Diese Arbeit wurde vorgelegt am Lehrstuhl für Fluidverfahrenstechnik
The present work was submitted to Chair of Fluid Process Engineering

Entwicklung eines vereinfachten strömungsdynamischen Modells für einen Mehrphasenschlaufenreaktoren auf Basis einer komplexen CFD-Simulation

Development of a Simplified Fluid Dynamics Model for a Multiphase Loop Reactor Based on a Complex CFD Simulation

Bachelor-Thesis
presented by
Tomas Satura

Examiner: **Prof. Dr.-Ing. Andreas Jupke**
Dr.-Ing. Sebastian Kaminski

Advisor: Maximilian von Campenhausen

Aachen, September 29, 2019

Rheinisch-Westfälische Technische Hochschule Aachen
Aachener Verfahrenstechnik

Aufgabenstellung

Viele Fermentationen müssen begast werden und sind auf Grund von Produktinhibierung in ihrer Effizienz limitiert. Mit dem neuartigen Konzept eines Mehrphasenschlaufenreaktors kann die Ausbeute dieser Fermentationen erhöht werden, indem in einem Apparat zeitgleich Luft bzw. Sauerstoff zugeführt und inhibierende Stoffe über eine flüssig-flüssig Extraktion abgetrennt werden.

In dieser Arbeit soll die Schlaufenströmung weiter optimiert werden, sodass bestmögliche Strömungsbedingungen für die Extraktion festgelegt werden können. Dazu wird das Strömungsverhalten unter Einfluss verschiedener Betriebseinstellungen in einer mehrphasigen CFD-Simulation untersucht. Als Softwaretool steht die Opensource Software OpenFOAM zu Verfügung.

______________________________ ______________________________

Ort und Datum der Abgabe Unterschrift des Betreuer

Eidesstattliche Erklärung

Name, Vorname Matrikelnummer (freiwillige Angabe)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/Masterarbeit* mit dem Titel

selbständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel verwendet. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Ich erkläre mich hiermit einverstanden, dass die vorliegende Arbeit in der Lehrstuhlbibliothek aufbewährt wird und kopiert werden darf und habe keine Einwände gegen eine Veröffentlichung der gesamten Arbeit oder von einzelnen Teilen daraus.

Ort, Datum Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

§156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit be-gangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vor-schriften des §158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Ort, Datum Unterschrift

Erklärung

Ich erkläre mich hiermit einverstanden, dass die vorliegende Arbeit in der Lehstuhlbibliothek aufbewahrt wird und kopiert werden darf und habe keine Einwände gegen eine Veröffentlichung der gesamten Arbeit oder von einzelnen Teilen daraus.

Ich bin damit einverstanden, dass die RWTH Aachen folgende Daten zu meiner Person im Internet veröffentlicht: Name, Vorname, Titel der Arbeit.

Mir ist bekannt,

□ dass ich diese Einwilligung jederzeit schriftlich mit Wirkung für die Zukunft widerrufen kann und meine elektronisch gespeicherten Daten unverzüglich gelöscht werden müssen und

□ die Aachener Verfahrenstechnik bei Widerruf meiner Einwilligung verpflichtet ist, mein Werk aus der Bibliothek des Instituts/Lehrstuhls zu entfernen.

Ort, Datum

Unterschrift

Acknowledgement

Simulations were performed with computing resources granted by RWTH Aachen University under project thes0609.

Since the most abbreviated version of the list of people I am grateful to does not fit in this page, I decided to use the space of Appendix B for this purpose. Please, kindly refer to this section if you are hoping to find yourself in this work.

"When I meet God, I am going to ask him two questions:
Why relativity?
And why turbulence?
I really believe he will have an answer for the first."

Werner Heisenberg

Contents

List of Figures

List of Tables

1. Introduction

Airlift loop reactors (ALRs) are a class pneumatically agitated multiphase reactors, that have witnessed a significant rise in popularity in the past two decades (Zhang *et al.* , 2016). In their review, Zhang *et al.* summarise some of the notable characteristics of ALRs such as relatively straight-forward design, favourable mass and heat transfer characteristics or generally low power consumption. Additionally, agitation in ALRs is facilitated solely pneumatically, which enables presence of biological material susceptible to mechanical damage by conventional means of agitation (Simcik *et al.* , 2011). Hence, favourable properties of these reactors have attracted significant research and industrial interest for applications such as production of organic compounds, waste-water treatment or biochemical processes Markos *et al.* (2004).

ALRs evolved from wide-spread bubble-column reactors by splitting the reactor into four main distinguishable parts: *riser*, *downcomer*, *gas separator* and *bottom clearance* (Lu *et al.* , 1994). The governing principle of the operation is inducing an upward flow in the riser by virtue of the lift of air, which enters at the bottom of the riser. In addition to the aggitation, aeration of the reaction medium is facilitated, enabling the desired reaction. Subsequently, the liquid-gas separation occurs at the top of the reactor and the fluid is recirculated via the downcomer to the bottom clearance and back to the riser. This operating principle is depicted on some common ALR set-ups in Figure 1.1.

More recently, an upgrade to the traditional ALR was proposed, whereby extraction liquid is introduced into the downcomer (Bednarz *et al.* , WO 2017/149099 AI). Thus, liquid-liquid extraction is integrated, permitting separation of the product inside the reactor. The requirement for the extraction liquid is for it to be of lower density than the reaction medium, in order for it to rise through the downcomer by the virtue of lift. The product-rich extraction liquid is then removed from the top surface.

It follows that for design of such multi-phase loop reactor (MLR), thorough understanding of its hydrodynamics is of paramount importance. Just as in case of ALRs, both empirical and

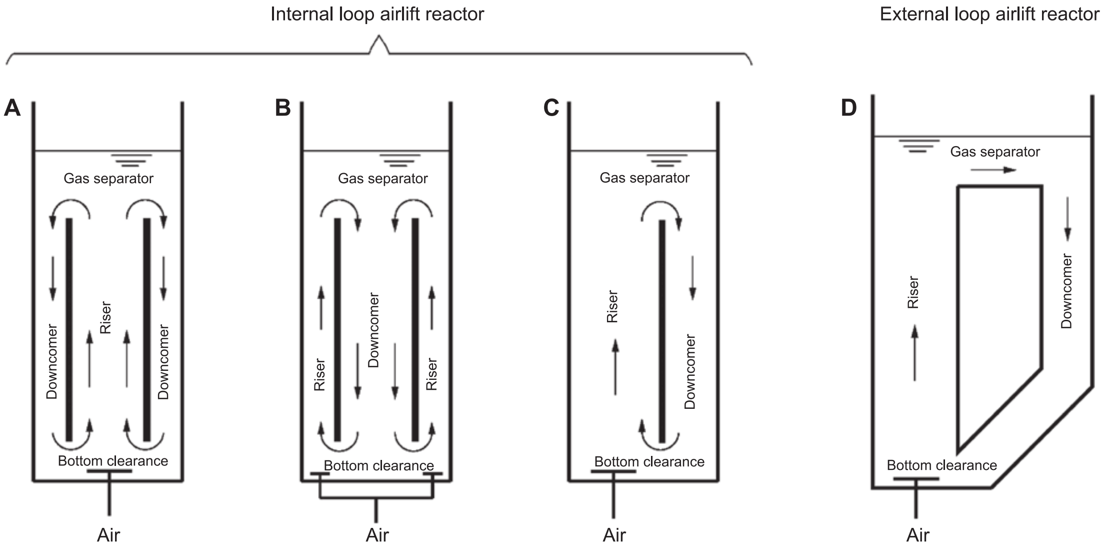


Figure 1.1.: Common set-ups of airlift reactors (Zhang *et al.* , 2016)

computational approach of obtaining data for at least preliminary description have been used, with the earlier work on ALRs relying almost exclusively on development of experiment-aided correlations ((Chisti *et al.* , 1988), (Markos *et al.* , 2004)). However, given the multitude of phenomena which are of research interest, the computational approach for research into the flow behaviour would be more convenient, given the time and effort required for experimental examination of the various design and operation parameters. Nevertheless, as modelling of multi-phase flow poses a significant research area even at the present level of computational fluid dynamics (CFD), such simulation is not straight-forward. Yet, advancements in the CFD made in the past 20 years enabled numerous researchers to perform simulation of multi-phase reactors to sarisfactory level of accuracy ((Monahan *et al.* , 2005), (Selma *et al.* , 2010), (Xu *et al.* , 2014),(Bhusare *et al.* , 2017)).

The underlying advancements in CFD are based on research into computational methods of multi-phase modelling, with notable contributions from Ubbink (1997), Rusche (2002) or Weller (2008). Most of the current simulations of ALRs are directly founded on the work of Rusche, which was focused on modelling dispersed multi-phase flow of high volume fractions. This significantly reduced computational expense of dispersed multi-phase modelling, making a CFD simulation of both ALRs and MLRs viable.

This motivation for this thesis is centered around a concept of a novel MLR which is intended for bio-chemical production of rhamnolipid. As it is expected the MLR design would benefit from improved understanding of the underlying hydrodynamics obtained by the computational

approach, the thesis builds on the recent CFD advancements to create a simplified MLR simulation. Finally, the quantitative data from the simulations are fitted by a correlation, which aims to describe internal re-circulation for the purposes of preliminary reactor design.

2. Theoretical Background of Relevant Computational Fluid Dynamics Concepts

CFD is an engineering discipline that employs data structures and numerical methods to simulate problems relating fluid-based systems. This chapter outlines some of its underlying mathematical principles and techniques.

2.1. Transport Equations

A flow can be described by balance equations of its momentum, mass, energy and present species. For a general quantity ϕ (either scalar, vector or tensor), a general conservation equation would be of the form:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\vec{U}\phi) = S(\phi) + \nabla \cdot (D_T \nabla \phi) \tag{2.1}$$

In the Equation (2.1), $\vec{U}$ corresponds to flow velocity and D_T to diffusivity of the given quantity. The left-hand side terms account for the corresponding rate of accumulation and convection, while the right-hand side terms describe its source and diffusion. For mass, given the conservation law, this equation takes the form of continuity equation:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{U}) = 0 \tag{2.2}$$

Similarly, the equation can be written for conservation of momentum, which is commonly referred to as the Navier-Stokes equation:

$$\frac{\partial \rho \vec{U}}{\partial t} + \nabla(\rho \vec{U} \otimes \vec{U}) = -\nabla p + \eta \nabla^2 \vec{U} + \frac{1}{3}\eta \nabla(\nabla \cdot \vec{U}) + \rho \vec{g} \tag{2.3}$$

Solution of the Navier-Stokes equation coupled with the continuity equation yields corresponding pressure-velocity flow-field. However, due to implicit and non-linear nature of the equation system, the solution is complex and requires application of numerical techniques and, for virtually all practical applications, significant extent of modelling. In the following sections, some of the important elements of such solution applied for multi-phase flow are outlined.

2.2. Finite Volume Method

There are three main numerical methods that are used to various extent in different fields of engineering. These methods are based on finite differences, elements and volumes. Out of them, most of the CFD softwares utilise the finite volumes method (FVM), as it inherently guarantees the laws of conservation (Versteeg & Malalasekera, 2007). Its ground principle is splitting the domain into a number of finite volumes called (*cells*). There is relatively vast theory on particular geometries of the cells, as well as advantages and mathematical implications thereof. Examples of these geometries are depicted in Figure 2.1. Since this work operates with hexahedral mesh, the other mesh types will not be further analysed. However, a very good introduction to this topic can be found in Miles & Farrashkhalvat (2003), together with mathematical methods for mesh generation employed in this work. Splitting the domain into cells enables numerical solution of the transport equations, whereby their integral is evaluated over the cell volume as:

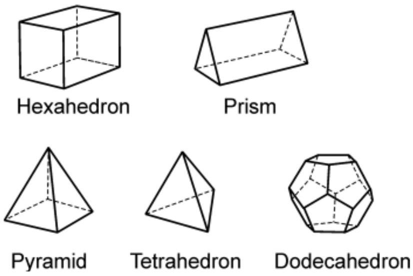


Figure 2.1.: Various cell geometries of a 3D mesh (Andersson *et al.* , 2012)

$$\iiint\limits_V \frac{\partial \phi}{\partial t} dV + \iiint\limits_V \nabla \cdot (\vec{U}\phi) dV = \iiint\limits_V S(\phi) dV + \iiint\limits_V \nabla \cdot (D\nabla\phi)\, dV \tag{2.4}$$

2.2.1. Discretization

Numerical solution to the integral form follows discretisation, in which each term is transformed to a linear algebraic contribution. This section provides a brief description of some of the underlying principles and schemes used in this process.

Firstly, each cell can be considered to be of infinitesimal dimensions. This permits direct evaluation of the source term, assuming its value is constant throughout the cell volume. Thus, its integral can be expressed in terms of its averaged value within the cell:

$$\iiint\limits_V S(\phi)dV = \overline{S}(\phi)V \tag{2.5}$$

Secondly, Gauss's theorem can be used to transform the divergence terms. As an example, the convection term from the Equation 2.4 would become:

$$\iiint\limits_V \nabla \cdot (\vec{U}\phi)dV = \oiint\limits_A (\vec{U}\phi) \cdot \hat{n}dA \tag{2.6}$$

In Equation 2.6, the term is integrated over the cell's outer faces with corresponding normal vectors $\hat{n}$. The hexahedral computational cell in question is depicted in Figure 2.2, in which N, S, E, W, T and B designate North, South, East, West, Top and Bottom respectively, and are used to refer to the respective faces.

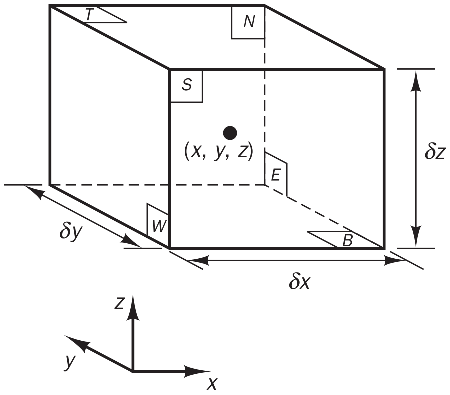


Figure 2.2.: A schematic of a hexahedral cell (Versteeg & Malalasekera, 2007)

Using subscript j referring to N, S, E, W, T and B, the divergence term can be evaluated as a sum:

$$\oiint_A (\vec{U}\phi) \cdot \hat{n} dA = \sum_j (A\vec{U}\phi)_j \tag{2.7}$$

where again values of ϕ and $\vec{U}$ are assumed constant over the face area A_j.

Similarly, the diffusion term can be transformed in the same fashion, followed by additional discretization of the gradient. Transformation of the gradient in any direction can be achieved by assuming linearity between the cell centre and the face in the direction. With the assumption of infinitesimal dimension of the cell, finite difference method can then be applied. For example, the difference in the direction of face T would be transformed as:

$$\left(\frac{\partial \phi}{\partial z}\right)_T = \frac{\phi_T - \phi_C}{\delta z/2} \tag{2.8}$$

Finally, the time derivative is commonly discretized using the forward Euler method (not to be confused with Euler-Euler method for multi-phase flow). This can be expressed as:

$$\frac{\partial \phi}{\partial t} = \frac{\phi_{n+1} - \phi_n}{{}_\triangle t} \tag{2.9}$$

where ${}_\triangle t$ is the time-step from instance n to $n+1$. Thus, ϕ is obtained iteratively using its previously computed value and the selected time step. Appropriate selection of its size is critical for convergence and is usually adjusted according to specified maximum *Courant number (Co)*. This is defined as:

$$Co = \frac{u_\triangle t}{{}_\triangle z} \tag{2.10}$$

Since maximum Co is constant, the time step is adjusted to satisfy Equation 2.10 for the given non-dimensionalysed velocity u and cell-size ${}_\triangle z$.

For all the spatial transformations, values of the transported quantities at the faces of each cell are required. However, since FVM computes and stores values corresponding to cell centres only, further discretization schemes are needed to estimate values of the transported quantities from their values at adjacent cell-centres.

2.2.1.1. Upwind Scheme

The upwind scheme approximates spatial variation of ϕ as a piece-wise constant function. The so-called *first-order upwind* conducts this approximation within the whole volume of the computational cell, hence for faces T and B it can be formulated as:

$$\phi_T = \phi_B = \phi_C \tag{2.11}$$

Since for a face shared between two cells, this results in it being assigned two different values of ϕ from the two adjacent cell centres, a further condition is imposed accepting only the value in the direction in which the solution is performed. This scheme is generally considered very robust, however it is only able to deliver first-order accuracy. Therefore, when needed, its accuracy can be increased introducing a modification into *second-order upwind scheme*:

$$\phi_{\mathrm{T}} = \phi_C + \frac{(\phi_C - \phi_{C.B})(z_T - z_C)}{z_C - z_{C.B}} \tag{2.12}$$

In Equation 2.12, subscript $C.B$ refers to the centre of the cell adjacent to the face B from the bottom. Even though this formulation generally enhances the precision, it might cause the solution to wind-up, as it is unbounded.

2.2.1.2. Linear Scheme

In the linear scheme, the spatial distribution of ϕ is approximated as a piece-wise linear function, hence the value on the face shared between two cells is obtained by interpolation between the corresponding centres. This results in only a single value assigned to a face, hence no further conditions are required. However, bounds can be introduced to limit the output value, as in some cases unbounded linear scheme can wind up to non-physical values. In contrast to first-order upwind, this scheme is second-order accurate.

2.2.1.3. Van Leer Scheme

Several schemes are based on mixing other discretization schemes based on certain condition. Van Leer scheme is based on such mixing, in which first- or second-order upwind are implemented in alternative fashion, making it benefit from stability of the former and accuracy of the latter.

The condition introduced is based on the second derivative of ϕ and can be formulated as:

$$|\phi_T + \phi_B - 2\phi_C| > |\phi_T - \phi_B| \tag{2.13}$$

If this condition is fulfilled, the van Leer scheme implements the first-order upwind scheme as in Equation 2.11. Otherwise, it proceeds to the second-order upwind scheme (Equation 2.12).

2.3. Turbulence Modelling

Albeit solution of the field described by the transport equations alone is possible, it is in practice rarely used for industrial application. The reason is the phenomenon of *turbulence*, which is even presently not sufficiently understood (Andersson *et al.* , 2012). The essence of turbulence are random changes to the velocity field, arising as inertial amplifications of the inlet disturbances. These are damped by viscous diffusion and amplified by the flow's inertia. Therefore, *Reynolds number* is often used to set a limit between laminar and turbulent regime:

$$Re = \frac{UL}{\nu} \tag{2.14}$$

In Equation 2.14 U refers to the flow's speed, ν to its kinematic viscosity and L to the characteristic length. According to Andersson *et al.* (2012), the computational cost t for such simulation is given by (Sc designates *Schmidt number*, which is of the order of `unity` for gasses and of the order of thousand for water):

$$t \propto Re^3 Sc^2 \tag{2.15}$$

Thus, *Direct Numerical Simulation (DNS)*, which is based on direct solution of the transport equations, becomes very expensive with increasing turbulence, giving rise to turbulence models that reduce computational cost.

2.3.1. Large Eddy Simulation (LES)

LES is based on Kolmogorov hypothesis, which states that eddies below certain length-scale exhibit isotropic behaviour. Therefore, resolution of the mesh is adapted to the so-called Kolmogorov length-scale specific for the given flow, and a spatial filter (analogous to low-pass filter in electrical engineering) is employed to filter out disturbances in the flow-field below this scale. Versteeg & Malalasekera (2007) formulate this spatial filter as:

$$\overline{\phi}(\vec{x},t) = \iiint G(\vec{x}-\vec{\xi},\Delta)\phi(\xi,t)d^3\xi \tag{2.16}$$

Where the used notation designates the cutoff length-scale as Δ, $\vec{x}$ as a mean position vector and $\vec{\xi}$ as its fluctuation vector. Thus, the filtered value $\overline{\phi}$ is obtained using filtering function G. Versteeg & Malalasekera (2007) further give an example of commonly implemented G, referred to as *top-hat* or *box filter*:

$$G(\vec{x}-\xi,\Delta) = \begin{cases} 1/\Delta^3 & |\vec{x}-\vec{x}'| \leq \Delta/2 \\ 0 & |\vec{x}-\vec{x}'| > \Delta/2 \end{cases} \tag{2.17}$$

Thus, the system only solves the source equations for large and medium eddies, while modelling effect of the small eddies by introducing a hypothetical quantity of *sub-grid viscosity*. Sagaut (2002) explains that this method is largely thought to have great potential to become the dominant method for turbulence modelling. Nevertheless, as the presently used LES schemes such as *Smagorinsky-Lilly* require rather-fine mesh refinement, employing this method is still beyond available capabilities for many industrial and academical applications.

2.3.2. Reynolds Averaged Navier-Stokes Equations (RANS)

The RANS approach was originally proposed by Reynolds more than hundred years ago (Andersson *et al.* , 2012). Its principle together with differences in its approach from DNS and LES are depicted in Figure 2.3. Unlike DNS or LES, RANS does not simulate effects of eddies directly, but instead accounts for these effects by introducing effective viscosity given as:

$$\nu_{eff} = \overline{\nu} + \nu_t \tag{2.18}$$

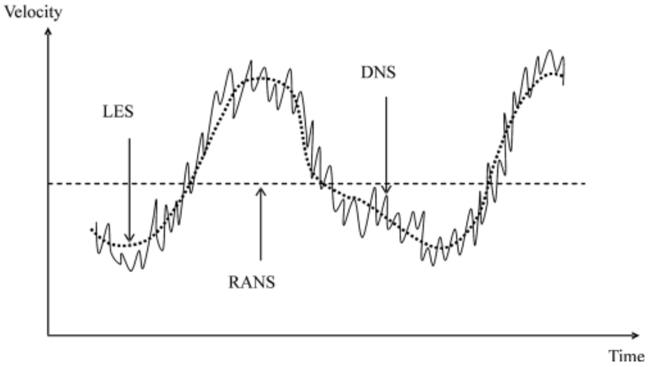


Figure 2.3.: Graphical comparison into resolution level of DNS, LES and RANS (Andersson *et al.* , 2012)

The effective viscosity has two contributions, the mean viscosity and turbulent viscosity. Similarly to the sub-grid viscosity in LES, turbulent viscosity has no physical meaning and is rather a statistical measure accounting for turbulent dissipation of the energy. Since this is the least expensive method to model turbulent behaviour, it has also been most commonly used in CFD practice (Andersson *et al.* , 2012). Numerous RANS models have been proposed, differing in the way of estimating ν_t, making them suitable either for general or specific applications.

2.3.2.1. Standard $k - \epsilon$ Model

Arguably the most wide-spread RANS model is the $k - \epsilon$ *model* (Andersson *et al.* , 2012), in which the turbulent viscosity is calculated from turbulence kinetic energy k and turbulent dissipation ϵ as:

$$\nu_t = C_\mu \frac{k^2}{\epsilon} \tag{2.19}$$

Here, turbulence kinetic energy k is defined as:

$$k = \frac{1}{2}\left(\overline{(u')^2} + \overline{(v')^2} + \overline{(w')^2}\right) \tag{2.20}$$

and viscous dissipation rate ϵ as:

$$\epsilon = \frac{k^{3/2}}{l_e} \tag{2.21}$$

where l_e designates turbulence length-scale, and u', v' and w' are deviations from the average value of velocity U in x-,y- and z- direction respectively. Both k and ϵ are propagated via a separate transport equation of the form of Equation 2.1. In the model, a number of empirical coefficients are introduced for the estimations, which together with C_μ provide the closure.

2.3.2.2. Realizable $k - \epsilon$ Model

Realizable $k - \epsilon$ model features realizability constraint on the predicted stress tensor (Andersson *et al.* , 2012). Andersson *et al.* (2012) further explain the difference between the standard and realizable model on the stress tensor in Reynolds stress tensor equation:

$$\overline{u^2} = \frac{2}{3}k - 2\nu_t \frac{\partial U}{\partial x} \tag{2.22}$$

Even though $\overline{u^2}$ by definition cannot take negative value, modelling ν_t by means of the standard $k - \epsilon$ model may render its value negative. Therefore, realizable $k - \epsilon$ model introduces variable value of C_μ, which corrects for this deviation, thus this model has been proven to give more accurate predictions of turbulence behaviour (Andersson *et al.* , 2012).

2.4. Multi-phase Modelling

Multi-phase nature of the flow introduces further challenges into the computational approach. Besides DNS, which would permit obtaining of the profiles directly, there are several other dominant modelling-based methodologies. These vary in their limitations and applicability.

2.4.1. Euler-Euler Approach

An important distinction in multi-phase modeling is based on the phases being dispersed or segregated. In case of dispersed phases, there are several commonly used approaches. These include for example an Euler-Lagrange approach, which tracks each individual particle of the dispersed phase within the continues phase. Thus, it makes it only suitable for low dispersed-phase volume fractions (<10%), as higher fractions require higher number of tracked particles, which results in simulations becoming very expensive. Substantial contribution to the multi-phase modelling was made by Rusche in his work on Euler-Euler method, which provided basis for modelling of systems without a limit on phase fraction, thus making it the most general method (Rusche, 2002).

In the Euler-Euler method, both phases are treated as interpenetrating continua. The volume fraction α defined as a fraction of volume of a cell occupied by the given phase is introduced and a separate set of transport equations is solved for each phase. For a phase j, the corresponding Navier-Stokes equation becomes:

$$\frac{\partial \rho_j \alpha_j \vec{U}_j}{\partial t} + \left(\rho_j \alpha_j \vec{U}_j \cdot \nabla\right) \vec{U}_j = -\alpha_j \nabla p + \nabla \cdot \left(\eta_j \alpha_j \nabla(\vec{U}_j)\right) + \rho_j \alpha_j \vec{g} + \sum \vec{F}_{i.j} \tag{2.23}$$

However, as seen in the Equation 2.23, the method requires a number of closure models $\vec{F_{i.j}}$ to account for effects such as drag, lift, virtual mass, bubble breakage etc. Therefore, Euler-Euler method strongly depends on availability of empirical data for the closure models and might become largely unstable in their absence or insufficient precision.

2.4.2. Volume of Fluid (VOF)

VOF treats phases as segregated, capturing position of their interface. An additional source term $\vec{F_\sigma}$ is added to the Navier-Stokes equation, accounting for the interfacial force. Scalar field α_I describes position of the interface and its transport equation is given by:

$$\frac{\partial \alpha_I}{\partial t} + \vec{U} \cdot \nabla \alpha_I = D_S \left|\nabla \alpha_I\right| \tag{2.24}$$

D_S should be understood as diffusive speed of propagation of the interface. Consequently, the Equation 2.24 is often simplified by introducing interface velocity:

$$\vec{U}_T = \vec{U} + D_S \hat{n} \tag{2.25}$$

where:

$$\hat{n} = \frac{\nabla \alpha_I}{|\nabla \alpha_I|} \tag{2.26}$$

Hence, the Equation 2.24 becomes:

$$\frac{\partial \alpha_I}{\partial t} + \vec{U}_T \cdot \nabla \alpha_I = 0 \tag{2.27}$$

The interface capturing can be performed using method described by Jasak & Weller (1998).

A single Navier-Stokes equation is solved in the form:

$$\frac{\rho\vec{U}}{\partial t} + \nabla\cdot\left(\rho\vec{U}\vec{U}\right) = -\nabla p + \nabla\cdot\left(\eta\nabla(\vec{U}) + \nabla(\vec{U})^T\right) + \rho\vec{g} + \sum\vec{F} \tag{2.28}$$

On one hand, an advantage of VOF is that it most of the times yields accurate results using only a single closure model F_σ for surface tension force. A drawback, on the other hand, is that it requires high mesh resolution to achieve the desired accuracy. As pointed out by Andersson *et al.* (2012), it would be possible to employ the method alone for dispersed bubbly flows, however, their diameter would need to be more than one order of magnitude larger than the cell-size. Therefore, it is mainly used for phases with large interface, where it is superior to Euler-Euler both in terms of precision and stability (Andersson *et al.* , 2012).

2.4.3. Closure Models

As explained in this chapter, due to various adopted simplifications, closure models are required to compensate for the introduced error. Concerning VOF and Euler-Euler, there are mainly 3 closure models that are considered. These are *drag*, *virtual mass* and *surface tension*.

2.4.3.1. Drag

Most of drag models utilise a form:

$$\vec{F}_D = C_D f(\vec{U}) \tag{2.29}$$

For multi-phase Euler-Euler simulations, this model is given as (Xu *et al.* , 2014):

$$\vec{F}_{D.k} = \alpha_c\alpha_d K((U)_d \overset{\rightharpoonup}{-}\ \vec{U_c}) \tag{2.30}$$

$$K = \frac{3}{4}\rho_c C_D \frac{\left|\vec{U}_d - \vec{U}_c\right|}{d_d} \tag{2.31}$$

with d_d corresponding to characteristic diameter of the dispersed phase and subscript c to continuous phase. Drag coefficient C_D is commonly evaluated via Schiller and Naumann correlation (Schiller & Naumann, 1935):

$$C_D = \begin{cases} \frac{24(1+0.15Re^0.683)}{Re}, & Re \leq 1000 \\ 0.44, & Re > 1000 \end{cases} \tag{2.32}$$

where Re is defined as:

$$Re = \frac{\left|\vec{U}_d - \vec{U}_c\right| d_d}{\nu_c} \tag{2.33}$$

2.4.3.2. Virtual Mass

Virtual mass is an artificial concept that accounts for inertia added to the continuous phase because of the dispersed phase's acceleration through its volume. It is estimated by (Selma *et al.* , 2010):

$$\vec{F}_{vm} = \alpha_c C_{vm} \rho_c \left(\frac{D_c \vec{U}_c}{Dt} - \frac{D_d \vec{U}_d}{Dt} \right) \tag{2.34}$$

with $\frac{D}{Dt}$ being *substantial derivative* and C_{vm} a *virtual mass constant*.

2.4.3.3. Surface Tension

Surface tension on the interface of two immiscible liquids is modelled by equation adapted from Brackbill *et al.* (1992):

$$\vec{F}_S = \sigma \kappa \nabla \alpha \tag{2.35}$$

using surface curvature κ:

$$\kappa = -\nabla \cdot \left(\frac{\nabla \alpha}{|\nabla \alpha_d|} \right) \tag{2.36}$$

2.5. Solving Algorithm

The underlying transport equations together with turbulence models and closure models make up a set of equations describing the flow field. This set is solved using an appropriate solving algorithm. In OpenFOAM framework, which is used in this work, the most common one is denoted `PIMPLE` (acronym combined from two different algorithms: Pressure-Implicit with Splitting of Operators (PISO) and Semi-Implicit Method for Pressure-Linked Equations (SIMPLE)). Its principle is described in Holzmann (2016) as follows: At any time step, the solver searches for a steady-state solution within its outer loop. It also has an inner correction loop which iterativly solves for explicit parts of the equation until they have converged. Once the error falls below specified tolerance, the solver exits its outer loop and proceeds to the next time step (Holzmann, 2016).

3. Solution Methodology

This section details the solution procedure, in which the simulations were set up. In addition, it provides reasoning for choices and selections undertaken in the process.

3.1. Choice of the CFD Tool

Currently, there are several commercially available CFD tools. Most of these provide reasonably intuitive and easy-to-use graphical interface, supplemented by customer support. Unfortunately, most of them are relatively expensive and do not offer the possibility to modify individual functions or edit the source code. Therefore, OpenFOAM was selected to conduct these simulations, as it is open-source and enables direct insight into the individual functions employed.

Simulations were set up and computed using OpenFOAM versions v1806 (from `www.openfoam.com`) and version 6 (from `www.openfoam.org`). Subsequent post-processing was conducted using ParaView 5.5.2.

3.1.1. OpenFOAM Description

OpenFOAM is a framework coded in `C++` and first released in 2004. Unlike most of the codes that were then generated in `Fortran`, using object-oriented programming of `C++` facilitated not only overall code reduction, but also straight-forward customization. OpenFOAM is primarily intended for operation on Linux systems and is controlled via the Linux Terminal. The original authors were Henry Weller, Hrvoje Jasak, Mattijs Janssens, Andrew Heather, Mark Olesen and Sergio Ferraris, then being part of Imperial College London.

3.2. General Task Description

The objective of the simulation is to capture behaviour of an internal MLR as depicted in the set-up A in Figure 1.1. To gain preliminary understanding of the MLR operation, effect of varied initial filling height and air inlet flowrate were studied on three different geometries, which differed most significantly in their ratio of riser to downcomer cross-sectional areas A_R/A_D. Since this parametric study is required both for the initial transient and subsequent steady-state, the simulations of initial several seconds need to be captured. Thus, to make this study feasible in terms of computational resources, all simulations are performed on a 2D axisymmetric wedge-shaped domain, representing a section of a cylindrical reactor with a dome-shaped bottom and of a straight-forward internal design.

The reaction liquid is assumed to be of the same properties as water and the extraction phase is approximated as n-Decanol. Air is introduced at the bottom part of the riser by two ring-shaped spargers. A third ring-shaped sparger introduces n-Decanol at the bottom of the downcomer. The operation is further assumed to be isothermal at 30 $^{\circ}C$, a requirement given by presence of thermally sensitive biological material inside the reactor. The reactor operates at ambient pressure and has an orifice open to the atmosphere at the top. Finally, the effects of the internal reaction or of the heat transfer are neglected, and the study is aimed exclusively at the hydrodynamics of the reactor.

3.3. Geometry and Pre-processing

Geometries for this work in .stl format were created by Maximilian von Campenhausen. They represented a plan-parallel section of the internal MLR and were provided in three design modifications intended for study of influence of the A_R/A_D ratio on overall hydrodynamics. Based on their inner radius, they are referred to as R10, R15 and R22 (Figure 3.1).

For meshing purposes, the front and back faces were removed and the geometry was split into a set of .stl files, each representing an individual component of the geometry (three for the different ring spargers represented as ring pipes, inner separation wall, outer wall, top wall and the inner face serving as the axis of symmetry). For the depiction, please refer to the Figure 3.2. This splitting was done for easier subsequent manipulation, as each of these files is then transferred into an individual patch of the mesh. Finally, the .stl files were merged by calling:

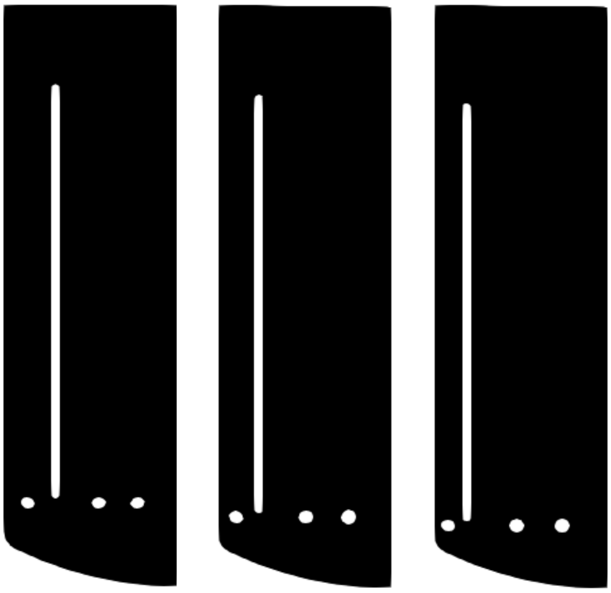


Figure 3.1.: Three examined geometries: R10, R15 and R22

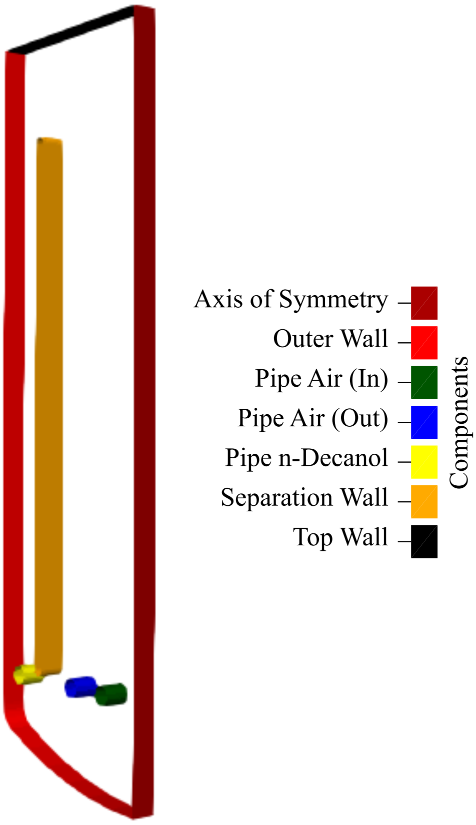


Figure 3.2.: Depiction of individual components of the geometry

```
$ cat *.stl > reactor.stl
```

which creates an output file `reactor.stl` containing the geometry. A sample from this file is shown in Table 3.1. There it can be noted, that every element of the geometry is referred to as *solid* and is described by a set of triangles, whereby each triangle is defined by its normal and three verteces.

Table 3.1.: Sample of `reactor.stl`

```
solid outer
   facet normal 9.903855e-01 1.383346e-01 -0.000000e+00
      outer loop
         vertex 8.961727e-02 1.639853e-01 0.000000e+00
         vertex 8.961727e-02 1.639853e-01 1.000000e-02
         vertex 9.000000e-02 1.612452e-01 0.000000e+00
      endloop
   endfacet
   facet normal 9.903855e-01 1.383346e-01 0.000000e+00
.
.
.
endsolid
solid top
.
.
.
endsolid
```

3.4. Mesh Generation

Mesh was generated using cfMesh toolbox contained in foam-extend platform. The toolbox was compiled and added to the OpenFOAM bash. Before the mesh generation, the geometry was pre-processed by calling:

```
$ surfaceFeatureEdges -angle 40 reactor.stl reactor.fms
```

This creates a new file `reactor.fms`, which enables more precise manipulation. Subsequently, meshing dictionary `meshDict` was constructed with parameters as listed in Table 3.2.

Table 3.2.: meshDict

```
maxCellSize 0.001; //sets maximum cell size to 1 mm

surfaceFile "reactor.fms"; //selects the geometry file for meshing

boundaryLayers              //sets boundary layers
{
    nLayers         4;
    thicknessRatio  1.2;
    patchBoundaryLayers
    {
        "inner"
        {
            nLayers           0; //no boundary layers at the axis of symmetry
        }
    }
   optimiseLayer 1;             //enables layer-optimiser
}
renameBoundary
{
    defaultName wedge;          //creates a joint patch "wedge"
    defaultType empty;
}
```

The cell size, number of boundary layers and their thickness ratio were chosen to achieve value of non-dimensional wall distance $y^+ < 5$. With this condition fulfilled, no further wall treatment by means of wall functions needs to be implemented in the later turbulence modelling. The preliminary estimation of the cell size is conducted via the following set of empirical correlations:

$$c_f = [2log_{10}(Re) - 6.5]^{-2.3} \tag{3.1}$$

$$\tau_w = \frac{1}{2}\rho U^2 c_f \tag{3.2}$$

$$u_T = \sqrt{\frac{\tau_w}{\rho}} \tag{3.3}$$

$$y_C = \frac{y^+ \nu}{u_T} \tag{3.4}$$

This method is based on the fully-turbulent flow assumption and utilizes friction coefficient c_f, wall shear stress τ_w and dimensionless speed u_T. After plugging in the desirable value of y^+, the system outputs value of the first cell-centre height, which is the half of the first-layer width. Since the thickness ratio is normally selected in the range of 1 to 1.5, its value in this case is adjusted accordingly. Upon completion of the full simulation, the actual value of y^+ can be obtained in postprocessing by calling:

```
$ mySolver  -postProcess -func yPlus
```

In this command, `mySolver` is replaced by the actual name of the used solver.

Finally, meshing of the so-called empty faces needs to be addressed. Since in OpenFoam, all geometries are 3D by default, the direction in which the domain should not be solved is only set to a single-cell thickness and the cell faces in this direction are set to empty. As this direction will need to be deformed to obtain a wedge (axisymmetric) domain, it is necessary to merge the two empty patches into one. The arising patch is labeled as `wedge` in the `meshDict`
.

Before running the meshing, it is important to set the following entry in the `controlDict`:

```
writeFormat     binary;
```

It is also possible to write outputs in *ascii* format, however, a bug in a later-used toolbox causes deviation of face normals from the average patch normal, if this format is selected.

The mesh can now be generated calling:

```
$ cartesian2DMesh
```

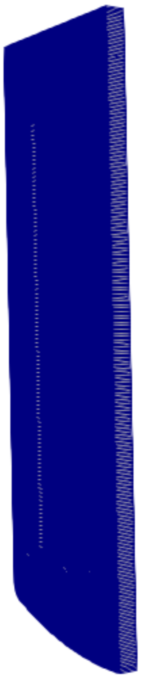


Figure 3.3.: Resulting parallel mesh

After the task is performed, a directory *polyMesh* is placed in the *constant* directory, containing the mesh.

Before the mesh is deformed to the axisymmetric shape, four conditions need to be fulfilled (`openfoamwiki.net`):

1. The desired axis of symmetry is a straight line parallel to the xy-plane
2. The desired empty faces (parallel to the xy-plane) are contained in a single patch
3. The mesh is 2D (i.e. z direction is only 1-cell thick)
4. The mesh does not cross $y = 0$

While the first condition was fulfilled in the geometry-splitting, the second and third condition were met by the set-up of the `meshDict` and selection of the solver. Nevertheless, since the domain crosses $y = 0$, it needs to be translated by a given vector in the positive y-direction. This is achieved by the command:

```
$ transformPoint -translate "(0 0.5 0)"
```

After the requirements have been satisfied, the package makeAxialMesh can be compiled from openfoamwiki.net. The execution is performed by:

```
$ makeAxialMesh -case . -axis inner -wedge wedge -overwrite
```

The three arguments set the execution in the current dictionary, designate the axis of symmetry to be the inner patch and designate the patch of empty faces wedge to be split into the two wedge-bounding patches. The code deforms cells in the way the wedge patches (by default re-labeled to wedge_pos and wedge_neg) acute angle of 5. The flag -overwrite causes the entries in the directory *polyMesh* are overwritten.

Subsequently, the mesh needs to be cleared, as there are still small faces left in the inner patch which need to be removed. For the purpose, a collapseDict is set up with following parameters: So after:

Table 3.3.: collapseDict

```
collapseEdgesCoeffs
{
    minimumEdgeLength   1e-5;
    maximumMergeAngle   179;
}
```

```
$ collapseEdges -overwrite
```

the edges falling below the specified values are collapsed. The exact values on the length and angle set in the dictionary are not that important as long as after the execution, the patches inner and wedge contain 0 faces. This can be checked by opening:

```
$ vi constant/polyMesh/boundary
```

Lastly, inlets and outlets need to be created on the existing boundaries. This is done in two steps. Firstly, corresponding sets of the boundary faces is cropped out and defined into the faceZones. Subsequently, new patches are defined from these sets. The first step is aided by topoSetDict. This is set up using following logic:

The cycle above is repeated with altering entries given in Table 3.5:

The dictionary is executed calling:

```
$ topoSet
```

Table 3.4.: topoSetDict

```
actions
(
    {
        name    temporary;  //resets the dummy faceSet
        type    faceSet;
        action  clear;
    }
    {
        name    temporary;  //Takes the boundary, on which the new
        type    faceSet;    //inlet/outlet should be created
        action  new;        //and stores it in the dummy "temporary".
        source  patchToFace;
        sourceInfo
        {
            name   top;     //the boundary to be cropped
        }
    }
    {
        name temporary;     //Crops out the faces on the given boundary
        type faceSet;       //that fall into the defined box.
        action subset;      //Stores the cropped faces in the dummy
        source boxToFace;   //faceSet "temporary".
        sourceInfo
        {
            box (0.07 0.38 0.001) (0.08 0.39 0.09);
        }   //the box is defined by inputting coordinates of 2 extreme points
    }
    {
        name    outletAir;  //declares the new set
        type    faceZoneSet;//stores the cropped-out faces
        action  new;
        source  setToFaceZone;
        sourceInfo
        {
            faceSet temporary;
        }
    }
```

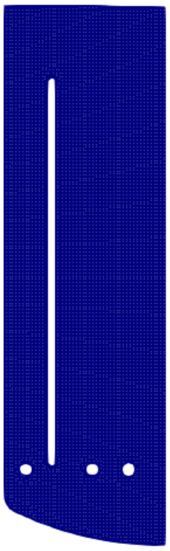


Figure 3.4.: Wedge-shaped mesh from y-x view

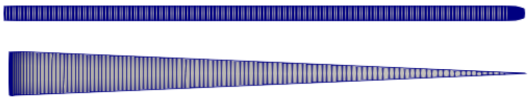


Figure 3.5.: Wedge-shaped mesh from z-y and z-x views

Finally, the new patches need to be constructed from the created sets. `createPatchDict` is employed:

The function is called:

```
$ createPatch -overwrite
```

This in the same time removes the empty patches. Finally, for all walls, the corresponding entries in the `constant/polyMesh/boundary` need to be altered as: This re-definition is helpful for further reference. Thuse, the mesh generation is completed. The mesh now has 3 inlets (2 for air and 1 for n-Decanol) and 2 outlets (1 for air at the top and 1 for liquids on the side).

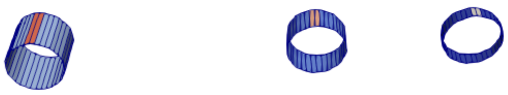


Figure 3.6.: Inlets

Table 3.5.: Changing entries of the `topoSetDict`

Cropped Boundary	New Boundary	Bounding Box
top	outletAir	(0.07 0.38 0.00) (0.08 0.39 0.09)
outer	outletDec	(0.089 0.41 0.00) (0.091 0.42 0.09)
pipeAirOut	inletAirOut	(0.0402 0.642 0.00) (0.0412 0.644 0.09)
pipeAirIn	inletAirIn	(0.01950 0.642 0.00) (0.02087 0.644 0.09)
pipeDec	inletDec	(0.07755 0.642 0.00) (0.07855 0.644 0.09)

(a) Outlet on the side riser

(b) Outlet on the top

Figure 3.7.: Outlets

3.5. Choice of Solver

As discussed in the previous chapter, the choice of the solver is made based on the parameters such as number of phases, segregation/dispersion, volumetric fractions of the dispersed phases, availability of empirical data or stability of the solver. The complication that arises in this simulation is depicted in Figure 3.8. Since the system consists of regions of both, segregated and dispersed phase, application of both VOF and Euler-Euler approaches is rather complicated. On one hand, applying Euler-Euler would be hindered by the fact that multitude of closure models would be required, so as to construct a stable set-up. Preliminary simulations applying Euler-Euler approach showed, the solution would diverge in situations where a

Table 3.6.: createPatchDict

```
pointSync false;
patches
(
    {
        name outletAir;     //the same cycle is repeated for "outletDec"
        patchInfo
        {
            type inletOutlet;
            inGroups (inletOutlet);
        }
        constructFrom set;
        set outletAir;
    }
    {
        name inletAirIn;    //the same cycle is repeated for
        patchInfo           //"inletAirOut" and "inletDec"
        {
            type patch;
        }
        constructFrom set;
        set inletAirIn;
    }
```

Table 3.7.: Changed entries in constant/polyMesh/boundary

```
        type        wall;
        inGroups    1(wall);
```

stronger mechanical wave was created, when a larger bubble or churn reached the top interface between water and air, or in case the surface of water reached an edge of a wall (either outer or separation). A work-around for these issues is possible as there are closure models for phenomena such as waves, coalescence or bubble breakage, however their implementation would be laborious and prone to errors due to possible inaccuracy of the experimental data. Albeit solution by VOF would be more robust, it would require a particularly fine mesh to deliver reliable results. This gave rise to an idea for coupling of the two approaches. This idea was pioneered by Cerne *et al.* (2001) in a work where implementation of either VOF or Euler-Euler was controlled by switching factor C_α, value of which was dynamic both in space and time. Applying this approach results in uneven number of equations solved for different parts of the domain, and therefore is not very practical. Hence, subsequent research efforts focused on coupling in a way where VOF is applied on the top of Euler-Euler, provided C_α flags the given region as sufficiently resolved for VOF. Nevertheless, the switching routine assigning the value of C_α remains problematic (Wardle & Weller, 2013).

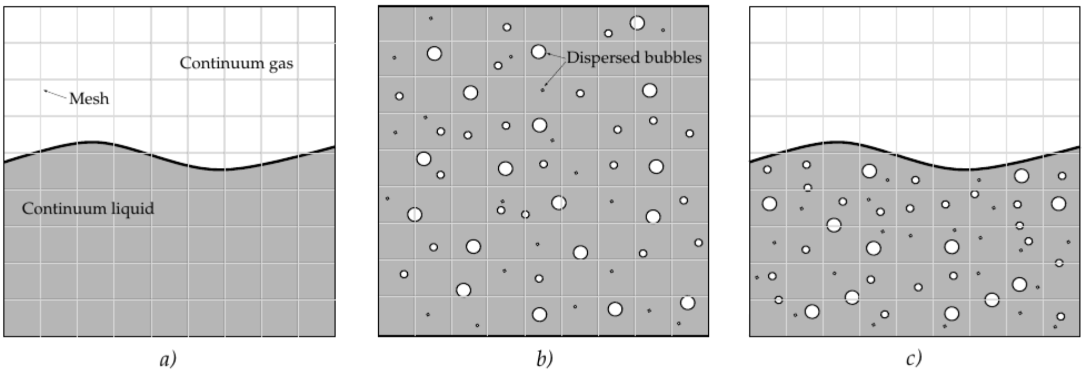


Figure 3.8.: Various two-phase regimes: segregated, dispersed and their combination (Tocci, 2016)

More recently, switching routines based on mesh resolution and interface curvature (Anez *et al.* , 2016), or critical Weber diameter (Shonibare & Wardle, 2015) were proposed. Shortcomings of these routines are in their generality and applicability, which have not yet been verified. Similar coupling is used in an OpenFOAM `multiphaseEulerFoam` solver devised by Welller based on his previous work (Weller, 2008). `multiphaseEulerFoam` is based on Euler-Euler approach, with coupled VOF applied on the top. C_α has a function of a binary switch, which enables or disables the interface sharpening. Initial term to the transport equation of α of a phase j is introduced:

$$\frac{\partial \alpha_j}{\partial t} + \vec{u}_j \cdot \nabla \alpha_j + \nabla \cdot (\vec{u}_c \alpha_k (1 - \alpha_j)) \tag{3.5}$$

Defining compression velocity $\vec{u}_c$ as:

$$\vec{u}_c = C_\alpha \left| \vec{u} \right| \frac{\nabla \alpha_j}{\left| \nabla \alpha_j \right|} \tag{3.6}$$

It should be noted that compression here refers to interface compression, i.e. numerical sharpening of the interface. `multiphseEulerFoam` treats C_α as a constant independent of time and space, which is specified by the user for a given phase-pair and applied uniformly throughout the domain. This approach raises a question on the level of error introduced by application of the interface sharpening even in the parts of the grid, where meshing would be insufficient for pure VOF. Tocci (2016) investigates this issue concluding this hybrid approach provides better description of experimental data than other either VOF or Euler-Euler when applied alone. In addition, the solver was also proven to provide accurate results even in cases when mesh was coarser than the predicted bubble diameter (Hlawitschka *et al.* , 2016). Finally, Wardle & Weller (2013) applies this hybrid approach on various levels of refinement and also vindicates its accurate performance on mesh refinement coarser than required for VOF. Thus, this hybrid solver was selected for this work and the simulations where conducted on mesh above the VOF refinement.

3.5.1. `multiphaseEulerFoam`

`multiphaseEulerFoam` is a solver for a system of many compressible fluid phases including heat-transfer (Greenshields, 2019). The main structure of the solver is depicted in Table 3.8.

The main framework of the solver is made up by three loops. The main loop is executed until the run time reaches the user-specified value. The inner loop operates PIMPLE solver which iterates the governing equation until the residual falls below pre-specified limit. Within every iteration, the corrector loop is operated. Application of closure models for drag, surface tension and virtual mass is embedded in the solver. Further closure models (such as lift or coalesence) might be added, but are not considered in this work.

Table 3.8.: multiphaseEulerFoam

```
while (runTime.run())
   {
       #include "readTimeControls.H"
       #include "CourantNo.H"
       #include "setDeltaT.H"
       runTime++;
       Info<< "Time = " << runTime.timeName() << nl << endl;
       // --- Pressure-velocity PIMPLE corrector loop
       while (pimple.loop())
       {
           turbulence->correct();
           fluid.solve();
           rho = fluid.rho();
           #include "zonePhaseVolumes.H"
           #include "UEqns.H"
           // --- Pressure corrector loop
           while (pimple.correct())
           {
               #include "pEqn.H"
           }
           #include "DDtU.H"
       }
       runTime.write();
       runTime.printExecutionTime(Info);
   }
```

The solver requires inputs of mesh, boundary conditions, applied closure models and operation parameters. These are supplied by dictionaries in directories `0`, `constant` and `system` and their assembly for this problem is described in following sections of this chapter.

3.6. 0

Directory `0` contains initial boundary conditions. Flow properties that need to be initialized here are α and U of each phase, p_rgh (dynamic pressure) and turbulence parameters (discussed in later section). α and U need to be defined separately for each phase. Hence, 7 dictionaries are required at this folder, excluding turbulence dictionaries, where the boundary conditions are defined for each boundary of the mesh.

3.6.1. α

At the start of the simulation, every cell needs to have a value of α assigned to it and sum of α of all components at each cell must be equal to *unity*. As there are many cells in the domain, direct assignment is only performed for the boundaries and `setFields` utility is used to assign values to internal cells. Three dictionaries are needed with nomenclature `alpha.*`, with `*` replaced by the phase names (`air`, `water` and `decanol`). Their structure is identical, hence only α`.air` is provided:

Assignments to the three groups (wall, inletOutlet and wedge) specified in the `polymesh/boundary` are made at the same time and do not need to be made individually. The entries that differ for each phase are inlet values (decanol has value of 1 at its inlet and 0 on the other two, water is 0 on all three) and phi, where the suffix of the phi entry is adjusted for the corresponding phase. This assigns corresponding flux to the patch.

The condition *wedge* is a generic condition that can be used for any quantity, indicating empty direction of an axially symmetric domain. *zeroGradient* is a special type of von Neumann boundary condition, indication the gradient of the quantity is *zero*. *inletOutlet* is again a generic OpenFOAM condition for outlets, whereby the flow can enter or leave the domain. This is in principle combination of other two generic conditions: *fixedValue* (here specified as inletValue, applied for the inlet direction) and zeroGradient (applied for the outlet direction).

Table 3.9.: alpha.*

```
dimensions      [0 0 0 0 0 0 0]; //SI dimensions of the quantity
internalField   uniform 0;
boundaryField
{
    wall
    {
        type            zeroGradient;
    }
    wedge
    {
        type            wedge;
    }
    inletOutlet
    {
        type            inletOutlet;
        phi             phi.air;
        inletValue      $internalField; //will be assigned by setFields
        value           $internalField;
    }
    inletDec
    {
        type            fixedValue;
        value           uniform 0;
    }
    inletAirIn
    {
        type            fixedValue;
        value           uniform 1;
    }
    inletAirOut
    {

        type            fixedValue;
        value           uniform 1;
    }
}
```

Generally, unless the entry following *value* refers to a fixed-value condition, its numerical value only serves the purpose of a place-holder and is overwritten by *setFields* assignment.

Making this assignment for the cells of the domain is performed subsequently. A *setFieldsDict* is constructed as:

Table 3.10.: `setFieldsDict`

```
defaultFieldValues
(
    volScalarFieldValue alpha.air 0
    volScalarFieldValue alpha.water 1
);
regions
(
    boxToCell
    {
        box (0 0.38 0.001) (0.09 0.430 0.009);
        fieldValues
        (
            volScalarFieldValue alpha.air 1
            volScalarFieldValue alpha.water 0
        );
    }
);
```

The set-up *setFieldsDict* assigns values of α to 0 for air and 1 for water, except for the region of the box, which is in the textitsetFieldsDict specified by its lower-most and upper-most points. The coordinates are selected in a way to make the top part of the reactor be filled by air and bottom by water. By varying the y-cooridnate of the uppermost point, the initial filling height is adjusted (this is given by the fact the domain is flipped in the y direction). The assignment is then executed by running:

```
$ setFields
```

3.6.2. `U` and `p_rgh`

Initial conditions for velocity need to be specified for each component independently. Conditions of velocity are coupled with conditions of pressure, i.e. their selection is mutually dependent. It follows, that a fixed value can only be specified for one of the quantities at each boundary. The velocity will therefore be fixed at inlets, where its value will be given by desired inlet flowrate of air and n-Decanol, and at the walls where the velocity is *zero* due to *no-slip boundary condition*. On the other hand, pressure needs to be specified at the outlet, where its value is given by the ambient conditions. Thus, `U.*` are constructed as:

This form would be applied for air and it follows that the difference between various `U.*` dictionaries will be in the inlet values. Condition type `pressureInletOutletVelocity` is a generic outlet condition, whereby the value of velocity is computed from the pressure field.

To maintain physical behaviour, `fixedFluxPressure` boundary condition type is applied at all boundaries, where the value of velocity is fixed. On the other hand, value of pressure needs to be given at the outlet, using `totalPressure` condition and specifying ambient pressure p_0. Thus, `p_rgh.*` is constructed as Table 3.12:

It should be noted that again here, the entries of `value` at `fixedFluxPressure` only serves as a place-holder and exact value at each face is computed by the solver.

3.7. constant

The `constant` directory contains the directory `polyMesh`, where the mesh is stored, and dictionaries `g`, `transportProperties` and `turbulenceProperties`, which sets the turbulence model and is discussed separately. Dictionary `g` simply sets value of gravitational acceleration with the entries:

```
dimensions      [0 1 -2 0 0 0 0];
value           (0 9.81 0);
```

These specify its SI dimensions and vector value. Note, this is again positive due to the orientation of the domain.

The `transportProperties` dictionary contains thermo-physical properties of each component and entries for closure models. Given the operation is assumed isothermal at 30 $°C$, the required inputs are kinematic viscosity (entry *nu*), density (entry *rho*), and particle diameter model and corresponding coefficients.

Table 3.11.: U.*

```
dimensions      [0 1 -1 0 0 0 0];
internalField   uniform (0 0 0);
boundaryField
{
    wall
    {
        type        fixedValue;
        value       uniform (0 0 0);
    }
    wedge
    {
        type            wedge;
    }
    inletOutlet
    {
        type        pressureInletOutletVelocity;
        value       uniform (0 0 0);
    }
    inletDec
    {
        type        noSlip;
    }
    inletAirIn
    {
        type        fixedValue;
        value       uniform (0 -0.27 0);
    }
    inletAirOut
    {
        type        fixedValue;
        value       uniform (0 -0.13 0);
    }
}
```

Table 3.12.: p_rgh

```
dimensions      [1 -1 -2 0 0 0 0];
internalField   uniform 1e5;
boundaryField
{
    wall
    {
        type        fixedFluxPressure;
        value       uniform 1e5;
    }
    wedge
    {
        type            wedge;
    }
    inletOutlet
    {
        type        totalPressure;
        p0          uniform 1e5;
    }
    inletDec
    {
        type        fixedFluxPressure;
        value       uniform 1e5;
    }
    inletAirIn
    {
        type        fixedFluxPressure;
        value       uniform 1e5;
    }
    inletAirOut
    {
        type        fixedFluxPressure;
        value       uniform 1e5;
    }
}
```

Table 3.13.: Thermophysical properties of present components

Component	Entry	Value	Source
decanol	rho	823	(Matsuo. & Makita, 1989)
	nu	1.1e-5	(Li & Lu., 2001)
	diameterModel	constant	provided input
	constantCoeffs	d 7e-4	
water	rho	996	`engineeringtoolbox.com`
	nu	8e-7	
	diameterModel	constant	(Rusche, 2002)
	constantCoeffs	d 1e-4	
air	rho	1.15	`engineeringtoolbox.com`
	nu	1.6e-5	
	diameterModel	isothermal	(Rusche, 2002)
	constantCoeffs	d0 1e-4; p0 1e5	

An Euler-Euler method further requires a definition of a drag model. This was implemented in the same way as the tutorial case bubbleColumn (based on Rusche (2002)). The implementation is as follows Table 3.14:

SchillerNaumann is the only model synchronised with `multiphaseEulerFoam` applicable to fluid mixtures. The models are blended, meaning the drag of the phase-pair is calculated based on corresponding volume fractions. The same drag model is applied for the other two pairs and entries are always ordered as (dispersed continuous).

Furthermore, Euler-Euler also applies virtual mass models, which are aided by constants C_{vm}. Since the model of virtual mass is somewhat artificial, there is a significant level of disagreement on what value is appropriate to be used. Bannari *et al.* (2008) give an overview of several authors and their approach to this concept. While in some works it is disregarded (Krishna

Table 3.14.: transportProperties

```
drag
(
    (air water)
    {
        type blended;
        air
        {
            type SchillerNaumann;
            residualPhaseFraction 0;
            residualSlip 0;
        }
        water
        {
            type SchillerNaumann;
            residualPhaseFraction 0;
            residualSlip 0;
        }
        residualPhaseFraction 1e-3;
        residualSlip 1e-3;
    }
```

et al. , 2000), this value was often thought to be 0.5 for phase-pair of any two components ((Darmana *et al.* , 2005), (Buwa *et al.* , 2006)). However, Pudasaini (2019) dispute this approach suggesting C_{vm} only approaches 0.5 for low volume fractions. Instead, it approaches 0 as the volume fraction of the dispersed phase approaches *unity*. This is supported by Tomiyama (2004), where a model to estimate C_{vm} based on *Eotvos number (Eo)* is developed. This approach was also supported by preliminary simulations in this work, where C_{vm} of 0.5 for air-water phase-pair resulted in an un-physical (very low or even negative) pressures. Thus, the values were set as follows:

Table 3.15.: Assignment of `virtual mass in transportProperties`

```
virtualMass
(
    (air water)     0.1
    (air decanol)   0.1
    (decanol water) 0.5
);
```

Lastly, the interface compression (C_{α}) needs to be set. This was turned on for air-water interface for reasons described in Section 3.5, so as to improve stability of the solver. Additionally, it was also applied for the air-decanol interface, as the two fluids were only expected to get in contact at the top of the column on a segregated interface. Consequently, the entries are in Table 3.16:

Table 3.16.: Assignment of interface compression in `transportProperties`

```
interfaceCompression
(
    (air water)     1
    (air decanol)   1
    (decanol water) 0
);
```

Since the VOF based approach operates with interfacial forces, tension values need to be provided.

Table 3.17.: Interfacial tension at the present phase-pairs at 30 °C

Phase-pair	Value kg/s^2	Source
(air water)	0.071	surface tension.de
(air decanol)	0.028	surface tension.de
(decanol water)	0.008	(Motomura, 1991)

3.8. Turbulence Model

The only applicable turbulence model for this simulation can be carried out by RANS method, as LES could result in non-physical behaviour on 2D grid (Andersson *et al.* , 2012). Albeit Wardle & Weller (2013) successfully applied Smagorinsky LES model in the 2D case, it is generally considered not to be suitable, besides requiring mesh refined down to Kolmogorov length-scale.

As for RANS methods, realizable $k - \epsilon$ is preferred over standard $k - \epsilon$, mostly in cases of multi-phase flows and on axially symmetric domains (Andersson *et al.* , 2012). There are even more suitable models able to capture bubble-induced turbulance such as Lahey $k - \epsilon$ (Lahey *et al.* , 1994), however these are not synchronized with `multiphaseEulerFoam` and would therefore require adjustment of the solver. Thus, `turbulenceProperties` is set as:

Table 3.18.: `turbulenceProperties`

```
simulationType RAS;
RAS
{
    RASModel        realizableKE;
    turbulence      on;  //default settings
    printCoeffs     on;
}
```

Turbulence modeling also requires boundary conditions for k, ϵ and ν_t. These are provided by dictionaries `k`, `epsilon` and `nut` in the `0` directory. As discussed in the section 3.4, the level of near-wall refinement enables using von Neumann zero gradient boundary conditions at the wall, instead of applying boundary conditions. Thus, zeroGradient condition is applied in `k`, `epsilon` and `nut` at all boundaries, except for inlets. At the inlet, the condition for `k` is set by:

Table 3.19.: `k` dictionary

```
inletDec
{
    type        turbulentIntensityKineticEnergyInlet;
    intensity   0.05;           // turbulent intensity
    phi         phi.decanol;   // flux corresponding to the inlet
    value       uniform 1e-3; // place-holder
}
```

The value of turbulent intensity used is based on work of Kools (2016) and Weber (2015), as well as on presumption that the in-flow would be at the middle level of turbulent intensity. Subsequently, for `epsilon` at the same boundary:

Table 3.20.: `epsilon` dictionary

```
inletDec
{
    type            turbulentMixingLengthDissipationRateInlet;
    mixingLength    5e-5;
    phi             phi.decanol;
    value           uniform 5;
}
```

The important entry in this boundary is the mixing length calculated using relation for axisymmetric jet as 0.075-times half of its width (Versteeg & Malalasekera, 2007). Lastly, since value of ν_t can be determined from values of k and ϵ, its corresponding boundary condition in `nut` would be:

```
inletDec
{
    type            calculated;
```

```
        value           uniform 0;
    }
```

3.9. system

Besides the already used dictionaries meshDict, collapseDict, topoSetDict, createPatchDict and setFieldsDict, four additional dictionaries are added to aid the solution.

3.9.1. fvSchemes

fvSchemes defines the discretization schemes applied for given terms of the source equations. This dictionary as well as fvSolution was adapted from the bubbleColumn tutorial (based on Rusche (2002)) using already introduced discretization schemes as:

```
ddtSchemes
{
    default         Euler;
}
gradSchemes
{
    default         Gauss linear;
}
divSchemes
{
    "div\(phi,alpha.*\)"    Gauss vanLeer;
    "div\(phir,alpha.*,alpha.*\)"   Gauss vanLeer;
    "div\(alphaPhi.*,U.*\)" Gauss limitedLinearV 1; //Gauss upwind implemented
    div(Rc)                 Gauss linear; //in case of instabilities
    "div\(phi.*,U.*\)"      Gauss limitedLinearV 1; //Gauss upwind;
    "div(phi,epsilon)"      Gauss upwind;
    "div(phi,k)"            Gauss upwind;
}
laplacianSchemes
```

```
{
    default         Gauss linear corrected;
}
interpolationSchemes
{
    default         linear;
}
snGradSchemes
{
    default         corrected;
}
```

3.9.2. fvSolution

Just as fvSchemes, fvSolution is adapted from the bubbleColumn tutorial. PIMPLE solving algorithm is used and solvers GAMG and MULES are employed to solve for pressure and volumetric fraction respectively. Common relaxation factors are used, with values of 0.7 for all quantities asides for pressure, which uses 0.3. Due to a high number of entries (most of which are rather standard or default), the full dictionary fvSolution is not reproduced here.

3.9.3. controlDict

Default entries are used for most of the entries in the controlDict and hence are not detailed in this section. However, the most important parameter that is set here is the maximum *Courant number* and time step as:

```
adjustTimeStep  yes;
maxCo           0.5;
```

Thus, the time step is always adjusted to meet the maximum Co. This value was selected relatively low, as the computation is prone to instabilities.

3.9.4. decomposeParDict

The final dictionary needed in this section is used for parallel computing. In order to obtain results in real time, the domain needs to be split for computation on several cores. Since the mesh is composed of approximately 28,000 cells, the decomposition is performed among 6 cores of RWTH SLURM computer cluster, as higher number of cores was not expected to provide any further benefit in terms of reduced computational time. The dictionary is composed as:

```
numberOfSubdomains 6;
method          simple;
simpleCoeffs
{
    n           (6 1 1);
    delta       0.001; //default
    dataFile    "";
}
distributed     no;
roots           ( );
```

The method `simple` distributes the total number of cells into the given number of neighbouring sub-domains in a way roughly equal split is achieved. The coefficient entry `n` gives a number of sub-domains in each direction (the order of coordinates is $x-y-z$ by default). Decomposition is executed by:

```
$ decomposePar
```

3.10. Simulation parameters

The conditions at which the simulations were conducted are summarised in Table 3.21, with the recirculate rate as the main observed parameter.

Table 3.21.: Simulation Parameters

Decanol Discharge	A_R/A_D	Initial Filling*	Air Discharge
2 L/hr	2.17	84.7%	0.5 vvm
	1.50	81.1%	0.75 vvm
	1.0	77.5%	1 vvm
		74.0%	1.5 vvm
			2 vvm

* differences in filling correspond to 1 cm in filling height

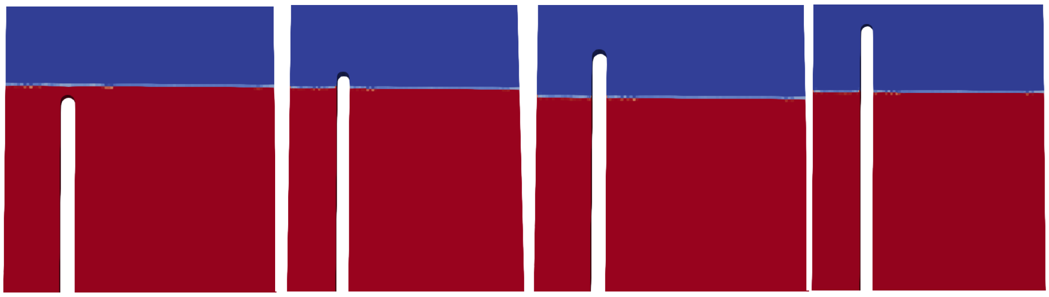


Figure 3.9.: Different filling heights relative to the separation wall

4. Simulation Results

A total of 60 simulations were performed using the specified conditions. Qualitative and quantitative data collected in their post-processing are presented in this chapter with corresponding discussion.

4.1. Simulation Limitations

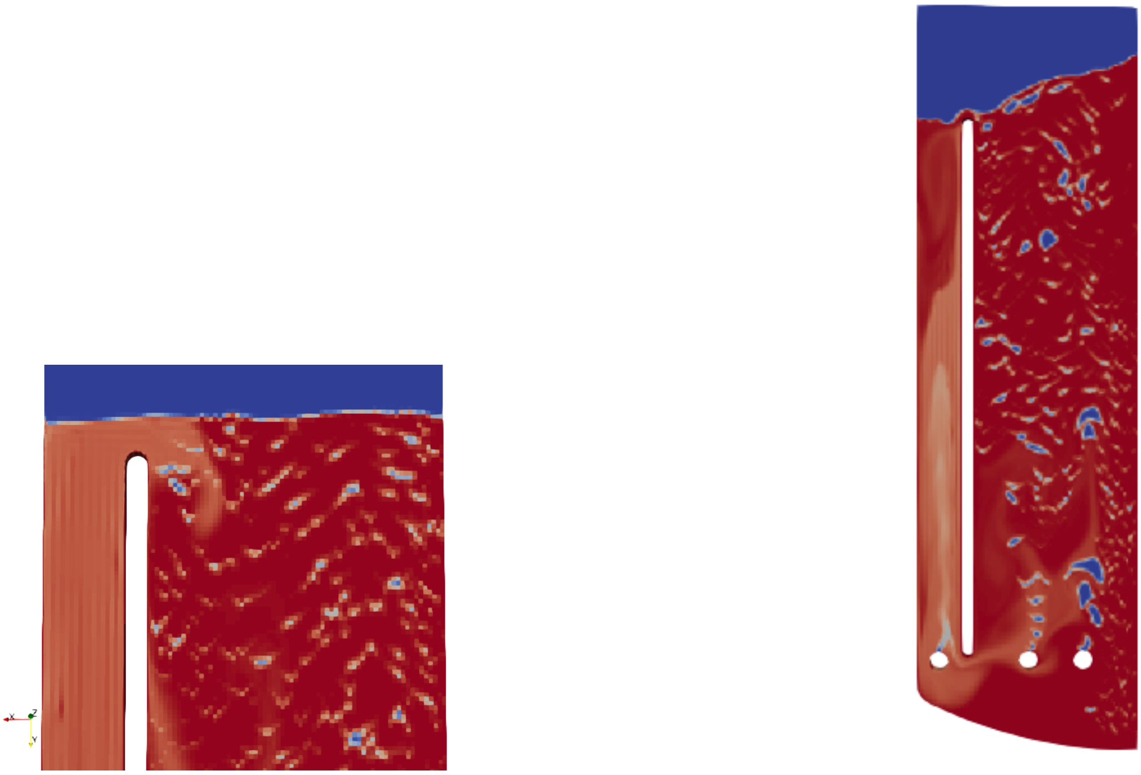
Firstly, limitations of these simulations need to be addressed. These can be broken down to limitations of the simulations as such and limitation based on availability of data for closure models. On the data side, it is deemed that separate investigation needs to be conducted to establish dependence of particle diameter on conditions such as hold-up of extraction phase, recirculation rate or width of downcomer. Value provided as an input to this work was admittedly based on insufficient experimental data and its deviation from reality could have observable impact on the results. Furthermore, an experimental validation for the selection of the virtual masses would help justifying the choice made in their regard.

The main limitations on the simulation side arise from the fact the case was considered as 2D. As a consequence, interactions in the radial direction were not captured, which is of chief influence on the ability to model turbulence. Furthermore, the turbulence model used does not account for bubble-induced turbulence, increasing its deviation with rising Reynolds number. Lastly, only closure models for drag, virtual mass and surface tension were implemented into the simulation, as these were expected to have the largest influence on the hydrodynamics. Nevertheless, an experimental study is required to assess the extent of error introduced by this assumption, as well as of the other simplifications adopted in the simulation process.

4.2. MLR Malfunction

From qualitative data, three different types of malfunctioning were observed. In the first instance, the initial height of the medium was set too low relative to discharge rate of air, and the flow rate of air was insufficient to lift the medium level above the edge of the separation wall.

Secondly, an effect observed exclusively at low air discharge rates, riser and downcomer did get hydrodynamically connected, yet, the lift of air was still insufficient to induce flow in the downcomer section. As a result, the lift of extraction phase dominated and extraction liquid penetrated into the riser.



(a) Extraction phase entering riser

(b) Flooding

Figure 4.1.: Faulty operations

Third observed type of malfunctioning was flooding. In flooding, recirculation flow rate dragged extraction phase towards bottom clearance and further to the riser, causing pollution of the riser by extraction fluid. The effect was directly proportional to the actual height of liquid above the separation wall and to air flow rate. Overall, varying extent of this condition (either partial or total) occurred commonly and further design measures are expected to be necessary to address it in the design phase.

4.3. Effect of Filling Height

Operation was most sensitive to filling height. Overall height has dramatic impact on recirculation rate, and, in theory, operation at any condition of air flow rate and A_R/A_D is feasible, provided height is accordingly maintained. This, however, also poses arguably the most dominant design challenge as perturbation as low as a couple of millimeters is capable of triggering flooding.

Lower and upper limit of filling height were observed. The upper limit is mainly affected by the fact that for given filling height of medium, discharge rate of air would need to be set to unreasonably low values, which would not permit sufficient aeration. Conversely the lower limit was found for values of filling height, which were too low for any tested discharge rate of air to elevate medium over the separation wall. The general observation is summarised in Table 4.1.

Table 4.1.: Feasibility of operation at various fillings

Phase-pair	**General Operation**
84.7%	Flooding
81.1%	Operation feasible
77.5%	Operation feasible
74.0%	Separation wall not overcome

From the observations made on the simulation results, it was noted that the filling level should, for the given geometries, be set below the edge of the separation wall to both circumvent flooding and permit better aeration by means of higher air discharge rate.

4.4. Effect of Air Discharge

Flow rate of air was adjusted in a symmetrical fashion, i.e. such that both spargers discharged equal volume of air. For better description of flow inside the riser, *superficial velocity* U_s was introduced such that:

$$U_s = \frac{\dot{V}_{air}}{A_R} = \frac{\dot{V}_{air}}{\pi \frac{D_R^2}{4}} \tag{4.1}$$

where $\dot{V}_{air}$ is the volumetric flowrate of air and D_R the diameter of the riser. Furthermore, $_\triangle h_0$ was used to describe the difference between the filling height and the edge of the separation wall. Employing these two quantities, it was observed that $U_s/_\triangle h_0$ needed to exceed certain threshold for the liquid to overcome the edge of the separation wall. Once this criterion was fulfilled, recirculation rate $\dot{V}_{rec}$ could be described by the following linear relation:

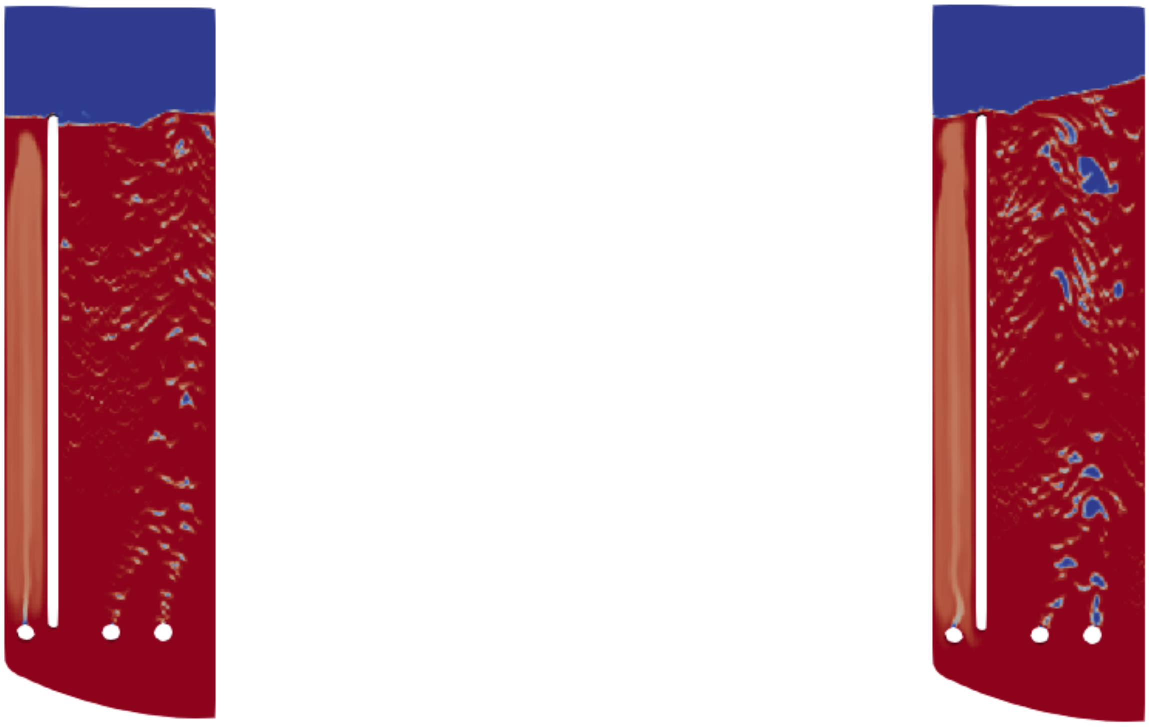
$$\frac{\dot{V}_{rec}}{\dot{V}_{air}} \propto \frac{U_{S.air}}{\Delta h_0} \tag{4.2}$$

Nevertheless, the limit to describing the critical $U_{S.air}/_\triangle h_0$ at which flooding occurs is more complex to capture mathematically. This is mainly due to the fact that flooding is not a binary phenomenon and occurs to different extent at wider range of recirculation rates. Therefore, further investigation into flooding limits would first need to establish permissible flooding extent. This extent could then be correlated to superficial velocity. It is further expected, that the limit of $U_{S.air}/_\triangle h_0$ is dependant on hold-up of extraction phase in the downcomer. Monitoring of these variables and their dependence was, however, beyond the scope of this work.

Additionally, air discharge rate had further effect on size and shape of the bubbles formed in the riser as depicted in Figure 4.2. Generally speaking, higher superficial velocity would shift flow regime towards churn flow, while lower values thereof would favour bubble flow. This observation is in good agreement with Rusche (2002).

4.5. Effect of Riser: Downcomer Area Ratio

In general, the effect of A_R/A_D was found to have a linear effect on the recirulation rate, matching the intuitive anticipation. Nevertheless, this effect is admittedly case-limited, since



(a) Bubble size at 0.5 vvm

(b) Bubble size at 2 vvm

Figure 4.2.: Bubble-size comprarion

different construction of the bottom part of the downcomer can result in varying pressure drop, which could then in turn introduce non-linearity to this dependence.

4.6. Considerations for Reactor-Design

Considering conditions described in Section 4.3, it follows that the successful operation requires efficient removal of extraction phase from the surface of reaction medium, to prevent its accumulation. Build-up of extraction phase would result in rising of the level inside the reactor, which would in turn cause flooding. However, achieving effective removal of the extraction phase is not straightforward and cannot be provided easily by means of an orifice, as mechanical waving is present on the surface of the medium, causing its level to fluctuate.

Circumvention of flooding is of paramount importance, as excessive introduction of extraction phase to the riser would necessitate tripping of the whole process, so as to enable the extraction liquid to settle on the top. It should be noted that in this study, only simplified geometries were studied and the real geometry would probably require implementation of further parts that would aim to increase pressure drop, control the recirculation and thus minimise flooding.

Bounds of the final operation window will be given by penetration of the extraction fluid to the top of the riser on one hand and by flooding on the other. The inlet flow rate of air will need to be set in the way to both enable required residence time and provide sufficient aeration, and the filling height will be selected in the way to maintain the operation set by bounds of the operation window.

4.7. Mathematical Model

A quantitative measure describing recirculation was required in the form of empirical correlation, in which the value of recirculation flowrate could be estimated from the given inlet volumetric flowrate of air, ratio of cross-sectional areas of riser and downcomer, and initial filling height. In order to achieve the required simplicity, linearization of the data was proposed based on observations from Sections 4.4 and 4.5 in the form:

$$\frac{\dot{V}_{rec}}{\dot{V}_{\mathrm{air}}}\frac{A_R}{A_D} = k\left(\frac{\dot{V}_{air}}{{\scriptstyle\Delta} h_0 A_R}\right) \tag{4.3}$$

Employing superficial velocity defined as:

$$U_{S.air} = \frac{\dot{V}_{air}}{A_R} \tag{4.4}$$

the Equation 4.3 simplifies to:

$$\frac{U_{S.water}}{U_{S.air}} = k\frac{U_{S.air}}{{}_{\Delta}h_0} \tag{4.5}$$

Where $U_{S.water}$ refers to the superficial velocity at the upper edge of the separation wall and $U_{S.air}$ to the superficial velocity of air at the air inlets. The proportionality constant k has dimension of time and is specific to the reactor geometry.

The simulation data was fitted only for the filling heights at which the operation was feasible. This almost inevitably required filling height below the upper edge of the separation wall to circumvent flooding and enable sufficient aeration. Only the simulations at which the circulation occurred were utilized for the fitting.

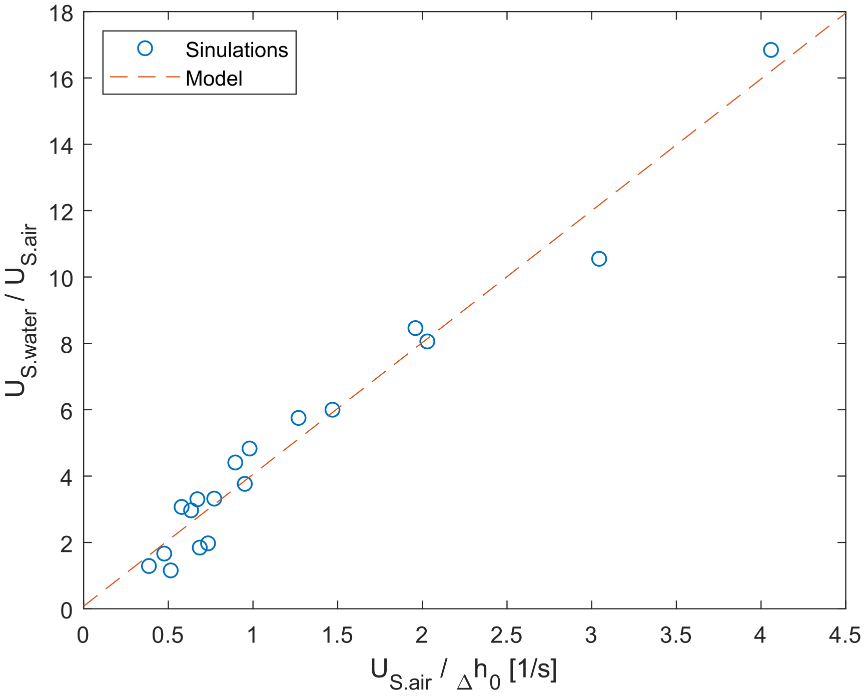


Figure 4.3.: Linearized simulation data and corresponding model

As it can be seen from Figure 4.3, even though the linearization achieved demonstrating the clear trend, the data points still remain scattered. This can be attributed mainly to the way steady-state was identified in the study. The state was considered stable once the extraction phase penetrated to the surface of the reaction mixture. However, the level of the liquid would still exhibit case-specific fluctuations, causing variable recirculation rate. Therefore, the specific value of the recirculation rate depends on the particular time-instances written down by the computational engine. In other words, if the write interval was selected at 0.1 s, the value of the steady-state recirculation was obtained by averaging recirculation at the instances separated by 0.1 s. However, since the fluctuations between the steady-state instances of the various cases were significant, the averaged values did also not follow a perfectly linear trend. Nevertheless, it is reasonable to assume averaging over higher number of steady-state instances would decrease the scattering.

Furthermore, the model does not provide exact boundaries of the model in terms of the malfunction of the reactor. Significantly higher number of simulations would be required to establish what the particular operation window for the given geometry is. However, based on the qualitative observation, rough limits of feasible operation are 0.75 vvm and 1.5 vvm given the validity range of the model of 0.35 1/s $< U_{S.air}/_{\Delta}h_0 <$ 4.5 1/s.

Given the selected conditions and mainly the definition of the steady state, the value of k modelling the hydrodynamics was established at 4.013 s, with R^2 equal to 0.965. The fitting was performed using `cftool` application of MATLAB 2019a. It should be noted, that this value of k is limited to preliminary estimates, as it is highly dependant on the steady-state level, and thus on the method of extraction-phase removal selected in an experimental set-up.

5. Conclusion

To design a novel airlift loop reactor, a computational study was required to provide preliminary understanding of the underlying hydrodynamics. The study was centered around creating a CFD simulation in OpenFOAM, whereby the case was simplified to 2D axially symmetric wedge-shaped domain.

The solution was carried out by a hybrid method combining Volume of Fluid (VOF) and Euler-Euler method. This approach solves a separate set of transport equations for each phase, proceeded by additional interface sharpening for selected phase-pairs. Turbulence was modelled by *realizable* $k - \epsilon$ method, which provided the best compromise between precision and computational expense, even though it was unable to account for the bubble-induced turbulence.

A parametric study was performed, observing effects of gas discharge and filling height on three different geometries, which differed primarily in A_R/A_D. Operation was observed both at the initial transient and at steady-state. Three different types of malfunction were identified, in which either the initial level height was too low relative to the voluemetric flowrate of air, and hence the circulation did not occur, the volumetric flow rate of air was too low and the extraction phase broke through to the riser from the top, or, most commonly, the circulation of of organic phase pushed the extraction phase into the riser from the bottom (flooding). In order to circumvent these issues, careful selection of initial filling height and air volumentric flowrate are of paramount importance. Exact study into the critical values was not perfromed, however based on the qualitative data it was concluded that for the given reactor geometries, the initial filling level needs to be below the upper edge of the separation wall and the combined inlet flowrate of air should be selected roughly in the range between 0.75 vvm and 1.5 vvm.

Subsequently, the hydrodynamics of the reactor was studied quantitatively. A linear relation describing recirculation based on the three studied parameters was identified in the form:

$$\frac{U_{S.water}}{U_{S.air}} = k \left(\frac{U_{S.air}}{\Delta h_0} \right) \tag{5.1}$$

While the advantage of the model is the fact it only utilizes a single empirical constant, its ability to predict the steady-state recirculation is only suitable for preliminary estimates. This is mainly given by the fact the particular steady-state is mainly governed by the method of removal of extraction phase. As the level of liquid inside the reactor rises until the removal is performed, the recirculation remains dynamic as long as the liquid level is not stabilised. Particular method of the removal was beyond the scope of this work, and thus an assumption was accepted, taking designed the method as perfect and capable of removing the extraction phase immediatly after penetrating to the surface of the organic phase. Thus, the state was considered steady once the extraction phase settled at the top of the reactor. Using this simplification, the recirculation rate may be for the given geometry estimated using k of 4.013 s.

There is a series of limitations influencing the precision of the simulations and of the model, requiring an experimental study to assess the scope of error they introduce. The most significant influence on the deviation of the simulation from the experiment is expected to be caused by the removal of the extraction phase, which will govern the actual steady-state liquid level, and thus the steady-state recirculation. Besides that, error of lesser significance is introduced by imperfect empirical data supplied to the simulation, limits of a 2D simulation to capture 3D behaviour or inability of the turbulence model to capture effects of bubble-induced turbulence. Nevertheless, it is expected that this work provides sufficient starting point for preliminary design of the reactor on laboratory scale.

The design of the reactor will need to rely on the bio-chemistry data describing the rate of production for the given operation conditions. These will set the required residence time and aeration rate. The remaining degrees of freedom, which are the ratio of cross-sectional areas and the initial filling height, may be selected. It should be noted, that in the selection it is reasonable to presume, the A_R/A_D would be attempted to be selected rather higher, as higher riser volume implies higher volume of reacting mixture, permitting higher productivity. Furthermore, as it can be observed from the provided model, the recirculation rate depends on the difference between the filling height and the edge of the upper wall. Therefore, the closer the initial filling level to the edge, the higher the influence of the filling height, and the higher precision in the initial filling is required.

Based on the described phenomena, the operation window of the reactor is rather narrow and careful selection of the design and operation parameters are required. Consequently, the operation is expected to be prone to disturbances, making it rather volatile. Therefore, further design should also examine ways of stabilisation of the operation such as introduction of additional pressure drops in the downcomer to circumvent flooding or restricting flow between riser and downcomer to control recirculation.

A. Steady-state Depiction of feasible operation

Demonstrated on geometry R15 at 77.5% filling and air discharge 1 vvm:

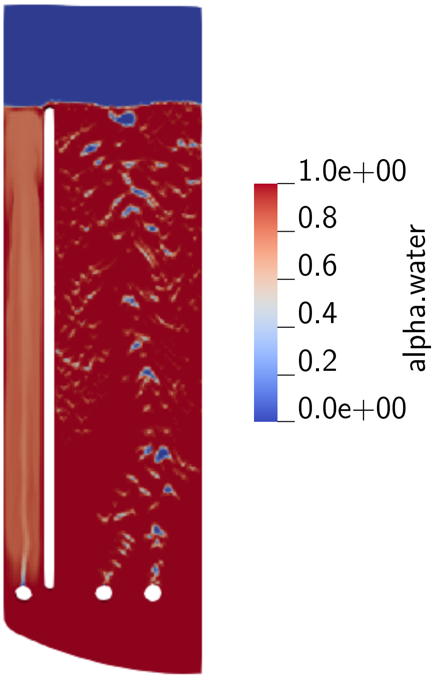


Figure A.1.: Fluid flow

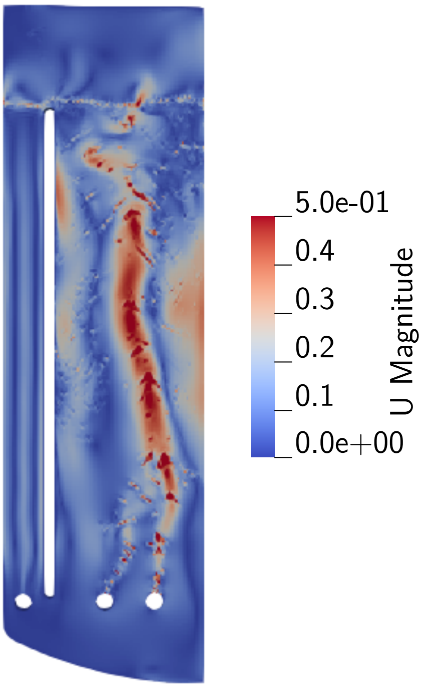


Figure A.2.: Velocity magnitude

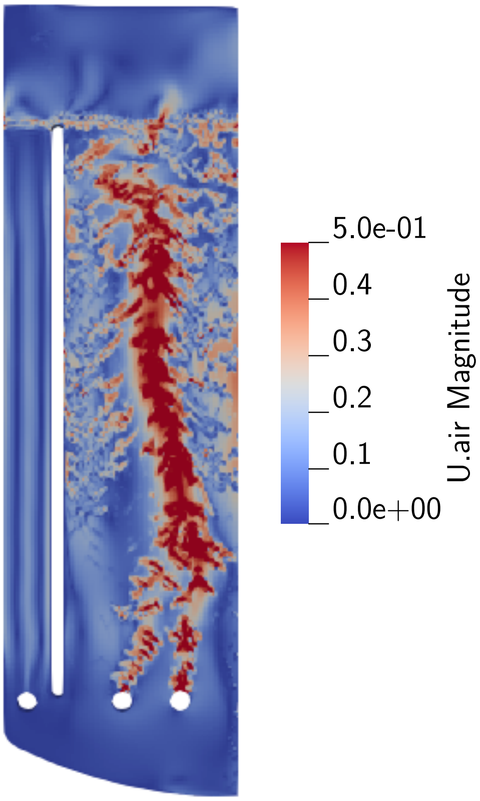


Figure A.3.: Velocity of air

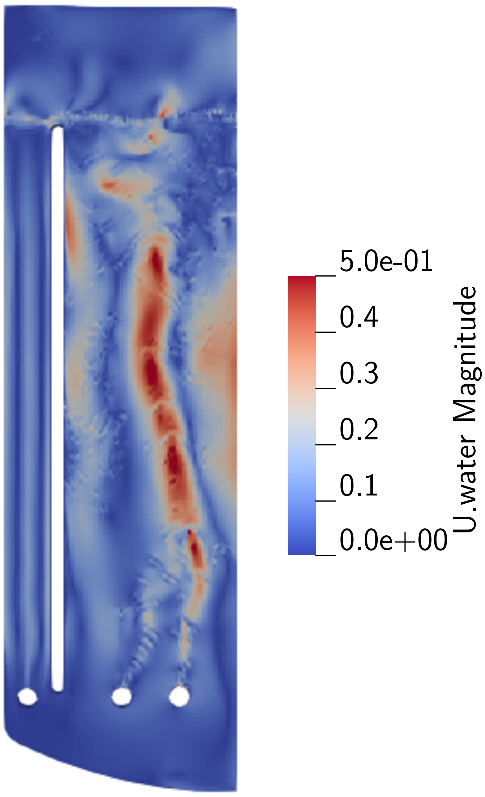


Figure A.4.: Velocity of water

B. Something I should have said at the very beginning ...

... just did not find the courage to occupy two front pages

Even though by now it might be too late, I believe I should still provide a brief crash course of computational fluid dynamics (CFD) to the readers who are approaching this thesis without prior background in this marvellous engineering discipline. Furthermore, since wilfully wasting pages for private messages is not punishable anymore in the appendix, this is also an ideal opportunity to acknowledge some very special people for their invaluable contribution either to this thesis or to the past 25 years of my life. So, very special thanks to....

... my supervisor Maximilian von Campenhausen. In the well-known and often-studied phenomenon of Poiseuille flow, the flow is driven by the virtue of exerted pressure gradient. Similarly, the driving force to which this work owes for its completion was the kind pressure (and encouragement) Maximilian has abundantly supplied me with.

... my friend and hopefully a future colleague Peter Benovsky. Much of progress in the past two decades of CFD was achieved owing to professor David Gosman, who inspired a number of future prominent CFD engineers to pursue research in this field. Given the number of people (including myself) that are now working in CFD thanks to Peter's inspiration, it would be by no means surprising, if the future generations give him equally as much credit as we nowadays give to Gosman.

... the single most influential authority in the world of aesthetics and my beloved sister Gabriela. This work is largely based on findings of very smart people and great figures in the world of contemporary CFD, such Henry Weller, Henrik Rusche, Kent Wardle or Hrvoje Jasak. In the very same fashion (and I really intended to use the word "fashion"), Gabriela was a prominent "influencer" for my outside-of-CFD decision-making.

... to my brother Lukas. Modelling of multi-phase flows for industrial applications was largely enabled by a break-through work of Henrik Rusche on the so called Euler-Euler approach (Rusche, 2002). Although not yet equally appreciated, it is expected that the highly innovative approach of mathematical modelling of membrane systems developed by Lukas Satura, which outclasses the conventional methods so badly that it practically proves them wrong, will deliver a similar step-change to this field (Satura, 2018).

... to my parents Tomas and Gabriela. CFD simulations usually require a lot of computational time on multiple computer cores to produce the required results. This is commonly referred to as them being “expensive”. The very same term is used in a different context, referring to when your son decides to go on a study escapade abroad.

... to my friend Lucia Krcmeryova. The original company distributing OpenFOAM (software used in this work) was folded due to low sales. The reason for this was likely the fact it failed to attract interest on the market because of its poor visual presentation. Experience of several men have since then shown that such visual failures can be circumvented, if one appoints Lucia a personal visual manager.

... to a great Singaporean "souldier" and my another good friend Bryan Chong. In the world of CFD (just as in the world of any contemporary science), the majority of significant breakthroughs are achieved by research teams of ideally two people. The same was the case for Jasak and Weller. However, unlike in their case, which ended up in a fork, ours usually ends up with chopsticks.

... to all my friends, colleagues, classmates and teachers. The finite volume method employed in CFD is based on splitting the domain of the problem to a number of small cells, on which the solution is performed. Despite their high count, the well-known “garbage in, garbage out” principle imposes a rather strict requirement on quality of every single one of them. If I were to apply the same principle to assess the quality of all the great people mentioned above, the fact my life has been very far from garbage would lead to a clear conclusion.

... to the kindest ladies from the Chinese restaurant in Vaals. For simply whether it is CFD or anything else in this world, it is most certainly very hard to do unless one is properly fed.

C. Nomenclature

Abbreviations

vvm	Vessel Volume per Minute
ALR	Airlift loop reactors
CFD	Computational fluid dynamics
DNS	Direct Numerical Simulation
FVM	Finite Volume Method
LES	Large Eddy Simulation
MLR	Multi-phase loop reactor
RANS	Reynolds Averaged Navier-Stokes Equations
VOF	Volume of Fluid

Subscripts

σ	Surface tension
c	Continuous phase
D	Drag
d	Dispersed phase
eff	Effective
t	Turbulent
vm	Virtual mass
B	Bottom
C	Centre

D	Downcomer
E	East
N	North
R	Riser
S	South
T	Top
W	West

Symbols

α	Volume fraction
Δ	Cutoff length-scale
$\dot{V}$	Volumetric flowrater
ϵ	Turbulence dissipation rate
η	Dynamic viscosity
$\hat{n}$	Normal vector
κ	Surface curvature
ν	Kinematic viscosity
$\overline{\nu}$	Mean kinematic viscosity
ϕ	General flow-transported quantity
ρ	Density
τ_w	Wall shear stress
$\vec{\xi}$	Fluctuation vector
$\vec{g}$	Gravitational acceleration
$\vec{U}$	Velocity
A	Area
C_α	Phase compression switch

C_D	Drag coefficient
c_f	Friction coefficient
C_μ	Turbulence model constant
C_{vm}	Virtual mass coefficient
C_{vm}	Virtual-mass constant
Co	Courant number
D	Diameter
d	Diameter
D_T	Diffusivity
Eo	Eotvos number
F_i	Closure model, modelling effect of a force
h	Height
k	Turbulence kinetic energy
L	Characteristic length
l_e	Turbulence length-scale
p	Pressure
Re	Reynolds number
Sc	Schmidt number
t	Time
u'	Deviation from mean velocity in x-direction
u_T	Dimensionless speed
V	Volume
v'	Deviation from mean velocity in y-direction
w'	Deviation from mean velocity in z-direction
y^+	Dimensionless distance from the wall

y_C	Height of a cell centre
z	Dimension coordinate

Bibliography

Andersson, B., Andersson, R., Hakansson, L., Mortensen, M., Sudiyo, R., & van Wachem, B. 2012. *Computational Fluid Dynamics for Engineerss*. Vol. 1. New York: Cambridge University Press.

Anez, J., Demoulin, F., Hecht, N., & Reveillon, J. 2016. A general purpose LES Model for Atomization. *In: European Conference Liquid Atomization*. Brighton: Spray Systems.

Bannari, R., Kerdouss, F., Selma, B., Bannari, A., & Proulx, P. 2008. Three-dimensional mathematical modeling of dispersed two-phase flow using class method of population balance in bubble columns. *Computers and Chemical Engineering*, **32**, 3224–3237.

Bednarz, A., Jupke, A., Schmidt, M., & Weber, B. WO 2017/149099 AI, Sep. 2017. *Multi-phase Loop Reactor and Method of Operation*.

Bhusare, V.H., Dhiman, M.K., Kalaga, D.V., Roy, S., & Joshi, J.B. 2017. CFD simulations of a bubble column with and without internals by using OpenFOAM. *Chemical Engineering Journal*, **317**, 157–174.

Brackbill, J.U, Kothe, D.B, & Zemach, C. 1992. A continuum method for modeling surface tension. *Journal of Computational Physics*, **100**(2), 335 – 354.

Buwa, Vivek V., Deo, Dhanannjay S., & Ranade, Vivek V. 2006. Eulerian-Lagrangian simulations of unsteady gas-liquid flows in bubble columns. *International Journal of Multiphase Flow*, **32**(7), 864–885.

Cerne, G, Petelin, S., & Tiselj, I. 2001. Coupling of the Interface Tracking and the Two-Fluid Models for the Simulation of Incompressible Two-Phase Flow. *Journal of Computational Physics*, **171**(2), 776 – 804.

Chisti, M.Y., Halard, B., & Moo-Young, M. 1988. Liquid circulation in airlift reactors. *Chemical Engineering Science*, **43**(3), 451 – 457.

Darmana, D., Deen, N.G., & Kuipers, J.A.M. 2005. Detailed modeling of hydrodynamics mass transfer and chemical reactions in a bubble column using a discrete bubble model. *Chemical engineering science*, **60**(12), 3383–3404.

Greenshields, C. J. (CFD Direct Ltd). 2019. *OpenFOAM User Guide*. OpenFOAM Foundation Ltd.

Hlawitschka, M. W., Drefenstedt, S., & Bart, H.J. 2016. Local Analysis of CO2 Chemisorption in a Rectangular Bubble Column Using a Multiphase Euler-Euler CFD Code. *J Chem Eng Process Technol*, **7**(300).

Holzmann, Tobias. 2016. *Mathematics, Numerics, Derivations and OpenFOAM®*.

Jasak, H., & Weller, H.G. 1998. *Interface Tracking Capabilities of the Inter-Gamma Differencing Scheme*. . Imperial College London, UK.

Kools, Fabian. 2016. *CFD-simulation of multiphase systems for design optimisation of an innovative reactor concept with integrated product separation*. bachelors, RWTH Aachen, Germany.

Krishna, R., van Baten, J. M., & Urseanu, M. I. 2000. Three-phase Eulerian simulations of bubble column reactors operating in the chur-turbulent regime: A scale upstrategy. *Chemical Engineering Science*, **55**.

Lahey, R. T., de Bertodano, M. L., & Jones, 0. C. 1994. Development of a k-e Model for Bubbly Two-Phase Flow. *Journal of Fluids Engineering*, **116**, 128–134.

Li, Z., & Lu., B. C.-Y. 2001. A molecular model for representing surface tension for polar liquids. *Chemical Engineering Science*, **56**, 6977–6987.

Lu, Wen-Jang, Hwang, Shyh-Jye, & Chang, Chun-Min. 1994. Liquid Mixing in Internal Loop Airlift Reactors. *Industrial & Engineering Chemistry Research*, **33**(9), 2180–2186.

Markos, J., Blazej, M., & Kisa, M. 2004. Scale influence on the hydrodynamics of an internal loop airlift reactor. *Chemical Engineering and Processing*, **43**, 1519–1527.

Matsuo., S., & Makita, T. 1989. Volumetric Properties of 1-Alkanols at Temperatures in the Range 298-348 K and Pressures up to 40 MPa. *International Journal of Thermophysics*, **10**(4), 885–897.

Miles, J.P., & Farrashkhalvat, M. 2003. *Basic Structured Grid Generation with an introduction to unstructured grid generation*. Vol. 1. Oxford: Elsevier Science.

Monahan, S. M., Vitankar, V. S., & Fox, R. O. 2005. CFD Predictions for Flow-Regime Transitions in Bubble Columns. *American Institute of Chemical Engineers Journal*, **51**(7), 1897–1923.

Motomura, Kinsi. 1991. Interfacial tension of organic solutions. *Pages 129–137 of:* Pétré, G., & Sanfeld, A. (eds), *Capillarity Today*. Berlin, Heidelberg: Springer Berlin Heidelberg.

openfoamwiki.net. Contrib/MakeAxialMesh. https://openfoamwiki.net/index.php/Contrib /MakeAxialMesh. Accessed:2019–08–09.

Pudasaini, Shiva P. 2019. A fully analytical model for virtual mass force in mixture flows. *International Journal of Multiphase Flow*, **113**, 142–152.

Rusche, Henrik. 2002. *Computational Fluid Dynamics of Dispersed Two-Phase Flows at High Phase Fractions*. Ph.D. thesis, Imperial College London.

Sagaut, Pierre. 2002. *Large Eddy Simulation for Incompressible Flows*. Vol. 3. Berlin, Heidelberg: Springer.

Satura, Lukas. 2018. Modelling and Optimisation of Membrane Process Operation. *Bachelor Thesis, Slovak University of Technology Bratislava*, Slovakia.

Schiller, L., & Naumann, A. 1935. A drag coefficient correlation (English trans.). *Zeitschrift des Vereins Deutscher Ingenieure*, **77**, 318–320.

Selma, B., Bannari, R., & Proulx, P. 2010. Simulation of bubbly flows: Comparison between direct quadrature method of moments (DQMOM) and method of classes (CM). *Chemical Engineering Science*, **65**(6), 1925 – 1941.

Shonibare, O. Y., & Wardle, K. E. 2015. Numerical Investigation of Vertical Plunging Jet Using a Hybrid Multifluid-VOF Multiphase CFD Solver. *International Journal of Chemical Engineering*, **2015**.

Simcik, M., Mota, A., Ruzicka, M.C., Vicente, A., & Teixeira, J. 2011. CFD simulation and experimental measurement of gas holdup and liquid interstitial velocity in internal loop airlif treactor. *Chemical Engineering Science*, **66**, 3268–3279.

SURFACE TENSION.DE. Surface tension values of some common test liquids for surface energy analysis. http://www.surface-tension.de. Accessed:2019–08–09.

TOCCI, FRANCESCO. 2016. *Assessment of a hybrid VOF two-fluid CFD solver for simulation of gas-liquid flows in vertical pipelines in OpenFOAM*. M.Phil. thesis, Politecnico di Milano, Italy.

TOMIYAMA, A. 2004. Drag lift and virtual mass forces acting on a single bubble. *Pages 22–24 of: Third international symposium on two-phase flow modeling and experimentation*.

UBBINK, ONNO. 1997. *Numerical prediction of two fluid systems with sharp interfaces*. Ph.D. thesis, Imperial College London.

VERSTEEG, H. K., & MALALASEKERA, W. 2007. *Basic Structured Grid Generation with an introduction to unstructured grid generation*. Vol. 2. Harlow: Pearson Prentice Hall.

WARDLE, K. E., & WELLER, H. G. 2013. Hybrid Multiphase CFD Solver for Coupled Dispersed/Segregated Flows in Liquid-Liquid Extraction. *International Journal of Chemical Engineering*, **2013**, 1–13.

WEBER, BENEDIKT. 2015. *CFD-simulation of multiphase systems as feasibility study of an innovative reactor concept with integrated product separation*. M.Phil. thesis, RWTH Aachen, Germany.

WELLER, H.G. 2008. *A new approach to VOF-based interface capturing methods for incompressible and compressible flow*. . OpenCFD.

XU, T., JIANG, X., YANG, N., & ZHU, J. 2014. CFD simulation of internal-loop airlift reactor using EMMS drag model. *Particuology*, **19**, 124–132.

ZHANG, T., WE, C., REN, Y., C., FENG, & WU, H. 2016. Advances in airlift reactors: modified design and optimization of operation conditions. *Reviews in Chemical Engineering*, **33**(2), 163–182.