in [6], [10], [91] and [7]. A ring of two measurement electrodes (also known as twin-plane sensors) were placed around the circumference of the riser at a given height above the injection portals at the bottom of the riser section. The use of the twin-plane sensors enabled the determination of the rise velocity of any observed Taylor bubbles and liquid slugs. The twin-plane ECT sensors were placed at a distance of 4.4 m (ECT-Plane 1) and 4.489 m (ECT-Plane 2) from the base of the riser. The capacitance WMS (WMS-Plane 3) was placed at 4.92 m away from the mixing section, at the base of the riser. In addition to the physical experiments, CFD simulations were carried out by Abdulkadir et al. in [3] using Star-CD and Star-CCM+ with a VOF method. In this case the VOF method in Star-CCM+ seems to provide good results. However in OpenFOAM the hybrid model is again used since a preliminary study with VOF failed in simulating the slug flow.

4.2.2 Case setup

The Solver settings and Simulation controls in OpenFOAM will not be reported again, since they are identical to the previous case (4.1.2.5 and 4.1.2.6), except for the maximum time step which is set to $5 \times 10^{-5}$.

4.2.2.1 Mesh

Fig. 4.10 shows the geometry for the computational flow domain. A mesh sensitivity analysis has been carried out in [5] in order to identify the minimum mesh density that is required to ensure that the solution is almost independent of the mesh resolution. In this work four different sets of grid are developed for a computational domain of 1 m length. This length is sufficient to carry out a test on the performance of the mesh with quite reasonably computational effort [5]. A comprehensive mesh dependency study on the 6 m long vertical pipe, with successively refined grids, would require huge computer times which is out of the scope of this thesis. The meshes that were investigated are shown in Fig. 4.11 and Table 4.2 lists the details of the grids employed.

The simulation results of the Taylor bubble by mean of the aforementioned four different sets of grid are shown in Fig. 4.12. From Fig. 4.12, the Taylor bubble shape can be clearly observed in all four meshes. However, the interface of the higher

<table>
<thead>
<tr>
<th>Grid name</th>
<th>Cross-section</th>
<th>z-direction</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>352</td>
<td>180</td>
<td>63360</td>
</tr>
<tr>
<td>b</td>
<td>513</td>
<td>207</td>
<td>106191</td>
</tr>
<tr>
<td>c</td>
<td>737</td>
<td>250</td>
<td>184250</td>
</tr>
<tr>
<td>d</td>
<td>1148</td>
<td>300</td>
<td>344400</td>
</tr>
</tbody>
</table>
Figure 4.10: 3D geometry of the computational flow domain showing the location of the experimental measurement.
Figure 4.11: Cross-sectional view of different computational grid used for mesh independent study.
Figure 4.12: Simulation with OpenFOAM for the Taylor bubble with different grid number.

Figure 4.13: Void fraction (see Eq. (4.4)) at the measurement section (0.5 m).
4.2. Case study 2

Figure 4.14: Computational mesh used for simulations by Abdulkadir et al. in [5].

Figure 4.15: Cross-Section grid of the computational domain in OpenFOAM.

mesh resolution is sharper than in the other cases. In order to evaluate whether or not the employed mesh resolution is adequate to obtain accurate solutions, the time for the Taylor bubble to reach a section located at a distance of 0.5 m downstream the inlet is compared in Fig. 4.13 for the four meshes [3]. From Fig. 4.12 and Fig. 4.13 it can be concluded that the mesh $c$ is adequate. The change in the results produced is very small when the number of cells is increased to the finer mesh $d$.

Fig. 4.14 shows the mesh used in [3]. The mesh used in this study is given by Fig. 4.15 and the same considerations of 4.1.2.1 hold also for this case.
In [48], [75], [107] and [5] CFD simulations with similar coarse meshes are described. The number of hexahedral cells is 1105500, the maximum orthogonality is 32.7 and the maximum skewness is 0.52. The experimental data were obtained over an interval of 60 s while the CFD covered 17 s both with OpenFOAM in the present work and with Star-CCM+ in [5]. About 36 days of real time were needed to simulate 17 seconds of physical time by performing a parallel computation on 12 processors with the upper limit on the time step set to 5e−5.

4.2.2.2 Turbulence

It is important to consider the turbulence in the numerical simulation as the bubbles rising through the liquid create a developing film around themselves and a wake behind them [98], [80]. The k-ε model with wall function is used as suggested by the multiphase flow study of Ramos-Banderas et al. [79] and as was also used in [4] and [30]. The flow inlet values for the turbulent kinetic energy k, and dissipation rate ε, are estimated using the equations proposed by Launder and Spalding [63].

4.2.2.3 Boundary and initial conditions

At the flow inlet, the mixture superficial velocity, \( U_m \), given by (2.2) is specified. The volume fraction of each phase is specified at the inlet as a homogeneous mixture:

\[
\alpha_{\text{liquid}} = \frac{U_{s,\text{liquid}}}{U_m} \quad \alpha_{\text{gas}} = \frac{U_{s,\text{gas}}}{U_m}
\]  
(4.6)

The superficial velocities are:

\[ U_{s,\text{air}} = 0.344 \text{ m/s} \text{ and } U_{s,\text{silicone oil}} = 0.05 \text{ m/s} \]  
(4.7)

At the outlet at the top of the riser, a fixed value for the pressure is specified. As in 4.1.2.3 the type fixedMean is used. At \( t = 0 \) the riser is completely filled with liquid at zero velocity.

4.2.2.4 Fluid properties

The relevant fluid properties are:

\[
\rho_{\text{air}} = 1.18 \text{ kg/m}^3 \quad \text{and} \quad \mu_{\text{air}} = 0.000018 \text{ kg/ms} \]  
(4.8a)
\[
\rho_{\text{silicone oil}} = 900 \text{ kg/m}^3 \quad \text{and} \quad \mu_{\text{silicone oil}} = 0.0053 \text{ kg/ms} \]  
(4.8b)

The air-silicone oil surface tension is specified as 0.02 N/m. The dimensionless numbers \( E_o, N_f, \) and \( M_o \) are presented in Table 4.3. The regime of this case corresponds to \( E_o > 100 \) and \( N_f > 300 \). According to Wallis et al. [44] when \( E_o > 100 \) the surface tension plays little role in determining the slug ascent velocity. In a later critical review of the literature Fabre et al [39] concluded that the viscosity acts essentially to develop the liquid velocity profile far ahead of the bubbles, but has no influence near the front where the inertia dominates. This condition is satisfied provided \( N_f > 300 \) and the rise velocity of the bubble is determined solely by liquid inertia [44].
Table 4.3: Dimensionless numbers.

<table>
<thead>
<tr>
<th>Dimensionless numbers</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eotvos number</td>
<td>$E_o = 1981.67$</td>
</tr>
<tr>
<td>Dimensionless inverse viscosity</td>
<td>$N_i = 9311.72$</td>
</tr>
<tr>
<td>Morton’s number</td>
<td>$M_o = 1.035 \times 10^{-6}$</td>
</tr>
</tbody>
</table>

4.2.3 Results

In Fig. 4.16 the vector plot of the velocity field around the leading Taylor bubble rising in the stagnant silicone oil is shown.

As shown in Fig. 4.16 the bubble has a round nose and fills almost the cross sectional area of the pipe. The liquid ahead of the bubble moves around the bubble as a thin liquid film moving downwards in the annular space between the pipe wall and the bubble surface. At the rear of that bubble, the liquid film plunges into the liquid plug behind the bubble and produces a highly agitated mixing zone in the bubble wake. This recirculation zone contains small bubbles shed from the bubble tail due to the turbulent jet of the liquid film. A similar observation was reported in [40].
Fig. 4.17 depicts the fully developed slug flow. Taylor bubbles can clearly be viewed followed by liquid slug bodies with dispersed small bubbles. The oscillatory motion of the liquid phase can be observed in Fig. 4.18 where the downward flow of silicone oil around the air bubbles is shown. The flow in the liquid slug body shown in Fig. 4.17 can be divided into two main parts: immediately below the rear of the bubble, where there is the formation of a recirculation and mixing region, also called wake region and the main body of the liquid slug body where the flow is gradually recovering its original and undisturbed state. For a fully developed continuous slug flow, the length of the liquid slug bodies between any pair of consecutive bubbles remains constant and long enough, so that all the bubbles are not interacting with each other, and are rising at the same translational velocity. The intermittence evident in the developing slug flow pattern indicate the extremely complicated nature of the hydrodynamics involved.

The void fraction is a fundamental quantity in the description and analysis of
two phase flows; in the case of slug flows the void fraction at any point varies with time. Therefore monitoring the time trace of cross-sectional average void fraction is desirable.

A fully developed flow is defined as one where the flow pattern does not change with the downstream distance, and it also does not change over time. Time series of void fraction and probability density function (PDF) of void fraction obtained from the CFD simulation are used to assess the change in the flow characteristics. The flow development over time is shown in Table 4.4 for different test section.

For large numbers of discrete samples, the PDF can be estimated by a normalized histogram calculation. Here the number of occurrences in the histogram is normalized by the total number of samples, thus representing a probability. For the histogram calculation, 200 bins in the phase fraction interval 0 to 1 were used. Costigan and Whalley proposed that twin peaked probability density function of recorded void fractions represent slug flow. The low void fraction peak corresponds to the liquid slug body while the high void fraction peak corresponds to the Taylor bubble.

The PDF of the time series of void fraction in Table 4.4 shows that the results obtained from 1.0 m are initially affected by entrance effects. At a distance of 4.0 m from the inlet the PDF of void fraction has taken the shape of slug flow showing a double peak.

It can be concluded that at 4.0 m the flow is fully developed. The experimental measuring instruments used by Abdulkadir et al. were located at 4.4 m (ECT-Plane 1), 4.489 m (ECT-Plane 2) and 4.92 m (WMS-Plane 3).

Fig. 4.19 shows a typical plot of a large trailing Taylor bubble (start-up) and leading train of Taylor bubbles (stationary) from the simulation with OpenFOAM. A qualitative comparison between CFD simulations with Star-CCM+ and experiment is given in Fig. 4.20 as reported by Abdulkadir et al. in.

The comparison between CFD simulations with OpenFOAM and experiments is shown in Fig. 4.21. The experimental data were kindly provided by Professor

Figure 4.18: Simulation with OpenFOAM for the fully developed slug flow: instantaneous velocity.
Barry Azzopardi and Professor Mukhtar Abdulkadir.

From the time traces in Fig 4.21 we can clearly distinguish the slug flow regime with a reasonably good agreement between the CFD and experimental results. The slug flow data have alternating periods of high and low void fraction. High void fraction marks the gas bubble passage, and low void fraction marks the passage of the liquid slug body with same entrained dispersed gas bubbles. However, the void fraction in the liquid slug body is seen to have lower values in the CFD simulations than in the experiments.

The PDFs corresponding to the void fraction time series in Fig. 4.21 and in Fig. 4.20 (for the CFD simulation with Star-CCM+) are shown in Fig. 4.22.
4.2. Case study 2

Figure 4.21: Time series for the void fraction in different planes; comparison between experiments and CFD simulations with OpenFOAM.

Again, we can notice the higher values of the void fraction in the liquid slug
Figure 4.22: PDF of cross-sectional average void fraction for the case of slug flow obtained from the experiments, from the CFD simulations with Star-CCM+ (in Abdulkadir et al. [5]) and with OpenFOAM.
body in the CFD results with both Star-CCM+ and OpenFOAM, as compared to the experiments. There is a good agreement between the CFD simulations and the experiments in predicting the same flow pattern, being slug flow: there is a twin-peaked PDF, with one peak being characteristic for the liquid slug body and the other peak being characteristic for the Taylor bubble.

By cross-correlating the void fraction signals from ECT-Plane 1 and ECT-Plane 2 (in Fig. 4.23), the transit time between the two sections can be measured; this together with the distance between the measurement sections enable us to calculate the velocity of the slug unit for the fully developed flow. Fig. 4.24 shows that the time delay between the planes in the CFD predictions with OpenFOAM and the experiments is 0.08 s, while in the CFD simulations with Star-CCM+ is equal to 0.075 s. Dividing the distance of 0.089 m between the Plane 1 and the Plane 2 by the delay time of 0.075 s gives a slug velocity of 1.19 m/s. The slug velocity given by the simulations with Star-CCM+ is slightly lower and equal to 1.11 m/s.

Fig. 4.25 represent a comparison between CFD simulations and experiments as the large leading Taylor bubble reaches the ECT-Plane 1 and the ECT-Plane 2. From Fig. 4.25 the void fraction in the leading Taylor bubble can be obtained. The liquid film thickness could not be measured directly using the ECT planes in [5] and the expression proposed by Fernandes et al. [40] is used. This is given by:

$$\delta = \frac{D}{2} (1 - \sqrt{\varepsilon_{TB}})$$  \hspace{1cm} (4.9)

In Eq. (4.9) $\delta$ represents the liquid film thickness, $D$ represents the internal pipe diameter and $\varepsilon_{TB}$ is the void fraction in the Taylor bubble. The void fraction and the liquid film thickness obtained are summarised in Table 4.5.

The comparison between CFD simulations and experiments is again reasonably good.
Figure 4.23: Comparison between experiments and CFD simulations with Star-CCM+ (from [5]) and with OpenFOAM.

Figure 4.24: Time delay of a Taylor bubble passing along two different measuring locations in the pipe.
### Table 4.4: Flow development along the vertical riser in the simulation with OpenFOAM.

<table>
<thead>
<tr>
<th>Distance from the inlet [m]</th>
<th>Void fraction traces</th>
<th>PDF of void fraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 (15 pipe diameters)</td>
<td><img src="image1" alt="Graph" /></td>
<td><img src="image2" alt="Graph" /></td>
</tr>
<tr>
<td>2.0 (30 pipe diameters)</td>
<td><img src="image3" alt="Graph" /></td>
<td><img src="image4" alt="Graph" /></td>
</tr>
<tr>
<td>3.0 (45 pipe diameters)</td>
<td><img src="image5" alt="Graph" /></td>
<td><img src="image6" alt="Graph" /></td>
</tr>
<tr>
<td>4.0 (60 pipe diameters)</td>
<td><img src="image7" alt="Graph" /></td>
<td><img src="image8" alt="Graph" /></td>
</tr>
<tr>
<td>4.4 (66 pipe diameters)</td>
<td><img src="image9" alt="Graph" /></td>
<td><img src="image10" alt="Graph" /></td>
</tr>
<tr>
<td>4.489 (67 pipe diameters)</td>
<td><img src="image11" alt="Graph" /></td>
<td><img src="image12" alt="Graph" /></td>
</tr>
<tr>
<td>4.92 (73 pipe diameters)</td>
<td><img src="image13" alt="Graph" /></td>
<td><img src="image14" alt="Graph" /></td>
</tr>
</tbody>
</table>
Figure 4.25: Comparison between experiments and CFD simulations with Star-CCM+ (from [5]) and with OpenFOAM for the leading Taylor bubble (Start-up).

Table 4.5: Comparison between experiments and CFD simulations with Star-CCM+ (from [5]) and with OpenFOAM for the leading Taylor bubble: void fraction and liquid film thickness.

<table>
<thead>
<tr>
<th></th>
<th>Experiments</th>
<th>OpenFOAM</th>
<th>Star-CCM+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Void fraction in the Taylor bubble</td>
<td>ECT-Plane 1</td>
<td>0.77</td>
<td>0.79</td>
</tr>
<tr>
<td></td>
<td>ECT-Plane 2</td>
<td>0.76</td>
<td>0.80</td>
</tr>
<tr>
<td>Liquid film thickness [mm]</td>
<td>ECT-Plane 1</td>
<td>4.10</td>
<td>3.72</td>
</tr>
<tr>
<td></td>
<td>ECT-Plane 2</td>
<td>4.30</td>
<td>3.54</td>
</tr>
</tbody>
</table>
Conclusions

Two-phase gas-liquid upward flow in vertical pipes or risers offers a challenge to CFD simulation methods. This is particularly because different flow regimes may occur, such as bubbly flow, slug flow, churn flow, or annular flow. VOF is suggested to be the only available multiphase flow model applicable to segregated flow regimes. However, further investigations have revealed that VOF method in its original form in OpenFOAM and Fluent is not suitable for the flow conditions leading to slug or churn flow i.e. where a large slippage between the two phases exists. This shortcoming in the VOF method in OpenFOAM and Fluent is explained by the overpredicted dispersion between the phases. This gives a too low liquid holdup fraction, which in turn also gives a too low hydrostatic head and thus a too low pressure drop. The focus then turns to find the improvement possibilities in the VOF method. The approach of coupling VOF with a two-fluid model is adopted and further developed in this thesis with the hybrid multiphase solver multiphaseEulerFoam which is available in the OpenFOAM CFD package. VOF keeps a sharp interface between segregated flow structures (such as a liquid film in annular flow that is separated from the gas core) and the Euler-Euler (or the two fluid model) is able to represent dispersed regions, such as in bubbly flow or in the liquid slug body that has entrained gas bubbles. The results highlight the application of the presented CFD solver to predict flow regimes for gas-liquid vertical flows, without a priori knowledge on the flow pattern.

The hybrid model is applied in two examples for which experimental data exist.

Case study 1
The main goal to simulate vertical upward air-water churn flow through a pipe with 50.8 mm internal diameter is achieved. From a comparison of the results obtained from the CFD simulations in OpenFOAM and the experiments, the following conclusions can be drawn:

- The prediction by OpenFOAM using multiphaseEulerFoam captured the overall physics of the flow quite successfully. Time series of cross-sectional averaged void fraction and contour plots of liquid holdup and velocity showed that the hybrid model was able to distinguish the flow pattern in the pipe, being churn flow.

- Very good agreement was detected for the pressure drop and liquid holdup fraction. The pressure drop was 2400 Pa/m in the experiments and 2370 Pa/m in the CFD simulations. The liquid holdup fraction in the experiments was estimated to be 0.24 based on the measured pressure drop whereas for
the simulations the time averaged liquid holdup fraction was 0.21. The experimentally determined holdup was higher than the CFD prediction and this can be partly due to the assumption of zero frictional pressure drop.

Case study 2
A detailed simulation of the slug flow in a 6 m vertical pipe with a 0.067 m internal diameter with air and silicone oil has been successfully carried out. The results were compared with the experimental measurements and the CFD simulations with Star-CCM+ reported in literature. The main findings are:

• The CFD simulations with OpenFOAM have shown to be able to qualitatively capture the distinctive characteristics of the slug flow. Three regions were observed: the Taylor bubble, the falling film and the wake region. The Taylor bubble can be seen moving vertically upwards whereas the liquid film is moving downwards. In the wake region there were some entrained bubbles that were carried upwards.

• The slug flow pattern can be considered fully developed at 4 m, 60 pipe diameters.

• A reasonably good agreement between the CFD simulations and the experiments was obtained for the time series of cross-sectional averaged void fraction over three monitoring planes. The slug flow data had alternating periods of high and low void fraction. High void fraction marked the gas bubble passage, and low void fraction marked the passage of the liquid slug body with same entrained dispersed gas bubbles. However, the void fraction in the liquid slug body was seen to have lower values in the CFD simulations than in the experiments.

• The agreement of the PDFs of cross-sectional average void fraction between the CFD simulations and the experiments was quite reasonable. There was a twin-peaked PDF, with one peak being characteristic for the liquid slug body and the other peak being characteristic for the Taylor bubble. The void fraction in the liquid slug body showed lower values in the CFD simulations than in the experiments.

• A perfect agreement was found for the velocity of the slug unit for the fully developed flow between the CFD simulations with OpenFOAM and the experiments. The CFD simulations with Star-CCM+ predicted a slightly lower value.

• A satisfactory agreement with the experimental data was observed regarding the formation and the shape of the leading Taylor bubble and for the liquid film thickness. The peak of the void fraction in the leading Taylor bubble was larger in the Star-CCM+ and OpenFOAM results than with the experiments. The length of the leading bubble was correctly predicted with OpenFOAM whereas it was lower in the Star-CCM+ results.
Appendix A

blockMesh with m4

m4 is a macro processor that can be used for the parameterization of the blockMesh dictionary. The scripts that generate the meshes for the two cases are listed below.

A.1 Case study 1

Table A.1: m4 Script to generate the blockMesh dictionary for the Case study 1.

```
// Run using:
// m4 -P blockMeshDict.m4 > blockMeshDict

// m4 definitions:
m4_definem4_changequote([,])
m4_definem4_changecom(/)m4_changequote([,])
m4_definem4_eosyscmd perl -e 'use Math::Trig; printf ($1)'
m4_definem4_define (calc, [m4_eosyscmd (perl -e 'use Math::Trig; printf ($1)')])
m4_definem4_define (VCOUNT, 0)
m4_definem4_define (vlabel, [[/ /] Vertex $1 = VCOUNT m4_define ($1 , VCOUNT) m4_define ([VCOUNT], m4_incr (VCOUNT))])

// Mathematical constants:
m4_definem4_define (pi, 3.1415926536)

// Geometry
m4_definem4_define (rOut, 0.0254)
m4_definem4_define (rInt, calc (0.905511811023622 * rOut))
m4_definem4_define (rNumberOfCells1st, 10) // in the first O-grid
m4_definem4_define (rNumberOfCells2nd, 7) // in the second O-grid
m4_definem4_define (tNumberOfCells, 20)
m4_definem4_define (zABnumberOfCells, 320)
m4_definem4_define (rGrading1, 1.8)
m4_definem4_define (rGrading2, 2.7)
m4_definem4_define (zGrading, 1)

// Plane A:
m4_definem4_define (zA, 0)
m4_definem4_define (rA, rInt)
m4_definem4_define (rOutA, rOut)
m4_definem4_define (rRelA, 0.690537084398977)
m4_definem4_define (rRelAc, 0.85)

// Plane B:
m4_definem4_define (zB, 2.54)
m4_definem4_define (rB, rInt)
m4_definem4_define (rOutB, rOut)
m4_definem4_define (rRelB, 0.690537084398977)
m4_definem4_define (rRelBc, 0.85)
```
// Plane A:
(calc (rRelA*rA*cos(\pi/4)) - calc (rRelA*rA*sin(\pi/4)) zA) vlabel(A0)
(calc (rRelA*rA*cos(\pi/4)) calc (rRelA*rA*sin(\pi/4)) zA) vlabel(A1)
(calc (-rRelA*rA*cos(\pi/4)) calc (rRelA*rA*sin(\pi/4)) zA) vlabel(A2)
(calc (-rRelA*rA*cos(\pi/4)) - calc (rRelA*rA*sin(\pi/4)) zA) vlabel(A3)
(calc (rA*cos(pi/4)) - calc (rA*sin(pi/4)) zA) vlabel(A4)
(calc (rA*cos(pi/4)) calc (rA*sin(pi/4)) zA) vlabel(A5)
(calc (-rA*cos(pi/4)) calc (rA*sin(pi/4)) zA) vlabel(A6)
(calc (-rA*cos(pi/4)) - calc (rA*sin(pi/4)) zA) vlabel(A7)
(calc (rOutA*cos(pi/4)) - calc (rOutA*sin(pi/4)) zA) vlabel(A8)
(calc (rOutA*cos(pi/4)) calc (rOutA*sin(pi/4)) zA) vlabel(A9)
(calc (-rOutA*cos(pi/4)) calc (rOutA*sin(pi/4)) zA) vlabel(A10)
(calc (-rOutA*cos(pi/4)) - calc (rOutA*sin(pi/4)) zA) vlabel(A11)

// Plane B:
(calc (rRelB*rB*cos(pi/4)) - calc (rRelB*rB*sin(pi/4)) zB) vlabel(B0)
(calc (rRelB*rB*cos(pi/4)) calc (rRelB*rB*sin(pi/4)) zB) vlabel(B1)
(calc (-rRelB*rB*cos(pi/4)) calc (rRelB*rB*sin(pi/4)) zB) vlabel(B2)
(calc (-rRelB*rB*cos(pi/4)) - calc (rRelB*rB*sin(pi/4)) zB) vlabel(B3)
(calc (rB*cos(pi/4)) - calc (rB*sin(pi/4)) zB) vlabel(B4)
(calc (rB*cos(pi/4)) calc (rB*sin(pi/4)) zB) vlabel(B5)
(calc (-rB*cos(pi/4)) calc (rB*sin(pi/4)) zB) vlabel(B6)
(calc (-rB*cos(pi/4)) - calc (rB*sin(pi/4)) zB) vlabel(B7)
(calc (rOutB*cos(pi/4)) - calc (rOutB*sin(pi/4)) zB) vlabel(B8)
(calc (rOutB*cos(pi/4)) calc (rOutB*sin(pi/4)) zB) vlabel(B9)
(calc (-rOutB*cos(pi/4)) calc (rOutB*sin(pi/4)) zB) vlabel(B10)
(calc (-rOutB*cos(pi/4)) - calc (rOutB*sin(pi/4)) zB) vlabel(B11)
}

// Defining blocks:
blocks
(
    // Blocks between plane E and plane F:
    // block0 - positive x O-grid block 1st belt
    hex (A5 A1 A0 A4 B5 B1 B0 B4) AB
    (rNumberOfCells1st tNumberOfCells zABNumberOfCells)
    simpleGrading (rGrading1 zGrading)
    // block1 - positive y O-grid block 1st belt
    hex (A6 A2 A1 A5 B6 B2 B1 B5) AB
    (rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block2 — negative x O-grid block 1st belt
hex (A7 A3 A2 A6 B7 B3 B2 B6 ) AB
(rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block3 — negative y O-grid block 1st belt
hex (A4 A0 A3 A7 B4 B0 B3 B7 ) AB
(rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block4 — central O-grid block
hex (A0 A1 A2 A3 B0 B1 B2 B3 ) AB
(tNumberOfCells tNumberOfCells zABNumberOfCells)
simpleGrading (1 1 zGrading)
// block5 — positive x O-grid block 2nd belt
hex (A9 A5 A4 A8 B9 B5 B4 B8 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block6 — positive y O-grid block 2nd belt
hex (A10 A6 A5 A9 B10 B6 B5 B9 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block7 — negative x O-grid block 2nd belt
hex (A11 A7 A6 A10 B11 B7 B6 B10 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block8 — negative y O-grid block 2nd belt
hex (A8 A4 A7 A11 B8 B4 B7 B11 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)

// Plane A:
arc A0 A1 (calc (rRelAc*rRelA*rA) 0 zA)
ar A1 A2 (0 calc (rRelAc*rRelA*rA) zA)
ar A2 A3 (calc (rRelAc*rRelA*rA) 0 zA)
ar A3 A0 (0 calc (rRelAc*rRelA*rA) zA)
ar A4 A5 (rA 0 zA)
ar A5 A6 (0 rA zA)
ar A6 A7 (0 rA 0 zA)
ar A7 A4 (0 rA zA)
ar A8 A9 (rOutA 0 zA)
ar A9 A10 (0 rOutA zA)
ar A10 A11 (0 rOutA zA)
ar A11 A8 (0 rOutA zA)

// Plane B:
ar B0 B1 (calc (rRelBc*rRelB*rB) 0 zB)
ar B1 B2 (0 calc (rRelBc*rRelB*rB) zB)
ar B2 B3 (calc (rRelBc*rRelB*rB) 0 zB)
ar B3 B0 (0 calc (rRelBc*rRelB*rB) zB)
ar B4 B5 (rB 0 zB)
ar B5 B6 (0 rB zB)
ar B6 B7 (0 rB zB)
ar B7 B4 (0 rB zB)
ar B8 B9 (rOutB 0 zB)
ar B9 B10 (0 rOutB zB)
ar B10 B11 (0 rOutB zB)
ar B11 B8 (0 rOutB zB)

// Defining patches: (rNumberOfCells tNumberOfCells)
patches
{ patch inlet
}
Appendix A. blockMesh with m4

(A1 A5 A4 A0)
(A2 A6 A5 A1)
(A3 A7 A6 A2)
(A0 A4 A7 A3)
(A3 A2 A1 A0)
(A5 A9 A8 A4)
(A6 A10 A9 A5)
(A7 A11 A10 A6)
(A4 A8 A11 A7)

patch outlet

(B0 B4 B5 B1)
(B1 B5 B6 B2)
(B2 B6 B7 B3)
(B3 B7 B4 B0)
(B0 B1 B2 B3)
(B4 B8 B9 B5)
(B5 B9 B10 B6)
(B6 B10 B11 B7)
(B7 B11 B8 B4)

wall wallPipe

(A8 A9 B9 B8)
(A9 A10 B10 B9)
(A10 A11 B11 B10)
(A11 A8 B8 B11)

mergePatchPairs

// ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗

A.2 Case study 2

Table A.2: m4 Script to generate the blockMesh dictionary for the Case study 2.

//Run using:
//m4 −P blockMeshDict3.m4 > blockMeshDict

//m4 definitions:

m4_changecom(//)m4_changequote([,])
m4_define(calc , [m4_esyscmd(perl −e 'use Math::Trig; printf ($1)' ) ])
m4_define(VCOUNT, 0)
m4_define(vlabel, [[[// Vertex $1 = VCOUNT m4_define($1, VCOUNT)m4_define([VCOUNT ], m4_incr(VCOUNT))]])

//Mathematical constants:
m4_define(pi, 3.1415926536)

//Geometry with central square, 1st cylinder, 2nd cylinder, 3rd cylinder
m4_define(rOutOut, 0.0335) // radius pipe
m4_define(rOut, calc(0.797014925*rOutOut)) // radius 2nd cylinder
m4_define(rInt, calc(0.75522388*rOutOut)) // radius 1st cylinder
A.2. Case study 2

19 // Grid points:
20 m4_define(rNumberOfCells1st, 5) // first O-grid
21 m4_define(rNumberOfCells2nd, 1) // second O-grid
22 m4_define(rNumberOfCells3rd, 8) // third O-grid
23
24 m4_define(tNumberOfCells, 11) // central square
25 m4_define(zABnumberOfCells, 1500) // axis direction
26
27 m4_define(rGrading1, 1.5)
28 m4_define(rGrading2, 1)
29 m4_define(rGrading3, 2.5)
30 m4_define(zGrading, 1)
31
32 // Plane Inlet – A:
33 m4_define(zA, 0)
34 m4_define(rA, rInt)
35 m4_define(rOutA, rOut)
36 m4_define(rOutOutA, rOutOut)
37 m4_define(rRelA, 0.720762613)
38 m4_define(rRelAc, 0.85)
39
40 // Plane Outlet – B:
41 m4_define(zB, 6)
42 m4_define(rB, rInt)
43 m4_define(rOutB, rOut)
44 m4_define(rOutOutB, rOutOut)
45 m4_define(rRelB, 0.720762613)
46 m4_define(rRelBc, 0.85)
47
48 FoamFile
49 {
50 version 2.0;
51 format ascii;
52
53 convertToMeters 1;
54
tvertices (11)
55 // Plane A:
56 (calc(rRelA*rA*cos(pi/4)) - calc(rRelA*rA*sin(pi/4)) zA) vlabel(A0)
57 (calc(rRelA*rA*cos(pi/4)) calc(rRelA*rA*sin(pi/4)) zA) vlabel(A1)
58 (calc(-rRelA*rA*cos(pi/4)) calc(rRelA*rA*sin(pi/4)) zA) vlabel(A2)
59 (calc(-rRelA*rA*cos(pi/4)) -calc(rRelA*rA*sin(pi/4)) zA) vlabel(A3)
60 (calc(rA*cos(pi/4)) -calc(rA*sin(pi/4)) zA) vlabel(A4)
61 (calc(rA*cos(pi/4)) calc(rA*sin(pi/4)) zA) vlabel(A5)
62 (calc(-rA*cos(pi/4)) calc(rA*sin(pi/4)) zA) vlabel(A6)
63 (calc(-rA*cos(pi/4)) -calc(rA*sin(pi/4)) zA) vlabel(A7)
64 (calc(rOutA*rA*cos(pi/4)) calc(rOutA*rA*sin(pi/4)) zA) vlabel(A8)
65 (calc(rOutA*rA*cos(pi/4)) -calc(rOutA*rA*sin(pi/4)) zA) vlabel(A9)
66 (calc(-rOutA*rA*cos(pi/4)) calc(rOutA*rA*sin(pi/4)) zA) vlabel(A10)
67 (calc(-rOutA*rA*cos(pi/4)) -calc(rOutA*rA*sin(pi/4)) zA) vlabel(A11)
(calc (rOutOutA * cos (pi / 4)) - calc (rOutOutA * sin (pi / 4)) zA) v label (A12)
(calc (rOutOutA * cos (pi / 4)) calc (rOutOutA * sin (pi / 4)) zA) v label (A13)
(calc (rOutOutA * cos (pi / 4)) - calc (rOutOutA * sin (pi / 4)) zA) v label (A14)
(calc (rOutOutA * cos (pi / 4)) - calc (rOutOutA * sin (pi / 4)) zA) v label (A15)

// Plane B:
(calc (rRelB * rB * cos (pi / 4)) - calc (rRelB * rB * sin (pi / 4)) zB) v label (B0)
(calc (rRelB * rB * cos (pi / 4)) calc (rRelB * rB * sin (pi / 4)) zB) v label (B1)
(calc (rRelB * rB * cos (pi / 4)) - calc (rRelB * rB * sin (pi / 4)) zB) v label (B2)
(calc (rB * cos (pi / 4)) calc (rB * sin (pi / 4)) zB) v label (B3)
(calc (rB * cos (pi / 4)) - calc (rB * sin (pi / 4)) zB) v label (B4)
(calc (rB * cos (pi / 4)) calc (rB * sin (pi / 4)) zB) v label (B5)
(calc (rB * cos (pi / 4)) - calc (rB * sin (pi / 4)) zB) v label (B6)
(calc (rB * cos (pi / 4)) - calc (rB * sin (pi / 4)) zB) v label (B7)
(calc (rOutB * cos (pi / 4)) - calc (rOutB * sin (pi / 4)) zB) v label (B8)
(calc (rOutB * cos (pi / 4)) calc (rOutB * sin (pi / 4)) zB) v label (B9)
(calc (rOutB * cos (pi / 4)) - calc (rOutB * sin (pi / 4)) zB) v label (B10)
(calc (rOutB * cos (pi / 4)) - calc (rOutB * sin (pi / 4)) zB) v label (B11)
(calc (rOutB * cos (pi / 4)) calc (rOutB * sin (pi / 4)) zB) v label (B12)
(calc (rOutB * cos (pi / 4)) calc (rOutB * sin (pi / 4)) zB) v label (B13)
(calc (rOutB * cos (pi / 4)) calc (rOutB * sin (pi / 4)) zB) v label (B14)
(calc (rOutB * cos (pi / 4)) calc (rOutB * sin (pi / 4)) zB) v label (B15)

// Defining blocks:
blocks
{
// Blocks between plane A and plane B:
// block0 – positive x O-grid block 1st belt
hex (A5 A1 A0 A4 B5 B1 B4 ) AB
(rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block1 – positive y O-grid block 1st belt
hex (A6 A2 A1 A5 B6 B2 B5 ) AB
(rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block2 – negative x O-grid block 1st belt
hex (A7 A3 A2 A6 B7 B3 B6 ) AB
(rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block3 – negative y O-grid block 1st belt
hex (A4 A0 A3 A7 B4 B0 B3 B7 ) AB
(rNumberOfCells1st tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading1 1 zGrading)
// block4 – central O-grid block
hex (A0 A1 A2 A3 B0 B1 B2 B3 ) AB
(tNumberOfCells tNumberOfCells zABNumberOfCells)
simpleGrading (1 1 zGrading)
// block5 – positive x O-grid block 2nd belt
hex (A9 A5 A4 A8 B9 B5 B4 B8 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block6 – positive y O-grid block 2nd belt
hex (A10 A6 A5 A9 B10 B6 B5 B9 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block7 – negative x O-grid block 2nd belt
hex (A11 A7 A6 A10 B11 B7 B6 B10 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block8 – negative y O-grid block 2nd belt
hex (A12 A8 A7 A11 B8 B4 B7 B11 ) AB
(rNumberOfCells2nd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading2 1 zGrading)
// block9 – positive x O-grid block 3rd belt
hex (A13 A9 A8 A12 B9 B6 B8 B12 ) AB
(rNumberOfCells3rd tNumberOfCells zABNumberOfCells)
simpleGrading (rGrading3 1 zGrading)
// block10 – positive y O-grid block 3rd belt

hex (A14 A10 A9 A13 B14 B10 B9 B13) AB

(rNumberOfCells3rd tNumberOfCells zANumberOfCells)
simpleGrading (rGrading3 1 zGrading)

// block11 – negative x O-grid block 3rd belt
hex (A15 A11 A14 B15 B11 B14 B10 B13) AB
(rNumberOfCells3rd tNumberOfCells zANumberOfCells)
simpleGrading (rGrading3 1 zGrading)

// block12 – negative y O-grid block 3rd belt
hex (A12 A8 A11 A15 B12 B8 B11 B15) AB
(rNumberOfCells3rd tNumberOfCells zANumberOfCells)
simpleGrading (rGrading3 1 zGrading)

edges
Plane A:
arc A0 A1 (calc (rRelAc*sRelA*rA) 0 zA)
arc A1 A2 (0 calc (rRelAc*sRelA*rA) zA)
arc A2 A3 (calc (rRelAc*sRelA*rA) 0 zA)
arc A3 A0 (0 calc (rRelAc*sRelA*rA) zA)
arc A4 A5 (rA 0 zA)
arc A5 A6 (0 rA zA)
arc A6 A7 (−rA 0 zA)
arc A7 A4 (0 −rA zA)
arc A8 A9 (rOutA 0 zA)
arc A9 A10 (0 rOutA zA)
arc A10 A11 (−rOutA 0 zA)
arc A11 A8 (0 −rOutA zA)
arc A12 A13 (rOutOutA 0 zA)
arc A13 A14 (0 rOutOutA zA)
arc A14 A15 (−rOutOutA 0 zA)
arc A15 A12 (0 −rOutOutA zA)

Plane B:
arc B0 B1 (calc (rRelBc*sRelB*rB) 0 zB)
arc B1 B2 (0 calc (rRelBc*sRelB*rB) zB)
arc B2 B3 (calc (rRelBc*sRelB*rB) 0 zB)
arc B3 B0 (0 calc (rRelBc*sRelB*rB) zB)
arc B4 B5 (rB 0 zB)
arc B5 B6 (0 rB zB)
arc B6 B7 (−rB 0 zB)
arc B7 B4 (0 −rB zB)
arc B8 B9 (rOutB 0 zB)
arc B9 B10 (0 rOutB zB)
arc B10 B11 (−rOutB 0 zB)
arc B11 B8 (0 −rOutB zB)
arc B12 B13 (rOutOutB 0 zB)
arc B13 B14 (0 rOutOutB zB)
arc B14 B15 (−rOutOutB 0 zB)
arc B15 B12 (0 −rOutOutB zB)

// Defining patches:
patches
patch inlet
(A1 A5 A4 A0)
(A2 A6 A5 A1)
(A3 A7 A6 A2)
(A0 A4 A7 A3)
(A3 A2 A1 A0)
(A5 A9 A8 A4)
(A6 A10 A9 A5)
(A7 A11 A10 A6)
(A4 A8 A11 A7)
(A9 A13 A12 A8)
(A10 A14 A13 A9)
(A11 A15 A14 A10)
(A8 A12 A15 A11)
patch outlet
{
  (B0 B4 B5 B1)
  (B1 B5 B6 B2)
  (B2 B6 B7 B3)
  (B3 B7 B4 B0)
  (B0 B1 B2 B3)
  (B4 B8 B9 B5)
  (B5 B9 B10 B6)
  (B6 B10 B11 B7)
  (B7 B11 B8 B4)
  (B8 B12 B13 B9)
  (B9 B13 B14 B10)
  (B10 B14 B15 B11)
  (B11 B15 B12 B8)
}

wall wallPipe
{
  (A12 A13 B13 B12)
  (A13 A14 B14 B13)
  (A14 A15 B15 B14)
  (A15 A12 B12 B15)
}

mergePatchPairs
{
}

// ****************************************************************************** //
Appendix B

Reynolds-averaged modelling for the mixture

For the release version of multiphaseEulerFoam 2.2.x, RANS is not a standard option. The changes required in order to use a RANS model for the mixture are listed below.

B.1 Listings

B.1.0.1 In UEqns.H:

Table B.1: Changes in UEqns.H

1. phaseModel& phase = iter();
2. const volScalarField& alpha = phase;
3. volVectorField& U = phase.U();
4. volScalarField nuEff(turbulence->nut() + iter().nu());
5. UEqns.set

B.1.0.2 In createFields.H:

Table B.2: Changes in createFields.H

1. autoPtr<incompressible::turbulenceModel> turbulence
2. ( incompressible::turbulenceModel::New(U, phi, fluid) )

B.1.0.3 In multiphaseEulerFoam.H:

Table B.3: Changes in multiphaseEulerFoam.H

1. while (pimple.loop())
2. {
3.   turbulence->correct();
4.   fluid.solve();
5.   rho = fluid.rho();
6.   #include "zonePhaseVolumes.H"
7. }
B.2 \textbf{yPlus function object}

\texttt{yPlusRAS} is a postProcessing function object that calculates $y^+$ \cite{76} for incompressible cases employing RANS turbulence.

One challenge in CFD is how to treat the thin near-wall sublayer, where viscous effects become important. In the near-wall region additional numerical difficulties/costs arise and fine grids are needed near the wall. However if a wall-function is applied, there is no need to resolve the viscous sublayer ($y^+ < 5$) and the buffer layer, and thus the traditional industrial solution is to use wall-functions.

For multiphase flow cases the utility \texttt{yPlusRAS} does not work, since it is looking in transportProperties for single phase transport properties. The code is changed in order to deal with the mixture properties.

\begin{table}[h]
\centering
\begin{tabular}{l}
\hline
\textbf{Table B.4: Changes in yPlus_{mixture}} \\
\hline
#include "fvCFD.H" \\
#include "incompressible/singlePhaseTransportModel/singlePhaseTransportModel.H" \\
#include "incompressibleTwoPhaseMixture.H" \\
#include "incompressible/RAS/RASModel/RASModel.H" \\
#include "nutWallFunction/nutWallFunctionFvPatchScalarField.H" \\
#include "fluidThermo.H" \\
#include "compressible/RAS/RASModel/RASModel.H" \\
#include "mutWallFunction/mutWallFunctionFvPatchScalarField.H" \\
#include "wallDist.H" \\
\hline
\end{tabular}
\end{table}

\begin{table}[h]
\centering
\begin{tabular}{l}
\hline
\textbf{Table B.5: Changes in yPlus_{mixture}} \\
\hline
#include "createPhi.H" \\
// singlePhaseTransportModel laminarTransport(U, phi); \\
incompressibleTwoPhaseMixture twoPhaseProperties(U, phi); \\
autoPtr<incompressible::RASModel> RASModel \\
( \\
\hline
\end{tabular}
\end{table}
Appendix C

fvSolution

Table C.1: fvSolution

<table>
<thead>
<tr>
<th>Field</th>
<th>Operation</th>
<th>Version</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>fvSolution</td>
<td>OpenFOAM: The Open Source CFD Toolbox</td>
<td>2.2.2</td>
<td><a href="http://www.OpenFOAM.org">www.OpenFOAM.org</a></td>
</tr>
</tbody>
</table>

FoamFile
{
version 2.0;
format ascii;
class dictionary;
location system;
object fvSolution;
}
PIMPLE
{
nCorrectors 3;
nNonOrthogonalCorrectors 1;
OuterCorrectors 1;
AlphaCorr 2;
AlphaSubCycles 5;
turbOnFinalIterOnly false;
}
solvers
{
p
   solver GAMG;
   agglomerator faceAreaPair;
   mergeLevels 1;
   cacheAgglomeration true;
   nCellsInCoarsestLevel 1000;
   tolerance 1e-8;
   relTol 0;
   smoother GaussSeidel;
   nPreSweeps 0;
   nPostSweeps 2;
   nFinestSweeps 2;
   minIter 3;
   maxIter 1500;
}
pFinal
{
   $p$
   tolerance 1e-8;
Appendix C. fvSolution

```c
    relTol  0;
}
pcorr
{
    $pFinal;
    tolerance  1e-8;
    relTol  0;
}
U
{
    solver PBiCG;
    preconditioner DILU;
    tolerance  1e-8;
    relTol  0;
    minIter  3;
}
UFinal
{
    $U;
    tolerance  1e-8;
    relTol  0;
    minIter  3;
}
k
{
    $U;
}
kFinal
{
    $UFinal;
}
omega
{
    $U;
}
omegaFinal
{
    $UFinal;
}
}
relaxationFactors
{
    fields
    {
        p  1;
    }
equations
    {
        U  1;
        k  1;
        omega  1;
        alpha.air  1;
        alpha.water  1;
    }
}```
Appendix D

fvSchemes

Table D.1: fvSchemes

<table>
<thead>
<tr>
<th>Field</th>
<th>Operation</th>
<th>Manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>F i e l d</td>
<td>OpenFOAM: The Open Source CFD Toolbox</td>
<td></td>
</tr>
<tr>
<td>O per a t i o n</td>
<td>Version: 2.2.2</td>
<td></td>
</tr>
<tr>
<td>A nd</td>
<td>Web: <a href="http://www.OpenFOAM.org">www.OpenFOAM.org</a></td>
<td></td>
</tr>
</tbody>
</table>

FoamFile
{
  version 2.0;
  format ascii;
  class dictionary;
  location system;
  object fvSchemes;
}

ddtSchemes
{
  default Euler;
}

gradSchemes
{
  default cellMDLimited leastSquares 1;
}

divSchemes
{
  "div\((\phi,\alpha.\star)\)" Gauss vanLeer01;
  "div\((\phi r,\alpha.\star,\alpha.\star)\)" Gauss vanLeer01;
  "div\((\phi l e m A l p h a . \star,\star)\)" Gauss limitedLinearV 1;
  div(Rc) Gauss linear;
  "div\((\phi l e m \star,\star)\)" Gauss limitedLinearV 1;
  div(\phi ,k) Gauss limitedLinear 1;
  div(\phi ,omega) Gauss limitedLinear 1;
  div(\((\mu E f f * d e v (T(\text{grad}(U))))\) Gauss linear;
}

laplacianSchemes
{
  default Gauss linear limited 1;
}

interpolationSchemes
{
  default linear;
}
snGradSchemes
{
    default limited 1;
}

fluxRequired
{
    default no;
    p;
    pcorr ;
    "alpha.*" ;
}
Appendix E

Velocity reconstruction in pEqn.H

Surface tension is included in the construction of the pressure equation itself, but it has been omitted from the reconstruction of the velocity field. Therefore the surface tension force needs to be added to the fvc::reconstruct(). In the original code line 6 is missing.

Table E.1: Velocity reconstruction in pEqn.H

<table>
<thead>
<tr>
<th>Line</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>phase.U() =</td>
</tr>
<tr>
<td>2</td>
<td>HbyAs[phasei]</td>
</tr>
<tr>
<td>3</td>
<td>+ fvc::reconstruct</td>
</tr>
<tr>
<td>4</td>
<td>(</td>
</tr>
<tr>
<td>5</td>
<td>rAlphaAUfs[phasei]*(g &amp; mesh.Sf())</td>
</tr>
<tr>
<td>6</td>
<td>+ rAlphaAUfs[phasei]*fluid.surfaceTension(phase)*mesh.</td>
</tr>
<tr>
<td>7</td>
<td>magSf()/phase.rho()</td>
</tr>
<tr>
<td>8</td>
<td>+ rAlphaAUfs[phasei]*mSfGradp/phase.rho()</td>
</tr>
<tr>
<td>9</td>
<td>) ;</td>
</tr>
</tbody>
</table>
Bibliography


