

DELFT UNIVERSITY OF TECHNOLOGY

MASTER THESIS CHEMICAL ENGINEERING

Scale-independent CFD modeling of gas holdup and liquid velocity in bubble columns

Name: Bram vanden Abbeele
Student number: 5417651
Date: 12/09/2022 - 03/07/2023
Daily supervisor: ir. Lars Puiman
Thesis committee: Dr. ir. C. Haringa,
Dr A.J.J. Straathof,
Dr. ir. D.A. Vermaas

April 11, 2022



Abstract

This research aims to address the challenge of finding sustainable methods for producing hydrocarbons in the context of a transition away from fossil fuels. Specific attention is given to the process of syngas fermentation, where syngas is used to produce ethanol. Since this process always had its limitations when it comes to mass transfer, this research focuses on the possibilities of integrating this process into the bubble column reactor, since previous studies have shown promising results for improved mass transfer with this type of reactor.

Ethanol, as the primary product of syngas fermentation, has a substantial impact on reactor hydrodynamics that engineers must account for during scale-up. CFD proves to be a cost-effective way to study these effects, providing valuable insights without the need for expensive experiments. To achieve accurate simulations of syngas fermentation, it is crucial to establish a foundation by investigating a water and air bubble column system. Experimental data for four reactor diameters (0.15m, 0.4m, 1.0m and 3.0m) have been utilized for this purpose.

This study intends to assess the effectiveness of a single CFD model in simulating hydrodynamic behaviours as gas holdup and liquid velocity and its scale independence. Beginning setting up a base model by replicating a water-air bubble column with a diameter of 0.4m and 1.0m used in the experimental setup. Subsequently, a previously developed model for bubble column reactors was step-wise incorporated. Through comparisons of simulation results with established correlations, the effects and instabilities associated with each step were identified, allowing us to setup an improved model.

The results obtained from the CFD simulations demonstrated that the implementation of the NITA iterative scheme reduced the simulation time by 3.7 and 5.3 times for $D=0.4\text{m}$ and $D=1.0\text{m}$ compared to the standard iterative scheme. The inclusion of the Yao and Morel BIT model greatly improved the ability to replicate experimental results. Incorporating the ideal gas law had a notable impact on the reactor with a diameter of 1.0m, as expected due to a 37% increase in gas density. However, it compromised the ability to predict hydrodynamics. Nevertheless, considering the fundamental nature of the ideal gas law as an equation relating pressure and volume, it was retained in the model. Although the incorporation of turbulent dispersion, Grace drag formulation and the RNG $k-\epsilon$ turbulence model showed ambiguous results, the Grace drag formulation and turbulent dispersion were included in the improved model in analogy with previous research.

The comparison of the improved model with experimental data revealed instability issues when simulating bubble columns with diameters of 0.15m and 3.0m. The model also failed to accurately capture the scale-independent gas holdup profile and exhibited a dependence on the measuring height, contrary to the consistent gas holdup observed in experiments. Although higher-order discretization improved simulation outcomes for $D=1.0\text{m}$, convergence issues arose, resulting in inconsistent gas holdup profiles. However, this higher-order discretization corresponds well with results derived from previous CFD models.

This assessment of the effectiveness of a single CFD model in simulating hydrodynamic behaviours across various column diameters for the design and optimization of bubble columns for syngas fermentation led to a more clear perspective on the strengths and weaknesses of the current model and a first attempt towards an improved model. Although the current model successfully simulated the reactor with diameters of 0.4m and 1.0m, it fell short in accurately reproducing gas holdup, liquid velocity profiles and the scale-independence phenomena. Nevertheless, a crucial initial step towards the goal of accurately simulating bubble columns for syngas fermentation has been made.

Contents

1	Introduction	7
1.1	Background	8
1.2	Objective	8
1.3	Research question	8
2	Theoretical background	9
2.1	Syngas fermentation	9
2.2	Bubble column reactor	9
2.2.1	Flow regimes	10
2.2.2	Bubble rise velocity	11
2.2.3	Gas holdup	12
2.2.4	Liquid velocity	13
2.3	scale-up	14
2.3.1	Wilkinson scale-up criteria	14
2.4	Computational fluid dynamics and bubble columns	15
2.4.1	Governing equations	15
2.4.2	Multiphase Modeling	15
2.4.3	Drag force model	16
2.4.4	Bubble swarms	17
2.4.5	Turbulence dispersion	18
2.4.6	Turbulence modeling	19
2.4.7	Multiphase turbulence model	20
2.4.8	Bubble induced turbulence	20
2.4.9	Non-iterative time advancement scheme (NITA)	21
3	Model development	22
3.1	Model geometry	22
3.2	Meshing	22
3.3	CFD model	23

3.3.1	Momentum transfer	24
3.3.2	Turbulence model	24
3.4	User defined function (UDF)	25
3.4.1	Swarm modification	25
3.4.2	BIT model	26
3.5	Assumptions, materials and boundary conditions	27
3.5.1	Assumptions	27
3.5.2	Materials	27
3.5.3	Boundary conditions	28
3.5.4	Solution	28
3.6	Data gathering and data processing	31
3.7	Validation	33
3.8	Simulation strategy	33
3.9	Model summary	35
4	Results and discussion	37
4.1	Base model	37
4.2	Iterative scheme	38
4.3	Swarm modification	39
4.4	BIT	42
4.5	Turbulent dispersion	43
4.6	Drag model	45
4.7	Turbulence modeling	47
4.8	Inclusion of the ideal gas law	48
4.9	Improved model	50
4.10	Detailed model comparison	52
4.10.1	Scale independent gas holdup	52
4.10.2	Measurement height	53
4.11	Mesh dependency	54
4.12	Higher order discretization	55

4.13 Ertekin model	56
5 Conclusion	58
6 Recommendations	60
6.1 Validation	60
6.2 Measuring range	60
6.3 Gas density	60
6.4 Simonnet model instability	60
6.5 Furter model improvement	61

List of abbreviations

Abbreviation	Description
AR	Aspect ratio
BCR	Bubble column reactor
BIT	Bubble Induced Turbulence
CFL	Courant–Friedrichs–Lewy
CODH	Carbon monoxide dehydrogenase
CO	Carbon monoxide
CO ₂	Carbon dioxide
CPU	Central processing unit
CFD	Computational Fluid Dynamics
CSTR	Continues stirred reactor
DNS	Direct Numerical Simulation
E ₀	Eötvös number
Exp	Experimental
H ₂	Hydrogen
ITA	Iterative time advancement
LES	Large Eddy Simulation
Mo	Morton number
NITA	Non iterative time advancement
N ₂	Nitrogen
O ₂	Oxygen
RANS	Reynolds-Averaged Navier-Stokes
Re	Reynolds number
RNG	Renormalization Group
RMSE	Root mean square error
Sim	Simulation
UDF	User Defined Function
UDMI	User Defined Memory
WLP	Wood-Ljungdahl pathway

Constants

Symbol	Description	Value	Unit
a	Liquid velocity correlation constant	2.976	-
b	Liquid velocity correlation constant	0.943	-
b _d	Diameter bubble	5.1	mm
C3	$k - \epsilon$ Pflieger and Becker BIT constant	1.0	-
C _{ϵ}	$k - \epsilon$ turbulence model constant	1.3	-
C _{vm}	virtual mass coefficient	0.5	-
c	Liquid velocity correlation constant	1.848	-
g	Gravitational acceleration	9.81	m/s ²
m	Exponent in drag correction equations	25	-
P _{Headspace}	Headspace pressure	101325	Pa
ρ_L	Liquid density	998.02	kg/m ³
μ_L	Dynamic viscosity	0.001	kg/(m s)
Sc _{t,L}	Turbulent Schmidt number for the bulk phase	0.9	-
σ_L	Surface tension	0.072	N/m
T _{BCR}	Temperature	293.15	K

Symbols

Symbol	Description	Unit
C_{vm}	Virtual mass coefficient	-
CD	Drag coefficient	-
CD _{Swarm}	Swarm drag coefficient	-
CD _∞	Single bubble drag coefficient	-
D	Reactor diameter	-
ϵ	Dissipation rate of turbulent kinetic energy	m ² /s ³
F_D	Drag force	N
F_{TD}	Turbulent dispersion force	N
H ₀	Static liquid height	m
H _{BCR}	Bubble column reactor height	m
H/D	Ratio of measuring height to reactor diameter	-
k	Turbulent kinetic energy	m ² /s ²
M ^{inter}	Interface momentum transfer	kg/(m ² s ²)
\dot{m}_G	Mass gas flow inlet	kg/s
n	Exponent in drag correction equations	-
p	Pressure	Pa
R	Bubble column radius	m
t	Time	sec
u	Velocity vector	-
U_{trans}	Superficial gas velocity at regime transition	m/s
V ₀	Central Liquid velocity	m/s
V _G	Gas volume	m ³
V _L	Liquid volume	m ³
Δx	cell size	m
$\nabla \alpha_G$	Gradient of gas phase fraction	-
ν_{bs}	Single bubble rise velocity	m/s
ν_{sg}	superficial gas velocity	m/s
α_G	Gas holdup fraction	-
$\bar{\alpha}_G$	Average gas holdup	-

1 Introduction

Nowadays, significant efforts are dedicated to developing a sustainable future, aiming to move away from fossil fuels and thereby limiting climate change[1]. However, the majority of the products and materials we currently rely on are still carbon-based. Consequently, finding sustainable methods for producing hydrocarbons has proven to be a challenging endeavor. One such method involves the conversion of syngas [2], a gas mixture primarily composed of carbon monoxide (CO), carbon dioxide (CO_2), and hydrogen (H_2). The syngas is typically derived from industrial processes, where otherwise it would be burned and the resulting CO_2 vented into the atmosphere; syngas utilization hence reduces the emission of greenhouse gasses [2]. By utilizing this syngas, the emission of greenhouse gasses is reduced.

Traditionally, the conversion of syngas has been achieved by the Fischer-Tropsch process, but this process relies on relatively high temperatures (150 – 300 °C) and pressures (30 bar) [3]. Moreover, this process relies on iron and cobalt catalysts, which require a specific H_2/CO ratio, to avoid catalyst poisoning [4]. A promising alternative for the chemical conversion is the use of microorganisms in a process called syngas fermentation. This fermentation is performed in a multiphase bioreactor, where syngas is converted to ethanol, offering several advantages. The process operates under mild conditions, reducing the overall energy demand and greenhouse emissions. Unlike the Fischer-Tropsch process, syngas fermentation does not require a specific H_2/CO ratio nor the use of metal catalysts [5].

However, it is important to note that the aqueous solubility of H_2 and CO is relatively low. Making the gas fermentation process limited to the gas-to-liquid mass transfer [6]. In a continuous stirred reactor (CSTR), one commonly employed method to enhance mass transfer limitations, is an increase of the power input to the impeller. This power increase leads to improved bubble breakup, resulting in a larger gas-liquid interface and thus mass transfer [7]. The use of stirred fermentors is unfavorable due to the high power input, which may limit economic feasibility, especially in the production of bulk chemicals with a relatively low value per tonne. While mass transfer increases linearly with stirrer speed, power consumption exhibits a cubic increase [8][9]. Therefore, achieving high mass transfer rates in an energy-efficient manner at industrial scale poses a significant challenge.

The bubble column reactor offers a promising solution to this challenge, as it requires a minimal amount of energy to achieve a high mass transfer coefficient. By introducing small bubbles at the reactor's bottom through sparging, a large surface area is created to enhance mass transfer [7][10].

The most common product of syngas fermentation is ethanol, which is known to have a profound influence on the bubble size distribution in the reactor through the decrease of surface tension, resulting in less bubble coalescence and subsequently decreasing the bubble mean Sauter diameter, and an increase in gas holdup [11][12][13][14]. It is essential that engineers account for these aspects in scale-up.

Experimental investigations and computational simulations, such as computational fluid dynamics (CFD), can provide insights into these phenomena without having to build expensive experimental setups. Therefore, conducting further research on the ability of CFD to capture important parameters such as gas holdup, liquid velocity and scale dependency of syngas bubble column design in relation to hydrodynamics, using CFD offers a promising avenue for investigation.

1.1 Background

In order to accurately simulate syngas fermentation using CFD, it is important to establish a foundation. This involves initially investigating a water and air bubble column system to determine if the gas holdup and velocity profiles can be reproduced using CFD. To conduct CFD simulations of bubble columns, reliable data and correlations are essential for validation. The available data is primarily constrained to bubble columns operating within the homogeneous regime. When considering reactors running in the heterogeneous regime, the availability of such data is limited, and there is even less data for non-water air mixtures.

Raimundo addressed this issue by providing an experimental database specifically designed for validation purposes [15] of heterogeneous run bubble columns. The experiments conducted by Raimundo involved a range of reactor diameters, ranging from 0.15m to 3m, with air and tap water as the gas-liquid mixture. These experiments were used to validate correlations for liquid velocity and gas hold-up. It was decided to begin with the water-air bubble column from Raimundo's experimental setup, as it had been validated using experimental correlations, making it valuable for comparison with CFD results (appendix A). Due to limited data and the absence of correlations for mixtures containing ethanol, validating the CFD results would prove challenging, therefore it was decided to start with a water air bubble column. Raimundo conducted experiments across a range of reactors, enabling us to investigate the ability of the CFD model to replicate the scale-independence of the bubble columns.

Regarding the existing work in this field of water-air bubble column CFD, Ertekin successfully replicated the gas holdup and liquid velocity profiles for both the 1m and 3m diameter columns using the database provided by Raimundo [16]. She did this using the model developed by Fletcher [17].

The objective of Fletcher's research was to assess the influence of various factors, including lift, drag, BIT (Bubble induced turbulence), and volume fraction correction terms for drag [18]. This investigation aimed to identify the most suitable models necessary to accurately simulate experimental bubble column results with CFD. Fletcher's findings revealed that incorporating terms to account for the drag reduction attributed to the presence of additional bubbles, along with the inclusion of BIT, was needed to achieve good agreement between simulation results and experimental data obtained from a reactor characterized by an initial liquid height of 1m and a diameter of 0.19m.

In this work, the initial step involved replicating the water-air bubble column geometry used in Raimundo's experimental setup. Subsequently, Fletcher's model [17] was incrementally integrated. The obtained results were compared with Ertekin's findings [16], as described in detail in the paragraph above, to identify any existing gaps in reproducing Fletcher's model.

1.2 Objective

The objective of this research is to develop a single CFD model capable of accurately simulating gas holdup and liquid velocity behavior in bubble columns across a range of scales, varying from 0.15m to 3m in diameter. The aim is to effectively capture the scale-independence gas holdup and liquid velocity while providing reliable predictions of gas holdup and liquid velocity. By achieving this objective, the research aims to enhance the understanding of bubble column hydrodynamics and enable more efficient process scale-up.

1.3 Research question

Can a single CFD model, developed by replicating experiment geometries, incrementally integrating a previously developed model, and validating simulation data with experimentally derived correlations, accurately simulate gas holdup and liquid velocity behavior in bubble columns ranging from 0.15m to 3m in diameter while effectively capturing scale-independence phenomena?

2 Theoretical background

This chapter introduces the concept of syngas fermentation in a bubble column reactor. After a short introduction on the syngas fermentation process, the bubble column reactor's physics are explained. This includes a introduction of its main characteristics, such as gas holdup and liquid velocity. Eventually leading us to an explanation on how a bubble column is simulated in CFD, going into the necessary models and methodologies employed in this computational approach.

2.1 Syngas fermentation

Syngas fermentation is a process where a gaseous mixture containing at least CO is to converted to a hydrocarbon product, using microorganisms. This is mainly ethanol, although other products are being explored [19]. The microorganisms used for this process are known as acetogens [5][20]. These microorganisms utilize the acetyl-CoA pathway known as the Wood-Ljungdahl pathway (WLP). In this pathway, CO and H_2 are used for their electrons and the CO and CO_2 are used as the carbon source. CO and H_2 release their electrons by the enzymes carbon monoxide dehydrogenase (CODH) for the CO and hydrogenase for the H_2 . The acetyl-CoA synthesis uses coenzyme A to form acetyl-CoA, by combining a carboxyl group derived from CO or CO_2 reduction, with a methylgroup originating from the reduction of CO_2 . The acetyl-CoA is then converted into organic acids and alcohols [20]. The most studied bacteria exploiting the acetyl-CoA pathway are the *c.ljungdahlii* and the *c.autoethanogenum* strains, both showing the ability to produce ethanol [21][3].

The main bottleneck of the syngas fermentation is the low solubility of CO and H_2 , which limits the mass transfer rate. In water at 25-37 °C and 1 atm, the solubility of hydrogen ranges from 0.0016-0.0014 g/L, while the solubility of CO ranges from 0.0016–0.0013 g/L. It is noteworthy that these solubility's represent only 60% and 3% of the solubility of oxygen, on a mass basis [22]. The low aqueous solubility's of CO and H_2 make the production of ethanol dependent on the gas-to-liquid mass transfer rate, which leads to a reduced productivity [5][7].

2.2 Bubble column reactor

As the gas fermentation is limited by the mass transfer, finding a way of achieving a high mass transfer rate in an energy efficient manner is needed to commercialize gas fermentation as an alternative way of producing ethanol. A bubble column is a promising reactor design for this purpose, as it can provide this large interfacial area. A bubble column reactor is a type of multiphase reactor widely used in various chemical and biochemical processes. It consists of a cylindrical column filled with a liquid, usually a solution or suspension, in which gas is sparged from the bottom, creating small bubbles and thus a large gas-liquid surface area, as can be seen in figure 1. After leaving the sparger, the bubbles coalesce with other bubbles. The rate at which this happens is dependent upon the liquid composition, as well as several other surface active components [10]. As figure 1 shows, a bubble column reactor can operate in continuous mode, allowing the liquid solution to

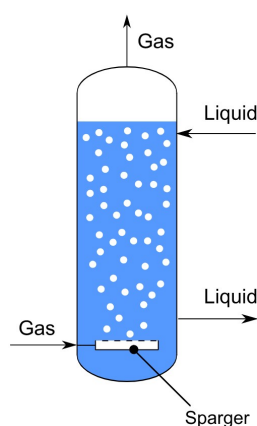


Figure 1: Bubble column reactor running in Continuous mode [23]

go in and out the reactor, allowing a continuously circulation for the extraction of products. Alternatively, it can also be operated in batch mode, where the liquid solution is held within the reactor.

As gas bubbles rise, they provide a large interfacial area between the gas and liquid phases, facilitating mass transfer between the two phases. This mass transfer can go two ways: the process is called stripping when dissolved gasses go from the liquid to the gas phase and when one or more components from the gas phase are absorbed by the liquid phase, we can speak of absorption.

The flow pattern in a bubble column is driven by the momentum exchange between the gas and liquid phase, creating a dynamic flow pattern within the reactor [24]. This dynamic behaviour makes it difficult to predict the gas holdup and liquid velocity within the reactor. However, when we time-average the gas holdup and liquid velocity a stable pattern emerges [25][26]. In heterogeneous flows, it is observed that in the center of the column the gas holdup is greater, taking with it the liquid phase, leading to an upward liquid velocity in the middle of the column[26].

As syngas fermentation encounters challenges related to mass transfer, there is the necessity to finding the optimal reactor design. In this regard, the bubble column is seen as a favorable choice for syngas fermentation. Its ability to provide a large interfacial area, and effective mixing proves to be ideal for delivering the reactants to microorganisms while minimizing mass transfer limitations. As said, The gas bubbles also contribute to the mixing of the liquid phase. With the movement of the bubbles turbulence is generated promoting liquid mixing, helping with the distribution of reactants.

Furthermore, a large variety of sparger designs have been proposed to improve mass transfer, from simple porous plates to more complex injectors that are able to generate very small bubbles and thus making bubble columns even more efficient.

2.2.1 Flow regimes

In a bubble column two types of hydrodynamic flow regimes and one transitional regime can be identified, one at low and one at high superficial gas velocities, and one in between. The bubbly flow (homogeneous) regime is observed at low superficial gas velocities. In this regime a narrow bubble size distribution is observed, generally in the range of 1-7 mm [27], and an almost uniform axial gas holdup. In this regime large-scale circulation flows are absent [28]. This regime will hold until a certain superficial gas velocity (U_{trans}) is reached. When reached, coalescence of bubbles takes place producing the first "large" bubbles. This region is called the transition regime [27]. The superficial gas velocity at which this occurs depends on the liquid and gas, as for water and air it lays at a superficial gas velocity of 0.02-0.04 m/s [10][29]. The churn turbulent flow regime is observed after the transition regime, at higher superficial gas velocities, in which a broader bubble size distribution is observed, which is independent of the sparger design. The gas holdup is no longer uniform, and shows a parabolic profile with higher gas holdups in the center of the column compared to near the wall [28]. These three flow regimes are shown in figure 2.

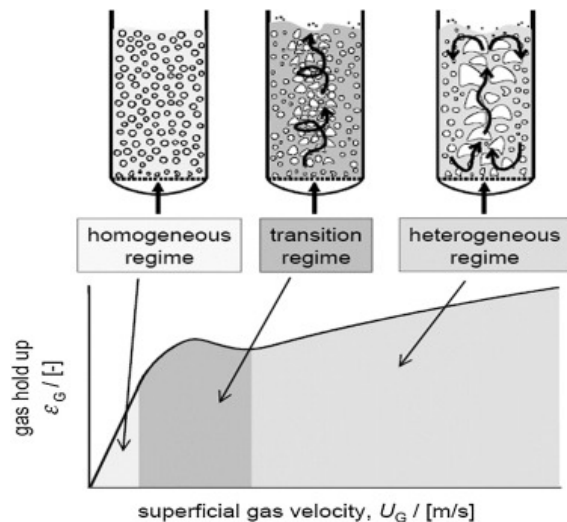


Figure 2: Bubble column flow regimes [27]

These three flow regimes are shown in figure 2.

Alcoholic solutions, such as ethanol, have shown an increase in U_{trans} [29], implying a homogeneous regime that remains stable at higher superficial gas velocities. The presence of ethanol in the liquid phase reduces the surface tension, thereby suppressing coalescence and resulting in a reduction in bubble diameter and an increase in stability of the flow regime. [12] Moreover, the bubbles exhibit not only smaller sizes but also a more spherical shape, as observed by Besagni [11]. Consequently, the stabilization of the homogeneous regime leads to an increase in gas holdup [27][11].

2.2.2 Bubble rise velocity

The rise velocity of a bubble is an important parameter to determine for example the momentum transfer of the dispersed phase on the continuous phase. It can also be used to get an initial guess for the gas holdup, as will be shown in section 2.2.3. A common way to determine the bubble rise velocity is by examining the shape of the bubble. Although Commonly used, problems may arise when applying this technique since the shape of the bubble is also dependent on the rise velocity. Wallis therefore suggested different equations to calculate the rise velocity, using the force balance for a spherical particle subjected to gravitational and buoyancy forces in which the drag coefficient is a function of the Reynolds number. By combining these equations he was able to make them dimensionless, thus containing only one variable, being either radius or rise velocity. Consequently, different rise velocity regions could be determined [30]. Several more attempts were made until Grace analyzed more data from bubbles rising in liquids and by using the dimensionless numbers Reynolds (velocity characteristic), Eötvös E_o (bubble size characteristic) and the Morton number (liquid property characteristics), later referred to as; Re , Eo , Mo (Chapter 2.4.3 we will go further into these dimensionless numbers). With these numbers he was able to divide the bubbles into three shapes; spheres, ellipses and spherical caps [31]. These shapes are shown in figure 3.



Figure 3: Bubble shapes from left to right spheres, ellipse and spherical cap. [32]

Spherical bubbles arise when the inertial forces are much smaller than the surface tension or viscous forces. As the bubble diameter grows and thus the rise velocity increases, the bubble changes to a spheroid form due to the resistance of the fluid on the bubble. When the bubble gets even bigger the rise velocity again increases and the bubble becomes even flatter and the middle of the sphere gets a dent.

The terminal rise velocity (v_{bs}) for a bubble dispersed in water of different shapes can be described as; [30][33]

Spherical bubble:

$$v_{bs,shp} = \begin{cases} \frac{g\rho_L d_b^2}{12\mu_L} & R_e < 1 \\ 0.14425g^{\frac{5}{6}} \left(\frac{\rho_L}{\mu_L}\right)^{2/3} d_b^{3/2} & 1 \leq R_e < 100 \end{cases} \quad (2.1)$$

Ellipse bubbles:

$$v_{bs,ell} = \sqrt{\frac{2.14\sigma_L}{\rho_L d_b} + 0.505gd_b} \quad (2.2)$$

and spherical cap bubbles:

$$v_{bs,cap} = 0.721\sqrt{gd_b} \quad (2.3)$$

Where, g equals the gravitational acceleration, ρ_L the liquid phase density, μ_L the liquid viscosity, d_b represents the bubble diameter and σ_L the surface tension. To then determine the bubble rise velocity, we can fill in the equations and see which one provides the lowest value. But, for simplicity reasons it can be assumed that the single bubble rise velocity is 0.25 m/s for $1 \text{ mm} < d_b < 10 \text{ mm}$. For bubbles $d_b > 10 \text{ mm}$ it is however needed to consider take the diameter into account as the rise velocity increases significantly with the bubble diameter [10], due to the increase buoyancy forces.

2.2.3 Gas holdup

Gas holdup can be seen as one of the more important parameters for mass transfer in a bubble column, since a greater gas holdup means a larger interfacial area. The average gas holdup in a bubble column is described by the volumetric ratio of gas versus the total volume of fermentation broth, given by equation 2.4. Not including the solid volume, since there are no solid particles in the bubble column for syngas fermentation.

$$\overline{\alpha_G} = \frac{V_G}{V_G + V_L} \quad (2.4)$$

Equation 2.4, requires measurement of the volume increase, but the gas holdup can also be estimated. One way of estimating the average gas holdup in homogeneous flows without having to measure it nor having to run a simulation, is derived from the superficial gas velocity and the single bubble rise velocity, see equation 2.5 [10].

$$\overline{\alpha_G} = \frac{\nu_{sg}}{\nu_{bs}} \quad (2.5)$$

Where, ν_{sg} represents the superficial gas velocity and ν_{bs} the single bubble rise velocity. This equation clearly exposes the need to know bubble rise velocity. This value mainly depends upon the bubble diameter as previously explained in 2.2.2, where we saw that for bubbles with a diameter between $1 \text{ mm} < d_b < 10 \text{ mm}$, the rise velocity can be assumed to be 0.25 m/s [10]. Since bubble columns are mostly run in the heterogeneous regime, meaning that we can't use equation 2.5.

Equation 2.4 and 2.5 already gave us some handles in determining the gas holdup, where equation 2.4 could be used when we know the volume increase and in case of homogeneous flow $\nu_{sg} < 0.02 \text{ m/s}$. But for a BCR which runs in the heterogeneous flow regime these equations cannot be used to estimate the gas holdup. For this we can use the average gas holdup equation developed by Raimundo based on experimental data from BCR with $D=0.15 \text{ m}$ to 3 m [15].

$$\overline{\alpha_G} = 0.49\nu_{sg}^{0.41} D^{-0.047} \quad (2.6)$$

Where, ν_{sg} again represents the superficial gas velocity and D the bubble column diameter. Raimundo, found the bubble column diameter to have a effect on the average gas holdup, especially in the range of ν_{sg} being 0.1 to 0.2 m/s, but not by much as the exponent indicates. The effect of the diameter is, however, very small. As an alternative, Equation 2.6 can be replaced by an equation involving only ν_{sg} with an exponent of 0.41 and a prefactor of 0.5 instead of 0.49. This modification results in a mean error of 8%.

Equation 2.6 provides a correlation to calculate the average gas holdup for a bubble column operated in the heterogeneous regime. However, to validate CFD models having a correlation describing the axial gas holdup is needed. Due to the chaotic nature of flow within a bubble column, the gas holdup changes constantly, making it difficult, to validate against. Therefore the gas holdup measurements are time averaged, which reveals the expected parabolic gas holdup profile for heterogeneous flow regimes. In 2001 Schweitzer setup a correlation which can predict the axial gas holdup [34]. Schweitzer used a parabolic equation to describe

the time-averaged gas holdup. This correlation is only meant for the stabilized region, meaning it holds for regions at least 1 m above the sparger and one diameter under the liquid with the BCR having at least a aspect ratio (AR) of 5. The axial gas holdup correlation setup by Schweitzer can be seen in equation 2.7:

$$\alpha_G(x) = \overline{\alpha_G}[-1.638(x^6 - 1) + 1.228(x^4 - 1) - 0.939(x^2 - 1)] \quad (2.7)$$

In this equation $x = \left(\frac{r}{R}\right)$, it also shows that if the gas holdup is normalized the axial gas holdup is independent of the column diameter. This normalized axial gas holdup profile can be seen in figure 4.

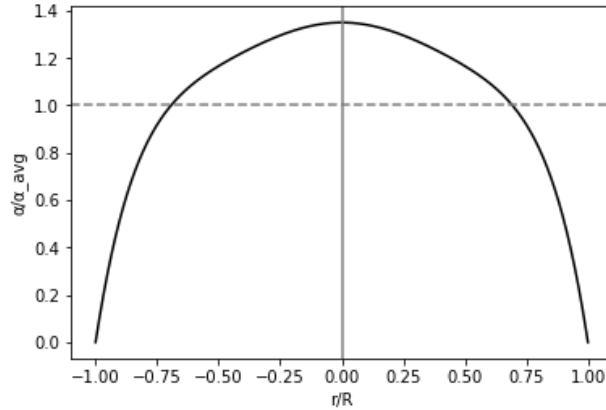


Figure 4: Normalized radial gas hold-up profile calculated using the relation developed by Schweitzer (equation 2.7).

2.2.4 Liquid velocity

The instantaneous liquid circulation in a bubble column can be described as very chaotic and complex, but as was the case with the gas holdup, when the liquid velocity is time averaged the flow pattern proves to be logical. Where, we see an upward flow in the central region and a downward flow near the wall. The transition between these flows lays at $r_{trans} = 0.7R$ [15], this is logical because for this radius the inner and outer circle have the same cross-sectional area. Forret used a polynomial model to describe this logical/stabilized velocity profile of a bubble column, which he said was valid for $D < 1m$:

$$V_L(x) = \frac{V_L(0)}{a - c} [a \exp(-b * x^2) - c] \quad [26] \quad (2.8)$$

With $a = 2.976$, $b=0.943$ and $c = 1.848$. Meaning that if the central liquid velocity is known the axial liquid velocity profile can be setup, this can for example be measured but also calculated using a correlation later setup by Raimundo, which predicted the central velocity with an average error of below 8%. Other than setting up an equation for the central liquid velocity, which can be seen in equation 2.9 Raimundo also found out that equation 2.8 agreed to his data from bubble columns up until $D=3m$, where it would slightly underestimate the liquid velocity near the walls. Figure 5 shows the liquid velocity profile which is to be expected for the reactor configurations, tested by Raimundo [15].

$$V_L(0) = 1.35\nu_{sg}^{0.16} D^{0.4} \quad (2.9)$$

Again, just like with the axial gas holdup, equation 2.8 also shows that if the liquid velocity is normalized by dividing by the central velocity, the liquid velocity profile is independent of the column diameter. The normalized and axial velocity profiles can be seen in figure 5.

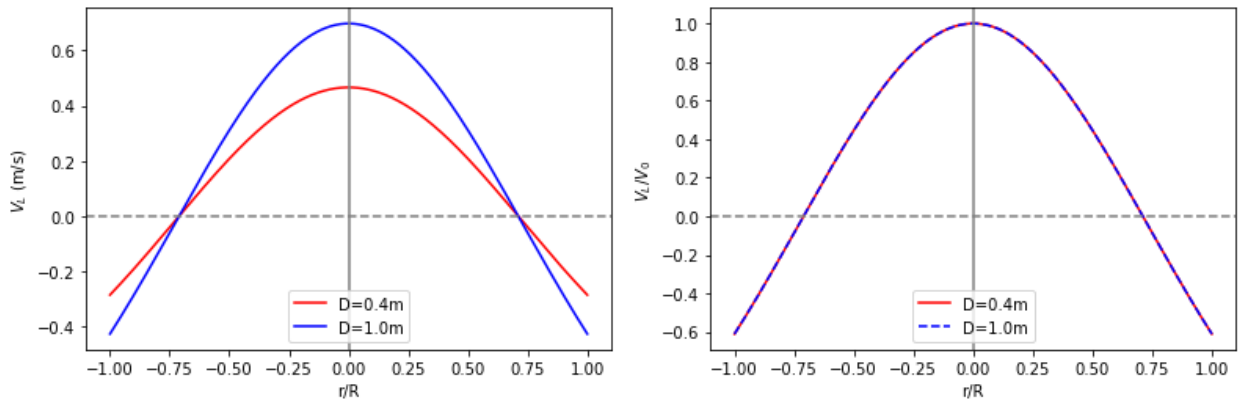


Figure 5: Liquid velocity profile for $D=0.4$ and 1.0 m (Left), Normalized liquid velocity profile for $D=0.4$ and 1.0 m (right), according to equation 2.8

2.3 scale-up

When discussing the scaling up of a reactor, the process involves transitioning from laboratory-scale to full-scale production, with intermediate pilot scales. A new reactor design starts in the laboratory, where small amounts of reactants are used to assess the feasibility and performance of the desired reaction. This phase also allows for evaluating the economic viability of the process without significant investments. Subsequently, the research progresses to larger volumes, where factors like heat transfer and hydrostatic pressure are investigated.

Equations 2.7 and 2.8 demonstrate the independence of normalized gas holdup and liquid velocity from the bubble column diameter. This established relationship provides engineers and researchers with predictive tools to determine the flow regime within large-scale bubble columns, for reactions occurring in mixtures comparable to water and air. In the following paragraph, additional scale-up criteria for achieving independent gas holdup, regardless of the bubble column diameter, are introduced.

2.3.1 Wilkinson scale-up criteria

Regarding the scale-up, a pioneer for the bubble column has been Wilkinson. By conducting experiments in two bubble columns with different diameters (0.15 m and 0.23 m), varying the liquids and combining these results with existing literature, he determined that the average gas holdup is independent of diameter and sparger design if three criteria are met. Meaning, that gas holdup data gathered from smaller scales can be applied for large scale reactors [35].

Criteria 1: The bubble column diameter should be greater than 0.15m.

Criteria 2: The aspect ratio (AR) (Initial liquid height divided by diameter) is greater than 5.

Criteria 3: The sparger openings should be greater than 1-2 mm.

In 2017 Giorgio Besagni, Lorenzo Gallazzini and Fabio Inzoli had a deeper look into criteria number two with respect to the influence of ethanol. Here they challenged this criteria by doing their own experiments with water and ethanol mixtures, from AR 1 to 15, as well as looking at data from literature. It was found that indeed an AR of 5 would guarantee that the gas holdup would not be influenced by the AR, but found this only to be true for water air mixtures [36].

2.4 Computational fluid dynamics and bubble columns

CFD is a very powerful computational tool, used for simulating and analyzing fluid flow behaviour and other related phenomena which can occur in fluid flows. CFD is a numerical approach where it discretizes governing equations such as the Navier-Stokes. CFD solves these equations iteratively over a small volume, accounting for the influence of neighboring cells [37]. The capability of CFD in analyzing and visualizing flow and reactions within a reactor enables engineers and researchers to gain insight into the performance of different kinds of reactors, such as bubble columns. In the rest of this section we introduce what is needed for setting up CFD simulations for bubble columns starting with the governing equations, followed by momentum transfer, turbulence modeling and different iterative scheme's.

2.4.1 Governing equations

The governing equations are the equations describing the conservation laws of mass, momentum and energy. In the case of bubble columns only conservation laws of mass and momentum are taken into account as there is no heat transfer in syngas fermentation. The equation describing these conservation laws are; the continuity and the Navier Stokes equations [38][17].

Continuity equation:

The continuity equation shown in equation 2.10 describes the conservation of mass and the change of mass within a finite volume.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad (2.10)$$

Where ρ is the density of the fluid, t is the time and \mathbf{u} is the velocity vector.

Navier-Stokes Equations:

The Navier-stokes equation described in equation 2.11, describes the conservation of momentum.

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \rho(\mathbf{u} \cdot \nabla) \mathbf{u} = -\nabla p + \nabla \cdot \tau + \rho \mathbf{g} \quad (2.11)$$

With the stress tensor given by:

$$\tau = \mu(\nabla \mathbf{u} + (\nabla \mathbf{u})^T) - \frac{2}{3} \mathbf{I}(\nabla \cdot \mathbf{u}) \quad (2.12)$$

Where P is the pressure, μ is the dynamic viscosity, g is the gravitational acceleration, μ is the molecular viscosity and I being the unit tensor. These equations form a system of partial differential equations that are solved simultaneously to obtain the flow behaviour. To solve these equations Fluent uses a finite volume method to approximate the derivatives of the governing equations and convert them into algebraic equations. From which it can model, flow, pressure, and other relevant parameters [37].

2.4.2 Multiphase Modeling

In the context of bubble column modelling, the presence of two phases makes it that we cannot use equations 2.11 and 2.10, as they are only valid for single phase systems and don't account for the momentum transfer between the phases. Therefore, in order to model multiphase flows in bubble columns, we need equations which take into account each phase and the momentum transfer between them. The Eulerian framework is commonly used for modeling multiphase flows and bubble columns due to its computational efficiency [25], as well as its feasibility at large scales, as it is not realistic to track all individual bubbles in such systems.

This approach solves the governing equations for each phase independently, considering the phase fraction present within a finite volume.

Multiphase Continuity Equation

$$\frac{\partial \alpha_n \rho_n}{\partial t} + \nabla \cdot (\alpha_n \rho_n \mathbf{u}_n) = 0 \quad (2.13)$$

Phase-averaged Navier-Stokes Equation

$$\frac{\partial \alpha_n \rho_n \mathbf{u}_n}{\partial t} + \alpha_n \rho_n (\mathbf{u}_n \nabla) \mathbf{u}_n = -\alpha_n \nabla p + \nabla (\alpha_n \tau) + \alpha_n \rho_n \mathbf{g} + \mathbf{M}_n^{inter} \quad (2.14)$$

The stress tensor is given by:

$$\tau = \alpha_n \mu (\nabla \mathbf{u}_n + (\nabla \mathbf{u}_n)^T - \frac{2}{3} I (\nabla \cdot \mathbf{u})) \quad (2.15)$$

Here, M_n^{inter} accounts for the interfacial forces between the phases. In the case of bubble column simulations, only drag and turbulent dispersion forces will be considered and discussed in subsequent chapters. This choice is motivated by the observation that including additional forces such as lift and added mass did not lead to improved agreement with experimental data [17][39].

2.4.3 Drag force model

In the field of fluid dynamics, understanding and quantifying the drag force acting on bubbles is essential for being able to model a bubble column, where the drag force is held to be most influential interface momentum force [17]. The bubble drag force is calculated using equation 2.16 and includes the drag coefficient (C_D), the bubble diameter (d_b), liquid density and the relative velocity between the two phases. The equation is then also multiplied by the gas volume fraction.

$$F_D = \frac{3}{4} \frac{C_D}{d_b} \rho_L \alpha_G (\mathbf{u}_G - \mathbf{u}_L) |\mathbf{u}_G - \mathbf{u}_L| \quad (2.16)$$

The determination of the drag coefficient for a bubble cannot be carried out using the same method as for a sphere. As explained in Section 2.2.2, bubbles don't always maintain their spherical shape. Only bubbles where $Re < 100$ retain this shape. However, even in those cases, the interface remains mobile, which impacts the drag coefficient. As bubble sizes increase, along with their velocities, the shape of the bubbles become more capillary or elliptical in shape, which impacts the drag coefficient of the bubble.

Two models have been evaluated to calculate the drag coefficient of non-spherical bubbles in this thesis. One of these models was developed by Grace [31], while the other was introduced by Tomiyama[40]. These models consider the non-spherical shape of bubbles and incorporate dimensionless parameters, such as Eo and Mo , to account for the influence of non-spherical shape on the drag coefficient. [31]

Eotvos number

The Eotvos number characterizes the relative importance of gravitational forces to surface tension forces acting on a bubble, and is defined as:

$$Eo = \frac{g(\rho_L - \rho_G)d_b^2}{\sigma} \quad (2.17)$$

The numerator is the gravitational force and the denominator the surface tension force. If the gravitational forces are much greater than the surface tension forces ($Eo \gg 1$) the bubble deforms. If surface tension forces are greater than gravitational forces ($Eo \ll 1$) the bubble are spherical.

Morton number

The Morton number takes into account the physical properties of the phases, described by:

$$Mo = \frac{\mu_L^4 g (\rho_l - \rho_g)}{\rho_l^2 \sigma} \quad (2.18)$$

Grace model

Using these numbers the grace model, calculates the drag coefficients for each shape and then compares them according to the following equation:

$$C_D = \max(\min(C_{D,ellipse}, C_{D,cap}), C_{D,sphere}) \quad (2.19)$$

Where the different drag coefficients are described by:

$$C_{D,sphere} = \begin{cases} \frac{24}{Re}, & Re < 0.01 \\ \frac{24(1+0.15Re^{0.687})}{Re}, & 0.01 \leq Re \end{cases} \quad (2.20)$$

$$C_{D,ellipse} = \frac{4}{3} \frac{gd_b (\rho_L - \rho_G)}{\mu_L^2 \rho_L} \quad (2.21)$$

$$C_{D,cap} = \frac{8}{3} \quad (2.22)$$

$$u_t = \frac{\mu_l}{d_b \rho_L} Mo^{-0.149} (J - 0.857) \quad (2.23)$$

$$J = \begin{cases} 0.94H^{0.757}, & 2 < H \leq 59.3 \\ 3.42H^{0.441}, & 59.3 < H \end{cases} \quad (2.24)$$

$$H = \frac{4}{3} Eo Mo^{-0.149} \left(\frac{\mu_L}{9.10^{-4} Pa \cdot s} \right)^{-0.14} \quad (2.25)$$

Tomiyama model

Tomiyama also accounted for bubble having different shapes by equation 2.26:

$$C_D = \max \left(\min \left(\frac{24 (1 + 0.15Re^{0.687})}{Re}, \frac{72}{Re} \right), \frac{8}{3} \frac{Eo}{Eo + 4} \right) \quad (2.26)$$

2.4.4 Bubble swarms

In heterogeneous flows, bubbles experience a lower drag in the wake of large bubbles, which leads to a higher rise velocity overall. These interactions are a result of a decrease in drag on a trailing bubble. As bubble columns used in the bioprocessing industry operate at high gas fractions to achieve high mass transfer rates, determining the drag reduction is sought to be very important in setting up CFD simulations[17]. To incorporate this reduced drag into models, a drag correction factor is used, and multiplied with the single, isolated bubble drag coefficient, as shown in equation: 2.27:

$$C_{D,swarm} = f(\alpha_G) C_{D,\infty} \quad (2.27)$$

Various attempts have been made in determining the right value for $f(\alpha_G)$. Simonnet setup a volume fraction correction term, and found that experimental data and the model gave reasonable agreement [41]. This equation is given by:

$$f(\alpha_G) = (1 - \alpha_G) \left[(1 - \alpha_G)^m + \left(4.8 \frac{\alpha_G}{1 - \alpha_G} \right)^m \right]^{-2/m} \quad (2.28)$$

This model is found to be valid for a range of bubble diameters, from 5 to 10 mm and a gas volume fraction of 30%. Later a modified form was developed by Fletcher, to account for higher gas fractions. Where he used his own experimental data, to modify the equation of Simonnet[17]:

$$f(\alpha_G) = \begin{cases} 1 & f'(\alpha_G) > 1 \\ 0.8f'(\alpha_G) & f'(\alpha_G) < 1 \end{cases} \quad (2.29)$$

Where $f'(\alpha_G)$ is equal to 2.28. One other drag reduction, which is also implemented in Fluent and commonly used by others [42][17][41] is:

$$f(\alpha_G) = (1 - \alpha_G)^n \quad (2.30)$$

Where it is common to fit the exponent n to the data, as it depends on the superficial gas velocity. Olmos found the range for this exponent to be between 0 and 4 [42]. 'n' typically has a value of 2 or 4 depending on the superficial gas velocity and bubble size [18]. When plotting the different equations as a function of gas holdup we get the following graph, showing the effect of the Simonnet modification and n :

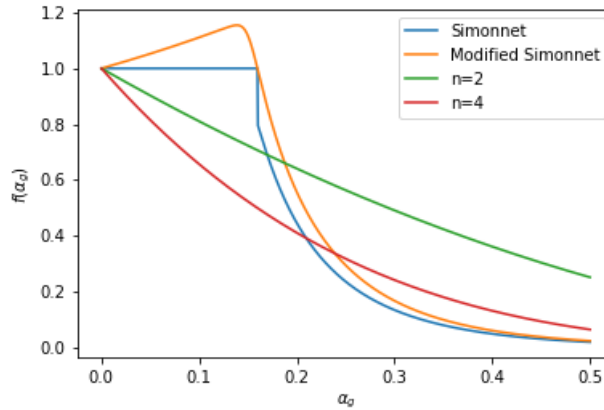


Figure 6: Comparison of different drag modification factors

2.4.5 Turbulence dispersion

One other momentum transfer term which should be accounted for is the turbulent Dispersion force. Dispersion is referred to the spread of particles, or medium. Turbulent dispersion, as the name suggests, is the dispersion force caused by turbulent eddies. The turbulent dispersion force in a bubble column accounts for the turbulence in the liquid phase transporting bubbles from areas of high gas holdup to areas of low gas holdup [18]. Modeling this turbulent dispersion is done by using the Favre-averaged drag relationship developed by Burns[43] which is available in Fluent:

$$F_{TD} = F_D \frac{v_{t,L}}{Sc_{t,L}} \left(\frac{\nabla \alpha_G}{\alpha_G} - \frac{\nabla \alpha_G}{\alpha_g} \right) \quad (2.31)$$

Where F_D is the inter-phase momentum transfer due to drag as described in 2.4.3 and $Sc_{t,L}$ is the turbulent Schmidt number for the bulk phase which has a standard value of 0.9.

2.4.6 Turbulence modeling

Turbulent flows, which can be characterized by chaotic behaviour, require additional modeling to account for the turbulent eddies and their effects on the flow. Different techniques are available to model these turbulent flows, depending on the level of detail required and computational resources available, such as: RANS (Reynolds-Averaged Navier-Stokes), LES (Large eddy simulations) and DNS (Direct Numerical Simulation) [37]. In this section we will look further into the $k - \epsilon$ models (a well known example of a RANS turbulence model). This will be done, by explaining the idea behind the single phase $k - \epsilon$ model and the different forms of this model. Later we build upon this to account for the dispersed phase turbulence and how this effects the $k - \epsilon$ modelling.

K-epsilon

The k-epsilon has several variations, the standard, RNG and realizable models. Their basis is the same, where they model two additional transport equations alongside the governing equations for mass and momentum. These additional equations calculate the turbulent kinetic energy (k) and its dissipation rate (ϵ). These two equations for turbulence allow for the determination of a turbulent length and time scale [37]. The difference between the three types of $k - \epsilon$ models lays in the method of calculating the turbulent viscosity, turbulent Prandtl numbers governing the turbulent diffusion of k and ϵ and the generation and destruction terms. What they all share are the following; generation due to shear buoyancy, accounting for the effects of compressibility, and heat and mass transfer [37]. Additionally, all equations allow for additional source terms to be added, for example for the inclusion of the turbulence generated by a bubble, which will be further discussed in chapter 2.4.8.

Standard $k - \epsilon$ model

The standard $k - \epsilon$ model has become one of the workhorses for practical engineering flow calculations since it was proposed by Launder and Spalding [37][44]. This standard model is seen as the workhorse because it offers a robust model with reasonable accuracy for a wide range of turbulent flows. One of the assumptions for the $k - \epsilon$ model is that we have a fully turbulent flow, and that the effects of molecular viscosity are negligible. These assumptions make it so that the standard $k - \epsilon$ model is only valid for fully turbulent flows. Considering that the Raimundo experiments were carried out in the heterogeneous regime [15], characterized by the presence of a fully turbulent flow, it is reasonable to expect that the utilization of the standard $k - \epsilon$ model would yield suitable outcomes for the bubble column reactor as previously seen by Ertekin and Fletcher [16][17].

RNG $k - \epsilon$

The RNG model is derived using a statistical technique called re-normalization group theory. Compared to the standard model, it includes the following refinements[37]:

- The ϵ equation has an extra term that improves the accuracy for rapidly strained flows.
- It accounts for the effect a swirly flow has on the turbulence, enhancing accuracy for swirling flows.
- Turbulent prandel number are constant for the standard form of the $k - \epsilon$ model, while RNG offers a analytical expression for the turbulent prandel number.
- Effective viscosity is calculated by a analytically derived differential equation, which gives the RNG $k - \epsilon$ model the ability to account for low-Reynolds number effects.

This extra term in the ϵ equation increase the turbulent dissipation rate in the central region, distinguishing it from the standard and realizable $k - \epsilon$ models, and playing a crucial role in accurately simulating rapidly

strained flows [45]. It is suggested that the higher estimated turbulent viscosities by the Standard formulation impede the gas flow in the core region, resulting in a more uniform gas holdup profile [45]. Consequently, the RNG model is expected to better capture the non-uniform gas holdup profile.

2.4.7 Multiphase turbulence model

Having seen how the turbulence is modelled for a single phase, it is easy to imagine the complexity of solving all computationally demanding equations when an extra phase is added to the process. For this reason, Fluent provides three forms of the $k - \epsilon$ model, one of which we utilize, known as the Dispersed turbulence model.

Dispersed turbulence model

This model is valid when the dispersed phase is very dilute. In the case where bubbles or small particles are present in the continuous phase, we can assume that the dispersed phase turbulence has a negligible influence on the continuous phase turbulence, meaning we only need to calculate the turbulence equations for the continuous phase [37]. This model can include the dispersed turbulence by adding extra terms (source terms) to include turbulence caused by the secondary phase, which in our case is the BIT (bubble induced turbulence), see section 2.4.8. The turbulence equations for the continuous phase is then modeled according to equations 2.32 and 2.33.

$$\frac{\partial}{\partial t}(\alpha_L \rho_L k_L) + \nabla(\alpha_L(\rho_L k_L \mathbf{u}_L) - (\mu_L + \frac{\mu_{t,L}}{\sigma_k})) = \alpha_L(P_L - \rho_L \epsilon_L) + T_{LG,k} \quad (2.32)$$

and,

$$\frac{\partial}{\partial t}(\alpha_L \rho_L \epsilon_L) + \nabla(\alpha_L \rho_L k_L \mathbf{u}_L - (\mu_L + \frac{\mu_{t,L}}{\sigma_\epsilon}) \nabla \epsilon_L) = \alpha_L \frac{\epsilon_L}{K_L} (C_1 P_L - C_2 \rho_L \epsilon_L) + T_{LG,\epsilon} \quad (2.33)$$

Where P_L is the turbulence production due to shear. C_1 , C_2 , σ_k and σ_ϵ are constants set to: 1.44, 1.92, 1.0, 1.3. $T_{LG,k}$ $T_{LG,\epsilon}$ are source terms, representing the turbulence generated by the bubbles in the continuous phase. Using the standard turbulent viscosity form used in the standard $k - \epsilon$ model as [37]:

$$\mu_{t,L} = 0.09 \rho_L \left(\frac{k_L^2}{\epsilon_L} \right) \quad (2.34)$$

2.4.8 Bubble induced turbulence

Bubble induced turbulence, as the name suggests, is the turbulence generated by the dispersed bubble phase. Previously we saw that the turbulence could be modelled via a dispersed turbulence model for a diluted system. In its standard form it assumes that there is a minimal impact of the dispersed phase turbulence on the continuous phase, but for the case of bubble columns it was found that, to be able to predict the liquid velocity and gas holdup distributions, accounting for the turbulence generated by the dispersed bubble phase was needed [46]. Pflieger and Becker stated that the turbulent energy produced by the bubbles results from the forces between the continuous and dispersed phase, and the local slip velocity [47], see equation 2.35 and 2.36. Similar approaches were also used by other authors trying to capture the bubble induced turbulence [48] [49].

$$T_{LG,k} = \alpha_L C_3 |M_{LG}| |\mathbf{u}_G - \mathbf{u}_L| \quad (2.35)$$

$$T_{LG,\epsilon} = C_5 \alpha_L C_3 |M_{LG}| |\mathbf{u}_G - \mathbf{u}_L| / \tau \quad (2.36)$$

where, C_1 has a constant value of 1.44, \mathbf{U}_G is the gas phase velocity vector, \mathbf{U}_L is the liquid phase velocity vector, C_3 is a constant (1.0), τ is the time scale dissipation, calculated from the turbulence timescale $\frac{k}{\epsilon}$ and M_{LG} is the inter phase momentum transfer, for which only drag is considered as this is regarded as the most dominant momentum transfer [17][39]. For the dissipation timescale, different models exist, one developed by Troshko and Hassan [50], build into Fluent and one developed by Yao and Morel [51]. Troshko and Hassan, determine the dissipation timescale, τ , based on the eddy residence time:

$$\tau = \left(\frac{2C_{vm}d_b}{3CD|\mathbf{u}_G - \mathbf{u}_L|} \right) \quad (2.37)$$

In the model developed by Yao and Morel, the dissipation timescale, τ , is determined using the length-scale related to the bubble diameter and turbulence eddy dissipation of the bubble :

$$\tau = \left(\frac{d_b^2}{\epsilon} \right)^{\frac{1}{3}} \quad (2.38)$$

2.4.9 Non-iterative time advancement scheme (NITA)

Digging deeper into the iterative part of CFD simulations, it is important to start by taking a closer look at the "standard" iterative scheme: Fluent solves the governing equations in an iterative way, called the iterative time-advancement (ITA) scheme. In this scheme, at every time step, all equations are solved iterative, until convergence is achieved. In this scheme all the equations account for each other, eliminating the splitting error. This requires a considerable amount of computational power because all governing equations have to be solved each iteration, meaning that, if the turbulence isn't impacted by the new iteration (convergence met), it is again calculated. The idea behind the NITA scheme is that you don't need to reduce the splitting error to zero, you only need to make it as the same order as the truncation error (Error between the exact and discrete solution) [37][52]. An overview of both scheme's can be found in appendix B.

3 Model development

The experimental configuration which will be replicated in Fluent is based on the work done by Raimundo [15]. As a brief recap on why we selected this configuration as our model basis: validating CFD models is crucial in determining their ability to reproduce real-world experiments. Raimundo acknowledged a gap in the availability of such data and provided these valuable measurements for a range of bubble column diameters, including gas holdup, liquid velocity and bubble diameter. Furthermore, this reactor has already been successfully simulated in Fluent [16] using a model developed by Fletcher [17], given an excellent basis on which our model can be based. In the following sections, we will introduce the various reactor configurations, discuss the model settings, assumptions and how the variables are measured.

3.1 Model geometry

Four different reactors have been simulated, with diameters of 0.15m, 0.4m, 1m and 3m. All reactors operate in batch mode, meaning there is no liquid inflow or outflow. The heights of the 1m and 3m diameter reactors were provided as 6.5m and 14m[15]. The heights of the remaining reactors were determined using the same height-to-diameter ratio as the 1m diameter reactor, which was 6.5. By multiplying the column diameter of the 0.15m and 0.4m with this ratio and rounding up, we were able to calculate the heights of these reactors as can be seen in table 4. With these dimensions we were able to build the geometry using *ANSYS Design Modeler 2021 R2*. A representation of the D=1.0m reactor can be found in appendix C. *ANSYS Design Modeler 2021 R2* was also used to define the boundaries, specifically the *mass flow inlet* and *pressure outlet* boundary conditions.

Diameter (m)	H_{BCR} (m)
0,15	1
0,4	3
1	6,5 [15]
3	14 [15]

Table 4: Reactor dimensions used for simulation.

3.2 Meshing

ANSYS Fluent Meshing has been used for setting up the fluid domain mesh. Each geometry contained, on average, 100.000 Poly-Hexcore elements. This value was chosen as a trade-off between accuracy and computational requirements. Previous mesh independence studies had shown that going coarser than 100.000 influenced the results, while going to a finer grid did not improve the results either [18][16]. A finer grid was also build to do our own mesh in dependency study, for this we used a mesh containing 200.000 elements. For the exact number of cells, and the cell sizes see table 5.

The Poly-hexcore mesh utilizes mosaic meshing, where two cell types are connected with Polyhedral elements [53]. In our case, it connected the outer poly-prism mesh with a hexahedral mesh in the core of the reactor. These two types of elements can be seen in figure 7. Hexahedral elements allow for accurate results and require less computational demand compared to other elements, such as polyhedral elements. This is because hexahedral elements, on average, have fewer faces than polyhedral elements, reducing computational load and memory requirements[53]. However, these elements are not well-suited for curved geometries. In such cases, the poly-prism mesh is used to mesh these curved areas, as it is better suited for those areas [53]. The resulting fluid domain mesh for the D=0.4m and D=1m reactors can be seen in Appendix D . All the settings for meshing can be found in table 6. If a parameter is not described, it is set to its default value.

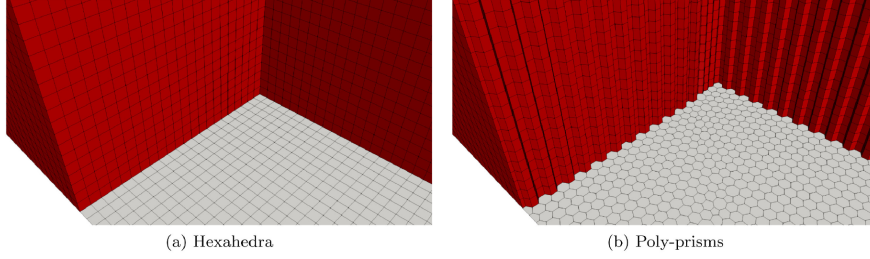


Figure 7: Elements used for meshing; Hexahedral (left) and Poly-prisms (right)

Meshing	Diameter (m)	Surface mesh		Volume mesh		
		Cells	Min size (m)	max size (m)	min lenght (m)	max lenght (m)
	0,15	99271	0,003	0,006	0,003	0,006
	0,4	118577	0,0125	0,025	0,0075	0,015
		200318	0,0065	0,013	0,0065	0,013
	1	96752	0,02	0,04	0,02	0,04
		207651	0,015	0,03	0,015	0,03
	3	116721	0,05	0,1	0,05	0,1

Table 5: Number of cells per geometry and the minimum and maximum cell sizes

Surface mesh	
Growth rate	1,2
size functions	curvature & proximity
Curvature normal angle	18
cells per gap	1
Add boudary layers	
location	Sparger
offset method type	Smooth-transition
number of layers	4
transition ratio	0,272
Growth rate	1,2
Volume mesh	
Fill with	poly-hexcore
Buffer layers	2
peal layers	2

Table 6: Mesh parameters used for the meshing of all columns

3.3 CFD model

In this section we present the models used in the simulations and their respective parameters, along with the simulation strategy and the data collection. Starting with discussing the models, followed by the assumptions, materials, and boundary conditions. Subsequently, explaining the data collection method, and conclude by providing an overview of the simulation strategy.

3.3.1 Momentum transfer

The developed models in this study make use of the Eulerian framework, which is known for its computational efficiency in solving multiphase flows [25]. This framework was recommended by Fletcher and utilized by Ertekin when simulating bubble columns from Raimundo [17][16]. Regarding momentum transfer, only drag and turbulent dispersion forces were considered, with the turbulent dispersion force modelled using the Burns model (equation 2.31).

For the drag force, two models, namely the Tomiyama and Grace drag models, were used, as a comparison will be performed to evaluate their performance. Both models are well-suited for gas-liquid flows in which bubbles can have a range of shapes [37]. For the drag force, a constant bubble size of 5.1 mm was assumed, with the diameter being based on the average bubble diameter measured by Raimundo [15]. Previous studies of water-air bubble column reactors have reported a range of bubble diameter between 3 to 13 mm [54], for which the bubble terminal velocity is approximately constant ($0.24 \pm 0.01 \text{ m/s}$) [55]. For that reason it is stated that the approximation for using a single bubble size is reasonable, given that using a single bubble size greatly reduces model complexity [54].

A drag reduction factor was used, to account for the bubble swarm effect. A value of 0.12 was chosen based on a average gas holdup of 30%, and using equation 2.29. In addition to that, the drag reduction model implemented in Fluent was used with $n=4$ (equation: 2.30), and the drag reduction model developed by Simonnet modified by Fletcher was used (equation: 2.29). As the latter was not available in Fluent as a standard model, a UDF was implemented. The details of how this was achieved can be found in subsection 3.4.1.

3.3.2 Turbulence model

The turbulence in this study was modelled using the widely adopted and computationally efficient standard $k - \epsilon$ model, due to the fully turbulent flow regime present in the reactor [37][16]. In addition to the standard turbulence model, the RNG $k - \epsilon$ model was also used to provide a comparison. It should be noted that the use of the RNG model was not strictly necessary, as there are no swirls present (time-averaged tangential velocity is close to zero [56]). The model constants for both of the turbulence models, can be seen in table 7.

In terms of the gas phase turbulence, it is modelled using the dispersed turbulence model, as other models resulted in convergence issues. This model assumes that the gas phase has a negligible influence on the liquid phase turbulence. The bubbles however do add turbulence to the liquid phase as Fletcher found out [17]. He saw that the addition of a BIT term was needed to comply CFD data with experimental data. To account for this, a source term has been added to the turbulence equation of the liquid phase.

Viscous model		
Model	$k - \epsilon$	$k - \epsilon$
$k - \epsilon$	Standard	RNG
Near-wall treatments	Standard wall functions	Standard wall functions
Model constants		
C _{mu}	0.09	0.0845
C ₁	1.44	1.42
C ₂	1.92	1.68
C _ε	1.3	1.3
TKE prandtl Number	1	-
TDR Prandtl Number	1.3	-
Dispersion Prandtl Number	0.75	0.75

Table 7: $k - \epsilon$ turbulence model settings

3.4 User defined function (UDF)

In Fluent, UDF's are custom programs that can be implemented to extend the capabilities of the simulation software. These programs are written in C++ using *Visual Studio Code* 1.78.2 and are able to access Ansys Fluent solver data [57]. These program's are then integrated into Fluent. This UDF capability in Fluent has been used to model the drag reduction factor due to the presence of bubble swarms, as well as the BIT. The *Visual Studio Code* codes for the thes UDF's is provided in Appendix E.

3.4.1 Swarm modification

The modified Simonnet drag reduction factor, developed by Simonnet and Fletcher (equation 2.29) had been implemented as UDF into Fluent. The modified version of the Simonnet model was implemented since the gas holdup exceeded the valid range ($\alpha_G > 0.3$) for the Simonnet drag reduction model. For stability reasons, the model assumed a minimum α_G of 0.01. This was implemented because when α_G is equal to 0, the definition of Simonnet results in a value equal to 1. Since Fletchers modification only specifies whether Simonnet is greater or smaller than 1, and the code only checks if the value is greater than 1, a value equal to 1 results in a drag reduction factor of 0.8, even when no gas is present. How this minimum function affected the drag reduction factor is shown in figure 8.

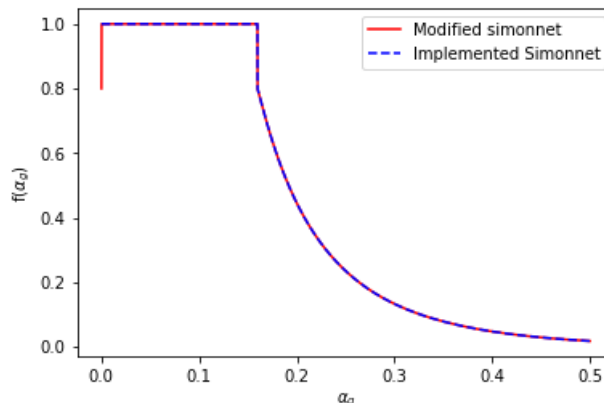


Figure 8: Illustrating the impact of considering a minimum gas holdup on the model output.

3.4.2 BIT model

The turbulence generated by the bubbles is accounted for by the Yao and Morel BIT model (see source [51]). This model was implemented in fluent as a UDF and then added as source term to the equations for turbulent kinetic energy (k) and turbulent dissipation (ϵ) of the liquid phase. When working with computationally generated data, it is possible to obtain non-physical outputs, which can subsequently lead to unrealistic and unstable UDF outputs, as was the case for the implementation of this model. To address this issue, limits were implemented into the both source term UDF's.

The slip velocity, which represents the relative velocity between the gas and liquid phases, was chosen to be limited due to its physical significance. The limit was determined based the maximum rise velocity of a single bubble and the maximum downward velocity. Terminal rise velocities reported for single bubbles with a diameter ranging from 3 to 13 mm were found to be within the range of $(0.24 \pm 0.01 \text{ m/s})$ [54]. Considering that the maximum downward velocity of the liquid phase was $\pm 1 \text{ m/s}$ [15], it can be assumed that the maximum relative velocity should be in the range of 1.24 m/s. Rounding up to 2 m/s, accounting for the increased rise velocity of a bubble in a swarm. This value initially resulted in unstable simulations. Therefore, the limit was raised to 5 m/s for both source terms, which provided stable simulations.

To visualize the impact of this slip limit on the output, a three-dimensional graph was created of the turbulent kinetic energy output to show the dependence of the source term on gas holdup and slip velocity and the influence of the slip limit (see Figure 9). The graphs demonstrate that by enforcing this slip limit, the maximum output is limited to $2.311 \times 10^7 \frac{\text{kg}}{\text{m}^3\text{s}^3}$ instead of $1.849 \times 10^8 \frac{\text{kg}}{\text{m}^3\text{s}^3}$, resulting in a decrease in turbulent kinetic energy by a factor of 8.

By implementing this limit, the non-physical outputs that could potentially affect the accuracy and reliability of the model are kept within acceptable limits, resulting in a more robust and physically meaningful simulation.

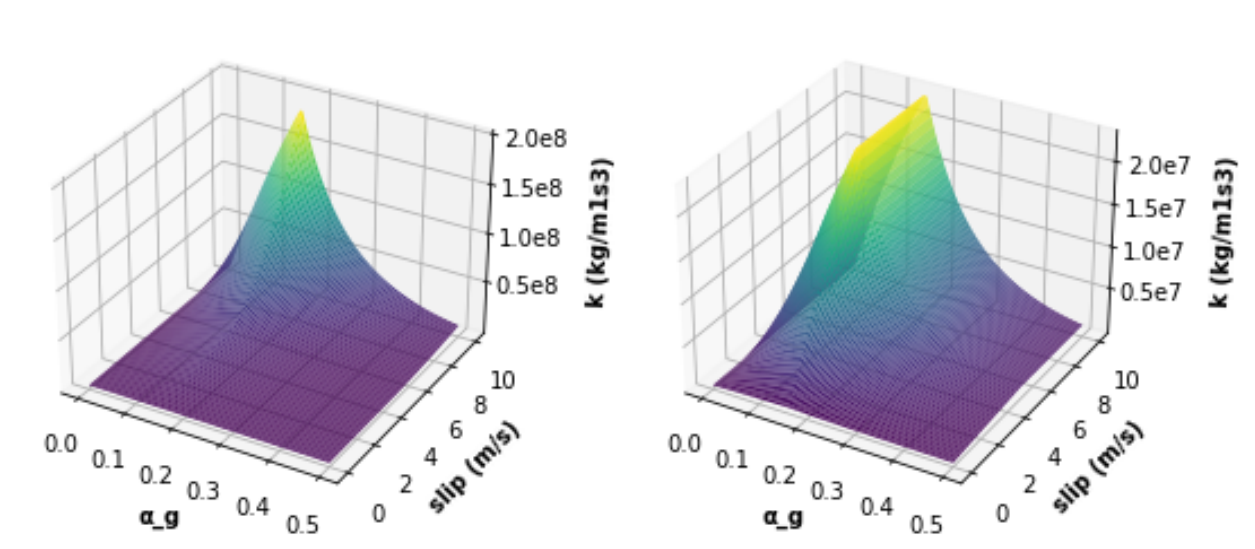


Figure 9: Kinetic energy model output; Output without slip limit (left), output with slip limit (right)

If we examine the equation for turbulent energy dissipation, equation 2.36, it becomes clear that the calculation of the energy dissipation time scale used by Yao and Morel (equation 2.38), relies on the turbulent eddy dissipation itself, leading to a non-linear relationship. Consequently, an iterative solving method is required. To achieve this, the value of the energy dissipation is saved after each inner and outer iteration using the User Defined Memory (UDMI) function in Fluent. This iterative process allows us to obtain the

correct value for the turbulent eddy dissipation.

To validate this method, different initial values were tested to assess their influence on the model output, as well as the number of iterations required to converge. In figure 10, three different initial values ($1e-9$, $1e10$, and $1e11$) were used, and the resulting energy dissipation was calculated for a slip velocity of 2 m/s and a gas holdup of 0.3. The graph clearly shows that the model output is independent of the initial value, and after eight iterations, all value stabilized. To start this iterative process a initial value of $1e-9$ had been assumed. To ensure the turbulent energy dissipation can reach a constant value, it is necessary to setup the solvers to perform at least ten iterations per time step.

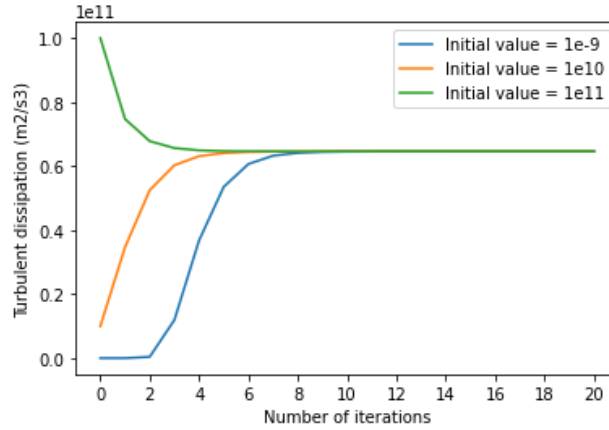


Figure 10: Validation of the number of iterations needed and the influence of the initial value. Results obtained with a gas holdup of 30% and a slip velocity of 2 m/s

3.5 Assumptions, materials and boundary conditions

3.5.1 Assumptions

In order to run the simulation several assumptions have been made:

- Constant operating temperature of 293.15K and pressure of 101325 Pa.
- Constant bubble diameter of 5.1 mm.
- No slip at the wall

3.5.2 Materials

The bubble column reactor was simulated with water and air. The properties of these components are presented in the table shown below.

	Water	Air	Unit
Density	998.02	1.204	kg/m^3
Molecular weight	18.02	28.966	g/mol
Viscosity	0.001	1.79E-05	$Kg/(m s)$

Table 8: Properties of materials used in simulations

3.5.3 Boundary conditions

There are three boundary conditions that had to be defined for the modelling of the bubble columns: one at the wall, and one at each inlet and outlet. At the wall, a *noslip* boundary condition was applied to both phases. The outlet boundary condition was set as *pressure outlet*, while the inlet was set as *mass flow inlet* boundary condition. The gas flow inlet at the inlet was calculated using the superficial gas velocity, gas density, and the area of the inlet (equation 3.1). As observed in Table 9, it is apparent that each geometry in the experiments had a distinct superficial gas velocity for which the experimental paper by Raimundo does not provide any justification.

It is important to consider the increase in air density caused by the hydrostatic pressure of the liquid. By using the ideal gas law (equation 3.2) and equation 3.1, the mass flow at the inlet could be calculated while considering the increased air density. All relevant data, including the final mass flow for each configuration, is shown in table 9.

$$Q_m = v_{gs} A_{sparger} \rho_{sparger} \quad (3.1)$$

$$\rho_{sparger} = \frac{P_{sparger} M_{air}}{RT} \quad (3.2)$$

Where v_{gs} represents the superficial gas velocity, $A_{sparger}$ the sparger area, $\rho_{sparger}$ air density at the sparger, $P_{sparger}$ refers to the pressure at the sparger, M_{air} is the molecular weight of air, R is the ideal gas constant (8.314 J/(Mol K)) and T the reference temperature.

Diameter (m)	H_0 (m)	v_{gs} (m/s) [15]	$P_{sparger}$ (bar)	$\rho_{sparger}$ (kg/m ³)	Q_m (kg/sec)
0.15	0.6	0.15	1.072	1.253	0.003
0.4	1.6	0.16	1.170	1.367	0.027
1	4	0.16	1.405	1.642	0.206
3	9	0.2	1.894	2.214	3.130

Table 9: Relevant data needed for the mass flow calculations

3.5.4 Solution

Discretization schemes

In CFD simulations, the governing equations are solved by discretization, and then solving them iteratively for each cell while considering the neighboring cells. ANSYS Fluent computes and stores these values at the cell centers. However, to account for convection terms, face values are necessary and need to be interpolated from the cell center values. Various techniques are available for interpolating these face values, including the upwind scheme, which obtains its value from the upstream cell [37]. ANSYS Fluent provides a range of upwind schemes, such as first-order upwind, second-order upwind, and QUICK. The base model's discretization, as indicated in Table 10, primarily uses first-order upwind discretization to ensure model stability. However, for achieving a more accurate solution, second-order discretization will also be employed for the momentum and volume fraction equations in chapter 4.12.

Solution methods	
Pressure-velocity coupling	
scheme	Phase Coupled SIMPLE
Spatial discretization	
Gradient	Least Squares Cell Based
Density	First Order Upwind
Pressure	Second Order Upwind
Momentum	First Order Upwind
Volume fraction	First Order Upwind
Turbulent Kinetic Energy	First Order Upwind
Tubulent dissipation rate	First Order Upwind

Table 10: Discretization schemes used in the multiphase model for simulating the Raimundo experimental setup.

Under-relaxation factors

Under relaxation factors are mostly applied to solution variables such as velocity, pressure, or temperature in order to enhance the convergence behavior of iterative solvers. The objective is to dampen the magnitude of the updated solution of these variables. This damping is necessary due to the non-linearity of the equations being solved [37]. By controlling the rate of change of the variables through under relaxation factors, a stable simulation can be achieved. The under relaxation factors for this work were determined by testing different values until a stable solution was obtained. Subsequently, these factors remained unchanged for subsequent simulations and are outlined in table 11.

Solution controls	
Pressure	0.5
Denisty	0.5
Body Forces	0.5
Momentum	0.7
Volume Fraction	0.5
Turbulent kinetic energy	0.5
Tubulent dissipation rate	0.5

Table 11: Under-relaxation factors as defined in Solution controls used in the multiphase model for simulating the Raimundo experimental setup

Time stepping

The time step plays a crucial role in simulating the behavior of fluid flow over time, as it directly affects the stability of the solutions. Smaller time steps generally lead to more stable solutions but take more time to run the simulation. Using a larger time step can result in unstable simulations, if the chosen step size is too large to accurately capture the flow features [58].

One important concept in determining the time step is the Courant number. The Courant number is a dimensionless parameter that relates the size of the time step to the spatial resolution and the velocity of the fluid [58]. It represents the maximum distance a fluid particle can travel in one time step and is calculated using equation 3.3. As a general rule, the time step should be smaller than the time it takes for a fluid particle to travel through a cell [58], implying a Courant number less than one.

$$CFL = \frac{V_{max}\Delta t}{\Delta x} \quad (3.3)$$

For all different reactors the same time step was assumed, as the goal was to have a model able to simulate all geometries. As in the paper by Ertekin [16], a time step of 0.001s was taken, and would result in a Courant number smaller than one, taking the maximum expected velocity of 5 m/s. Except for $D=0.15\text{m}$, based on its minimum cell length of 0.003m the minimum time step for Courant to be 1 should be 0.0006, therefore it was arbitrarily chosen to use a time step of 0.0001s.

In order to maintain a model capable of simulating various reactor geometries, a uniform time step was adopted across all reactor configurations. In the study by Ertekin [16], a time step of 0.001s was employed, resulting in a Courant number below one when considering the maximum anticipated fluid velocity of 5 m/s (the maximum slip velocity employed for the BIT model). However, for the specific case of $D=0.15\text{m}$, where the minimum cell length is determined to be 0.003m, a minimum time step of 0.0006s is necessary to satisfy a Courant number of 1. Consequently, a time step of 0.0001s was arbitrarily selected for this particular scenario.

Simulation procedure

As the goal is to develop one single model able to capture the hydrodynamics of all reactors, each simulation has the same simulation procedure. Where the first second would be run without the drag modification factor, after which it would be enabled and run for a total of 30 seconds, before measuring the gas holdup and liquid velocity. It had been assumed that when the first bubble reaches the surface, a stabilized flow has been reached. Assuming a terminal rise velocity of a bubble in a bubble column is approximately 0.24 m/s [55], for the bubble to reach the surface of the highest reactor is approximately 27.5 seconds, looking at the gas holdup profile from start to 30 seconds, it can be seen that from second 10 till 15 (figure 11), already a stable gas holdup is measured, which supports our assumption that after 30 seconds a stable simulation is reached, after which the gas holdup and liquid velocity can be measured for 60 seconds. 60, had been chosen as the time average values didn't change anymore.

In order to develop a comprehensive model capable of accurately capturing the hydrodynamics of diverse reactor configurations, a standardized simulation procedure was employed for each case. The procedure entailed an initial simulation period of one second without the inclusion of the drag modification factor, followed by enabling the the drag modification and running the simulation for a total duration of 30 seconds of flow-time. It was assumed that a state of stabilized flow is achieved once the first bubbles reach the reactor surface. Based on literature estimates of a bubble's terminal rise velocity in a bubble column being approximately 0.24 m/s [55], it can be assumed that the time required for the bubble to ascend to the surface of the tallest reactor ($H_0 = 6.6\text{m}$) is approximately 27.5 seconds. Analyzing the gas holdup profile from the start to 30 seconds for the largest reactor, it becomes evident that a stable gas holdup is already observed between seconds 10 and 15 (see Figure 11). This observation supports our assumption that a stable simulation is achieved after 27.5 seconds. Following this stabilization, gas holdup and liquid velocity were measured for an additional 60 seconds, as it was determined that time-averaged values did not exhibit further changes. When higher order discretization schemes, as described in Chapter 4.12, were enabled, the simulation initially ran for 20 seconds of flow-time to reach a state of stable gas holdup (Figure 11). Following this, the higher order discretization was activated and the simulation continued until 30 seconds of flow-time had elapsed, after which time-average data was measured.

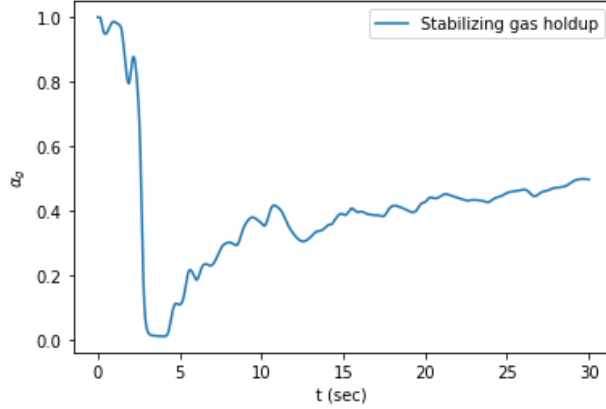


Figure 11: Measured average gas holdup over time for $D=3.0m$ obtained using the improved model settings.

3.6 Data gathering and data processing

Before going into the measured variables and methodologies used, it is important to first established the locations of the sampling points. The data collection locations were consistent with those used by Raimundo [15], who obtained data at four different (H/D) ratios: 0.5, 1.25, 2.5 and 3.75. These locations can be seen in figure 14. At each height, a total of 60 data points were collected, distributed evenly across four planes, with each plane containing 15 data points and separated by 45° from one another. At each of these points, the gas holdup, axial velocity and slip velocity were measured and provide us. All planes are then combined, providing four data points for each radial location. Figure 12 gives a representation of each plane and how the average plane compares to them.

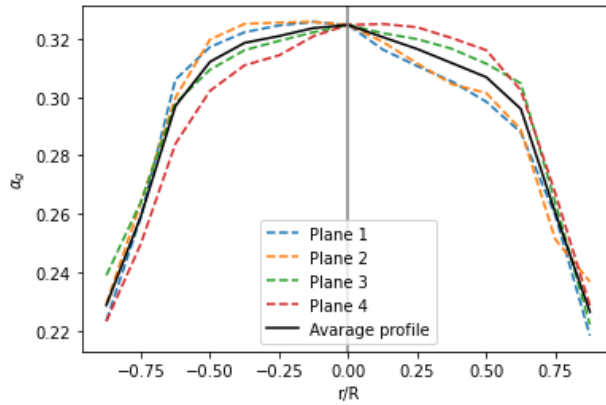


Figure 12: Comparison of gas holdup profiles for each plane with the average gas holdup profile. Results obtained with base model and NITA solver for $D=0.4m$

Considering the transient nature of bubble column simulations and the dynamic flow patterns seen in these systems, instantaneous data alone does not provide the level of detail required for analyzing gas holdup and liquid velocity. To generate the expected parabolic profiles of gas holdup and liquid velocity, time-averaging of the data was necessary. To achieve this, the *Welford's algorithm* [59] was used. Equation 3.4 (time-averaging) was implemented into Fluent. An overview of the data gathering process can be seen in Figure 13. It involved achieving a stable simulation first, followed by time-averaging the measured gas holdup.

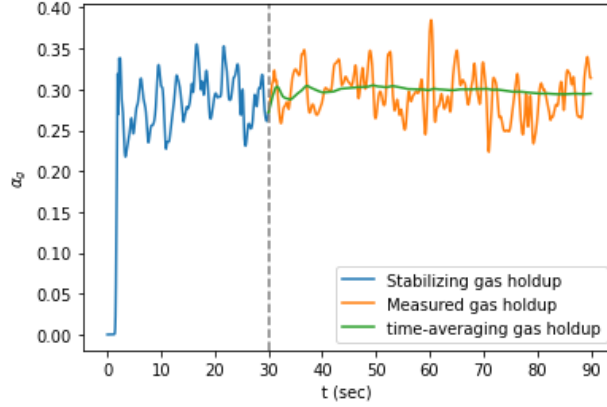


Figure 13: The average gas holdup over time for a $D=0.4m$ with time averaged gas holdup using the base model settings.

$$\bar{x}_n = \frac{(n-1)\bar{x}_{n-1} + x_n}{n} \quad (3.4)$$

Where \bar{x} represents the time-averaged value and n stands for the current time step.

The implementation of these equations was done using a UDF, which was implemented to perform the calculations at the end of each time step. These variables were then stored in a UDM and exported as a *.txt* file for the analysis. The UDF responsible for the time-averaging process can be found in appendix E. The python code used for the data processing and putting in the measuring point in the reactor are provided in appendix F.

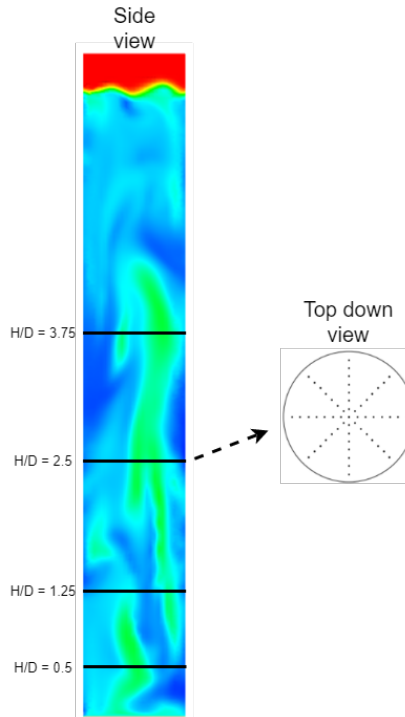


Figure 14: Schematic view of the 1m in diameter column with the sampling locations

3.7 Validation

Validation plays a crucial role in obtaining meaningful data from CFD simulations. Consequently, it was decided to simulate the experimental setup by Raimundo, as his work was specifically aimed at providing validation data for bubble columns [15]. The validation process is performed by computing the rolling average gas holdup and liquid velocity, as explained in 3.6, using equation 3.4. These values are then compared with the gas holdup and liquid velocity correlations (equation 2.7 and 2.8), that are consistent with the experimental measurements of Raimundo [15].

To assess the agreement between the simulation and the correlation, two comparisons were performed. First, the average gas holdup and liquid velocity from the simulation were compared to the corresponding values from the correlation. Second, the individual points of the simulated gas holdup and liquid velocity were compared to the correlation using root mean square error (RMSE) as a measure of deviation.

When comparing the average gas holdup, it is crucial to consider a specific measurement range, namely between $\frac{z}{R} = -0.875$ and 0.875 . This range was selected because the simulation measurements of gas holdup were available exclusively within this interval. However, since a simulation is not limited to a specific range, it is recommended to measure the data at a range of $\frac{z}{R} = -1$ to 1 , as discussed in Chapter 6. Taking this limited range into account is necessary to prevent underestimation of the gas holdup average when comparing it with the correlation. The correlation considers the entire range, which includes lower gas holdups outside of the specified range. By focusing on the measured range, the comparison provides a more accurate assessment of the average gas holdup, ensuring that the correlation's estimation aligns with the relevant simulation data.

The comparison between each data point and the the correlation results has been performed using the RMSE to quantitatively assess the impact of additional models on the simulation results, as seen in figure 3.5. The RMSE measures the average deviation between the observed values and the true (correlation) values.

$$RMSE = \sqrt{\sum \frac{(x_{observed} - x_{true})^2}{N}} \quad (3.5)$$

Later in chapter 4.10, a more detailed comparison is made between the experimental results and the model outcomes to assess the model's ability to replicate the experimental results. This comparison extends beyond comparing the RMSE and average gas holdup.

It also examines the model's ability to reproduce the measurement height independence of the stabilized region and the scale-independence of the model. By taking into account these additional aspects, a more comprehensive validation of the model's performance can be established.

3.8 Simulation strategy

In CFD simulations, it is widely known that the incorporation of different models can give rise to instabilities. To address this, the modeling approach initially starts in its minimal form (base model), as outlined in Table 12, in order to mitigate and recognize potential instabilities. The model will then be gradually expanded by incorporating different models to assess their influence on the gas and liquid flow fields. Once the improved model is established, a comprehensive analysis is conducted to assess whether the model is able to recreate the scale-independence and measuring height independence seen in the experimental setup of Raimundo [15], as well as compare the results with those achieved by Ertekin [16]. Furthermore, attempts are made to optimize the improved model further by using higher discretization schemes as well as a mesh independancy study. A visual representation of the strategy going from the base model to the improved model, including the tested models at each step, can be seen in Figure 15. The steps to achieve the improved design have only been applied for the bubble columns with diameters of 0.4 m and 1.0 m.

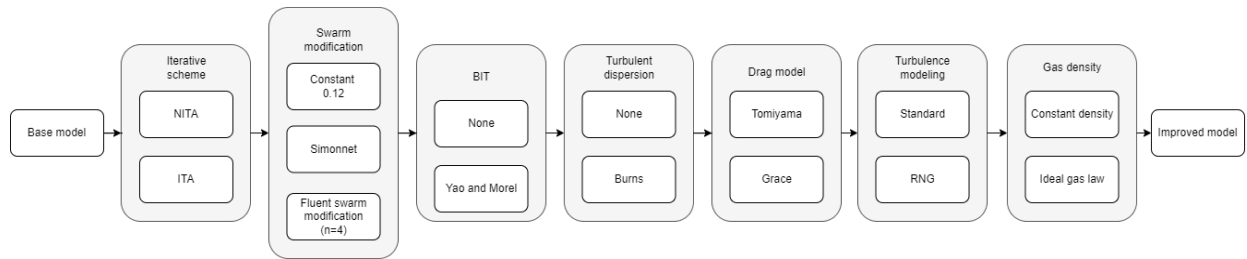


Figure 15: Step-wise flow chart, displaying how the model is going to expand each step

Solver	ITA
Drag model	Tomiyaama
K-eps model	Standard
Dispersed phase turbulence	Dispersed
Swarm modification	0.12

Table 12: Simulation settings for the base case model

3.9 Model summary

This chapter provides a summary of all the model settings employed to simulate the experimental reactor geometries from Raimundo.

parameter	symbol	D=0.15	D=0.4	D=1.0	D=3.0	unit
Geometry parameters reactor						
Reactor diameter	D	0.15	0.4	1	3.0	<i>m</i>
Reactor diameter	A	0.02	0.13	0.79	7.07	<i>m</i> ²
Meshing						
Surface mesh						
100k min cell size	Δx	0.003	0.0125	0.02	0.05	<i>m</i>
100k max cell size	Δx	0.006	0.025	0.04	0.1	<i>m</i>
200k min cell size	Δx	-	0.00665	0.015	-	<i>m</i>
200k max cell size	Δx	-	0.013	0.03	-	<i>m</i>
Growth rate	-	1.2				-
size functions	-	Curvature & proximity				-
curvature normal angle	-	18°				-
cells per gap	-	1				-
Volume mesh						
100k min cell size	Δx	0.003	0.0075	0.02	0.05	<i>m</i>
100k max cell size	Δx	0.006	0.015	0.04	0.1	<i>m</i>
200k min cell size	Δx	-	0.00665	0.015	-	<i>m</i>
200k max cell size	Δx	-	0.013	0.03	-	<i>m</i>
fill with	-	Poly-hexcore				-
buffer layers	-	2				-
peal layers	-	2				-
Meshing boudary layers						
Loction	-	Sparger				-
offset method type	-	Smooth-transition				-
number of layers	-	4				-
transition ratio	-	0.272				-
Growth rate	-	1.2				-
Mesh size						
100k	-	99271	118577	96752	116721	Cells
200k	-	-	200318	207651	-	Cells
Computational resources						
Required CPUs		24	24	24	24	No. CPU's

Table 13: Summary of model settings for each of the reactor geometries

parameter	symbol	D=0.15	D=0.4	D=1.0	D=3.0	unit
Operating conditions						
Temperature	-	293.15				K
Headspace pressure	$P_{Headspace}$	101325				Pa
Bottom pressure	$P_{sparger}$	107199	116990	140487	165943	Pa
Initial liquid height	H_0	0.6	1.6	4	6.6	m
Gas phase properties - air						
Density base model	ρ_L	1.204				kg/m^3
Density improved model	ρ_L	Ideal gas law				kg/m^3
Gas viscosity	μ_L	1.79E-05				$kg/(m\ s)$
Superficial gas velocity	V_{sg}	0.15	0.16	0.16	0.2	m/s
Mass gas flow inlet	\dot{m}_G	0.003	0.027	0.206	2.741	kg/s
Bubble diameter	d_b	5.1				mm
Inlet mole fraction						
Nitrogen	N_2	0.79				mol_{N_2}/mol_G
Oxygen	O_2	0.21				mol_{O_2}/mol_G
Liquid phase properties - water						
Liquid density	ρ_L	998.02				kg/m^3
Liquid viscosity	μ_G	0.001				$kg/(m\ s)$
Dispersion properties						
Surface tension	σ	0.072				N/m^2

Table 14: Summary of model settings for each of the reactor geometries (continuation)

4 Results and discussion

This chapter presents the results obtained from the various CFD models developed throughout this study. Sections 4.1 to 4.8 outline the steps taken to enhance the base model. In Section 4.9, a comparison is made between the base model and the improved model, followed by a comprehensive analysis delving into scale-independence and measuring height independence. Subsequently, a mesh dependency study is conducted, and higher-order discretization models are implemented. Finally, a detailed comparison is done between the model developed by Ertekin and the improved model. A full overview of each model, its stability and run number are presented in Appendix G.

4.1 Base model

As stated in section 3.8, a base model was initially established according to the settings outlined in table 12. Firstly the performance of this model was assessed by comparing the simulation results to the experimental correlation profiles for liquid velocity and gas holdup (equations 2.7 and 2.8). The simulation results, including RMSE, average gas holdup and central velocity value's, are presented in table 15. The resulting gas holdup and velocity profile graphs can be seen in figure 16.

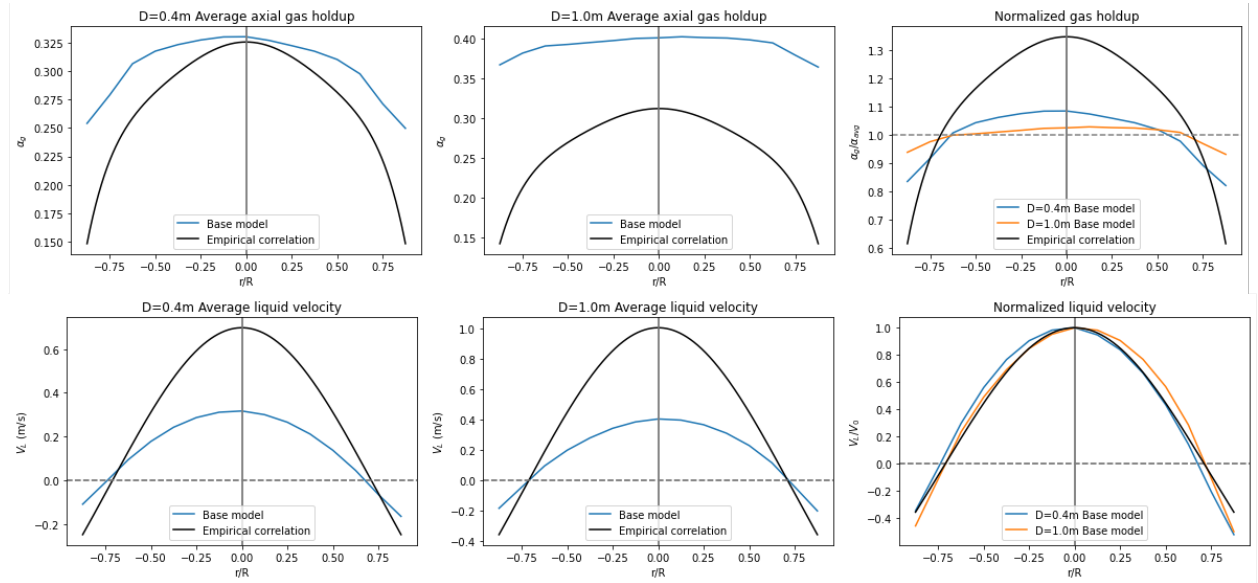


Figure 16: Base model gas holdup (top) and liquid velocity (bottom) profiles for $D=0.4m$ (left), $D=1.0m$ (middle), and non-dimensionalized (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4	26.7%	30.7%	0.048	0.698	0.317	0.233
D1.0	25.6%	39.1%	0.141	1.007	0.405	0.366

Table 15: Average gas holdup, liquid velocities and RMSE obtained from the base model simulations for two different diameters ($D=0.4m$ and $D=1.0m$).

From figure 16, we observe that the model shows the anticipated parabolic shape for the gas holdup and liquid velocity. However, when examining the gas holdup profile, particularly for $D=1.0m$, it appears to remain

relatively flat. This flat profile becomes more apparent when considering the normalized gas holdup profile. Furthermore, the model tends to overestimate the gas holdup, with the experimental results indicating an average gas holdup of 26.7% and 25.6% for $D=0.4$ and $D=1.0$ m, while the model yields an average gas holdup of 30.7% and 39.1%. A visual analysis also reveals that the performance of $D=1.0$ m is worse than the $D=0.4$ m gas holdup profile, as the latter captures the gas holdup gradient closer to the wall better, while $D=1.0$ fails to do so. Additionally, the RMSE values for $D=0.4$ and $D=1.0$ m were determined to be 0.048 and 0.141, respectively.

Turning to the liquid velocity profiles, it becomes evident that both velocity profiles are underestimating the central region. Specifically, the central velocity for $D=0.4$ m and $D=1.0$ m is 45.4% and 40.2% lower, compared to the experimental values. Analyzing the normalized velocity profile reveals it corresponds reasonably well with the experimental values. This is because the ratio between the central velocity and the measured velocity is constant for each radial coordinate, independent of its true value. Consequently, the normalized liquid velocity profile is not suitable for comparing the flow behavior observed across different scales.

4.2 Iterative scheme

Building upon the base case, the model is expanded by incorporating the NITA iterative scheme to improve computational efficiency. Considering that the same governing equations as well as momentum transfer models are applied, it is expected that the gas holdup and liquid velocity profiles will remain largely unaffected by implementing the NITA iterative scheme. The results from this comparison are presented in table 16 and figure 17.

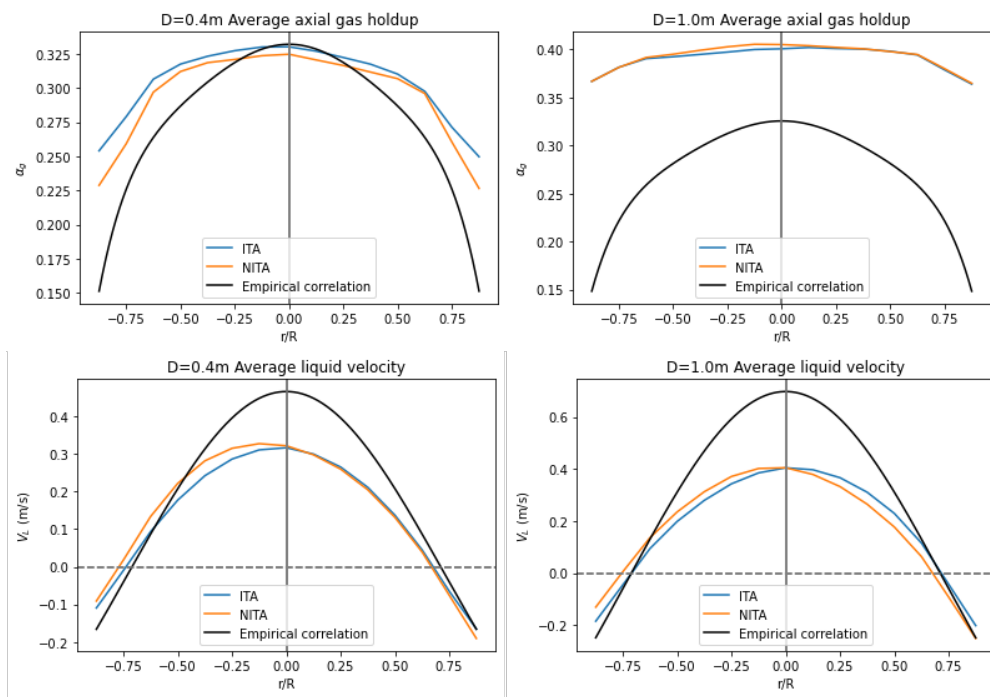


Figure 17: Graphs comparing the gas holdup (top) and liquid velocity (bottom) profiles of the Base model and the Base model with the NITA iterative scheme for $D=0.4$ m (left) and $D=1.0$ m (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 ITA	26.7%	30.7%	0.048	0.698	0.317	0.233
D0.4 NITA	26.7%	29.5%	0.037	0.698	0.322	0.227
D1.0 ITA	25.6%	39.1%	0.141	1.007	0.405	0.366
D1.0 NITA	25.6%	39.2%	0.143	1.007	0.405	0.370

Table 16: Average gas holdup, liquid velocities and RMSE values obtained from the base model and the base model with NITA iterative scheme for $D=0.4m$ and $D=1.0m$.

When examining the graphs, it becomes clear that both iterative solvers give nearly identical results, which is in line with the expectations. In the case of $D=0.4m$, the simulated gas holdup shows a marginal increase of only 1.2%, while the central velocity is only 0.005 m/s higher. Similarly, when considering $D=1.0m$, minimal change is observed in central velocity, while the gas holdup increased by only 0.1%.

The goal of implementing the NITA solver was not to achieve a more accurate solution but rather to reduce the computational time required for simulations. Comparing the simulation times of the NITA solver and the ITA solver (figure 18), shows that the NITA solver achieved simulation speeds that were approximately 3.7 and 5.3 faster, compared to the ITA solver.

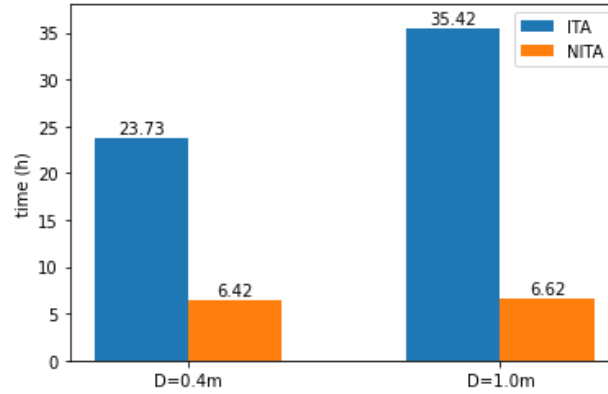


Figure 18: Runtime comparison between the ITA and NITA iterative solver schemes for $D=0.4m$ and $D=1.0m$.

This decrease in computational time can be attributed to the reduction in the number of calculations that are required to be solved, as explained in 2.4.9. This is achieved by first converging the momentum equations before advancing to the next set of equations. This approach however increasing the splitting error, did not have a notable impact on the solution. This suggests that the splitting error remained smaller than the truncation error[37], which represents the difference between the exact and discrete solution.

Due to its reduction in simulation time while maintaining an accurate solution, the NITA solver was selected for all subsequent simulations.

4.3 Swarm modification

Until now, we have assumed a constant drag modification to account for the presence of bubble swarms. In this step we will evaluate the impact of a swarm modification that considers the local gas holdup by implementing two models. One model, referred to as the modified Simonnet model (equation: 2.29), was used in the work of Ertekin during the validation of the experiments conducted by Raimundo. The second model utilizes the Grace drag formulation (equation: 2.19), where the parameter 'n' was set to 4. It is

important to note that the latter swarm modification model, based on Grace drag, can only be activated with the Grace drag model. Hence, to ensure a fair comparison, the Grace drag formulation was chosen for both simulations. In chapter 4.6, a comparison will be made between the Tomiyama and Grace drag formulations. The results of this comparison are presented below.

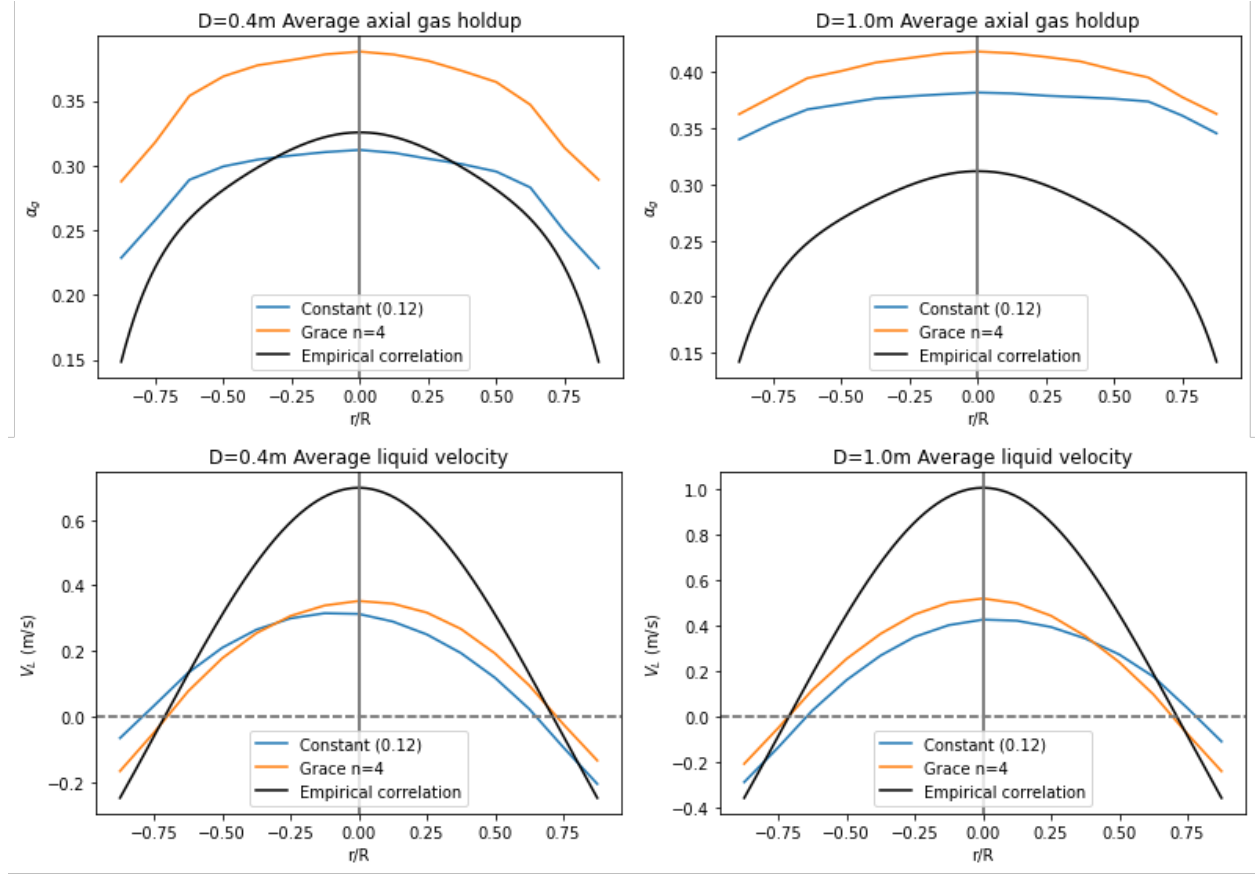


Figure 19: Graphs comparing the gas holdup (top) and liquid velocity (bottom) profiles of the Base model with NITA and a constant drag modification, and the Base model with NITA and the Grace drag modification for $D=0.4m$ (left) and $D=1.0m$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 constant	26.7%	28.5%	0.033	0.698	0.313	0.237
D0.4 grace n=4	26.7%	35.5%	0.090	0.698	0.353	0.206
D1.0 constant	25.6%	37.0%	0.121	1.007	0.426	0.358
D1.0 grace n=4	25.6%	39.8 %	0.146	1.007	0.519	0.300

Table 17: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model with NITA and a constant drag modification, as well as the Base model with NITA and the Grace drag modification for $D=0.4m$ and $D=1.0m$.

As is apparent from the graphs, the modified Simonnet model was not included in the results due to its instability when running in Fluent. Chapter 6, provides detailed explanation on what measures can be taken to increase the stability of the model. Due to these instabilities, it was only possible to compare the constant drag modification with the one provided in Fluent.

The analysis of liquid velocity reveals a slight increase in the central liquid velocity. Specifically, for $D=0.4\text{m}$ and $D=1.0\text{m}$, the observed increments are 0.04 m/s and 0.093 m/s , respectively. However, these differences are relatively small.

Upon analyzing the gas holdup profiles, it is evident that the implementation of the Grace drag formulation leads to an overall increase in gas holdup. For $D=0.4\text{m}$, a increase of 7.0% is observed, while for $D=1.0\text{m}$, the increase is 2.8% . This difference in the impact of the Grace formulation can be attributed to the difference between the constant drag modification and the Grace drag modification being greater for $D=0.4\text{m}$ compared to $D=1.0\text{m}$. The higher gas holdup values are a result of the enhanced drag force acting on the bubbles, as drag force is less reduced until a gas holdup of 41.1% (Figure 20) is achieved, in comparison to the constant drag modification. Consequently, the bubbles experience reduced rise velocities, leading to longer residence times and consequently higher gas holdup values. Furthermore, the gas holdup profile is less flat, suggesting that bubbles near the walls tend to migrate towards the central region. This behavior can be attributed to the greater drag force experienced by the bubbles near the wall, as the drag reduced less in that region. It is expected that the phenomenon of bubble accumulation in regions with higher gas holdup will be more pronounced when employing the Simonnet model, where there is a greater difference in the drag modification between the central and near-wall regions.

Unfortunately, it was determined that the Grace drag force modification could not run in a stable manner when implementing the BIT. Therefore, the decision was made to proceed with the constant drag modification instead.

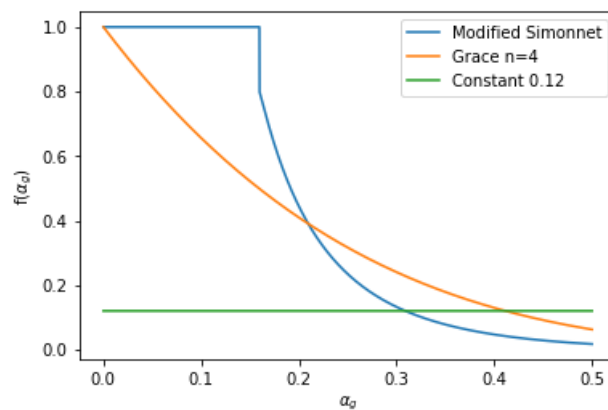


Figure 20: Modeled drag modification as a function of gas holdup, compared to the constant drag modification of 0.12

4.4 BIT

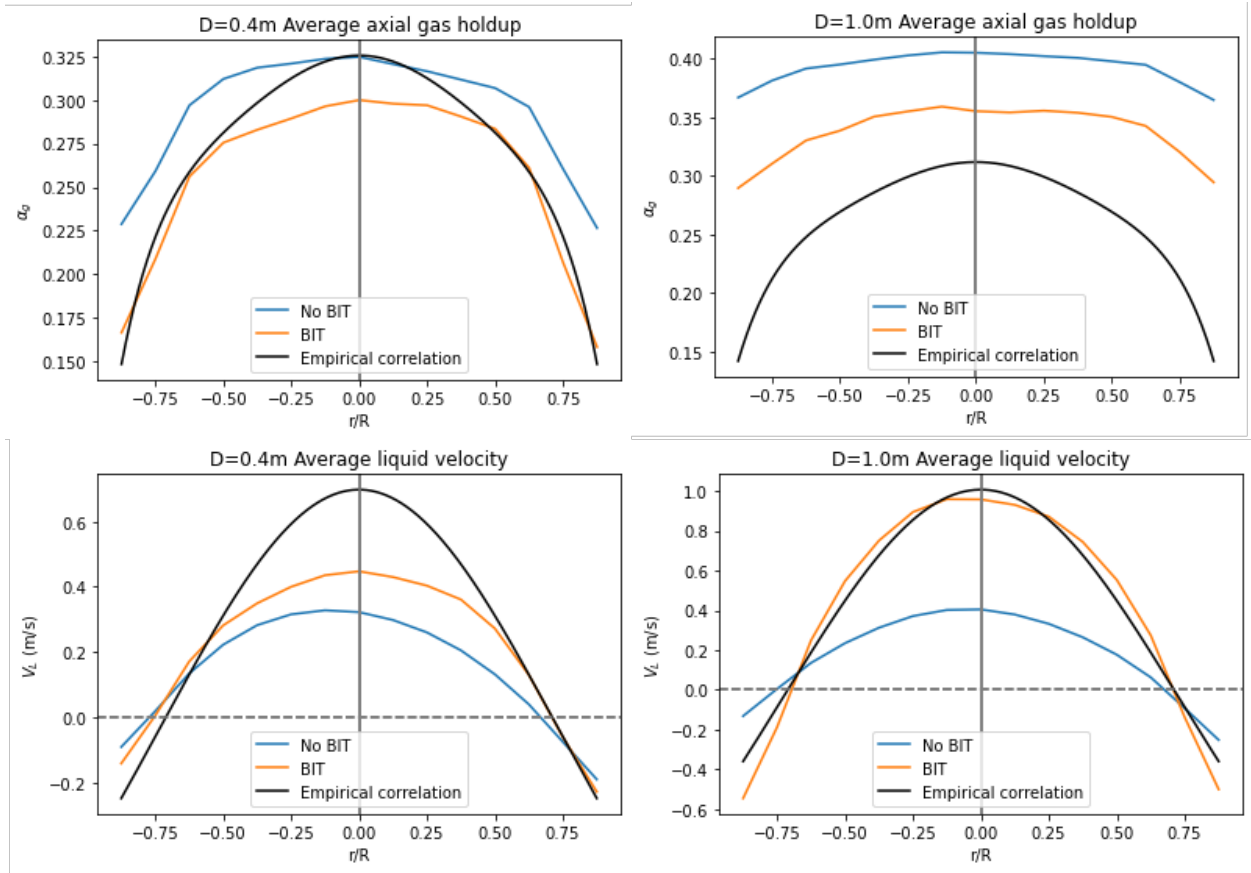


Figure 21: Comparison between the gas holdup (top) and liquid velocity (bottom) profiles of the Base model (including NITA) with and without BIT for $D=0.4\text{m}$ (left) and $D=1.0\text{m}$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 no BIT	26.7%	29.5%	0.037	0.698	0.322	0.227
D0.4 BIT	26.7%	25.8%	0.016	0.698	0.447	0.140
D1.0 no BIT	25.6%	39.2%	0.143	1.007	0.405	0.370
D1.0 BIT	25.6%	33.7%	0.088	1.007	0.958	0.087

Table 18: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model (including NITA) with and without BIT for $D=0.4\text{m}$ and $D=1.0\text{m}$.

The graphs above show the base model with the NITA iterative scheme with BIT enabled. When comparing this to the model without BIT, we visually see that BIT produced a notable improvement in both liquid velocity and gas holdup for both scales. When looking in greater detail at the gas holdup, it is clear that, the RMSE almost halved in both cases, decreasing from 0.037 to 0.016 and from 0.143 to 0.088, for $D=0.4$ and $D=1.0$, confirming the visual observations. Especially for $D=0.4\text{m}$, a better agreement with the experimental correlation can be observed. However, the central gas holdup is still underestimated. The same accounts for the model with $D=1.0\text{m}$, in which a reduction of the average gas holdup of 4.8% had been achieved, but the gas holdup is still underestimated, particularly near the walls. Regarding the liquid velocity profile, similar improvements can be identified. For the model with $D=0.4\text{m}$, we see an increase of the central velocity and

a decrease in the velocity closer to the wall, converging more towards the experimental value. As evident from the decrease in the RMSE values from 0.227 to 0.140 for $D=0.4\text{m}$ and from 0.370 to 0.087 for $D=1.0\text{m}$, it is clear that Notable improvements have been achieved.

Previous studies conducted by Fletcher and McClure [18][54] have also demonstrated the improved agreement with experimental data resulting from the inclusion of BIT. In line with Fletcher’s findings, the present study demonstrated a notable increase in the energy dissipation rate. (Figure 22)

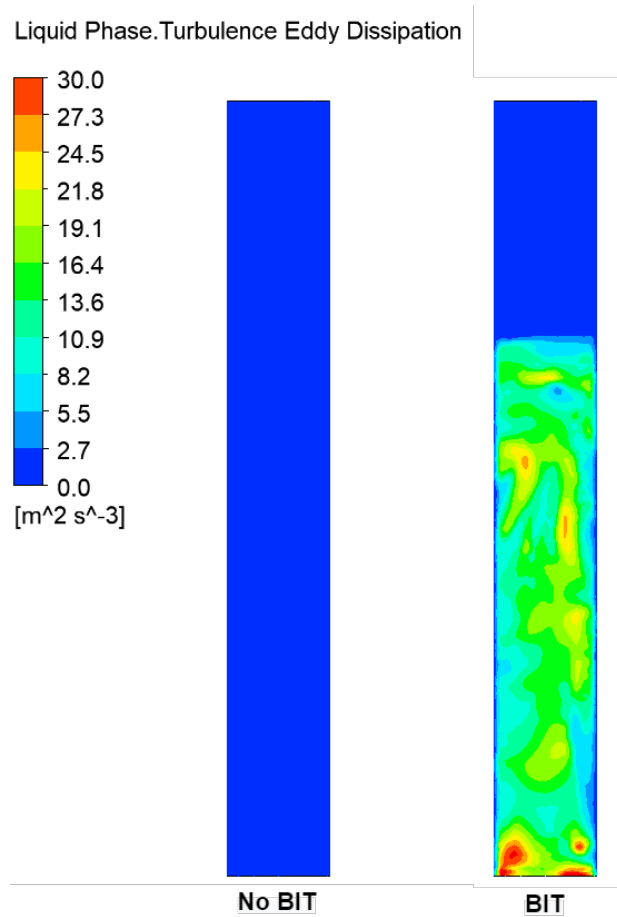


Figure 22: Comparison of turbulent dissipation rates between base model (including NITA) without BIT (left) and with BIT (right) for $D=0.4\text{m}$.

4.5 Turbulent dispersion

After incorporating the BIT into the base model, which already included the NITA solver, the influence of the turbulent dispersion force was examined. This force was modeled by activating the Burns turbulent dispersion model in Fluent.

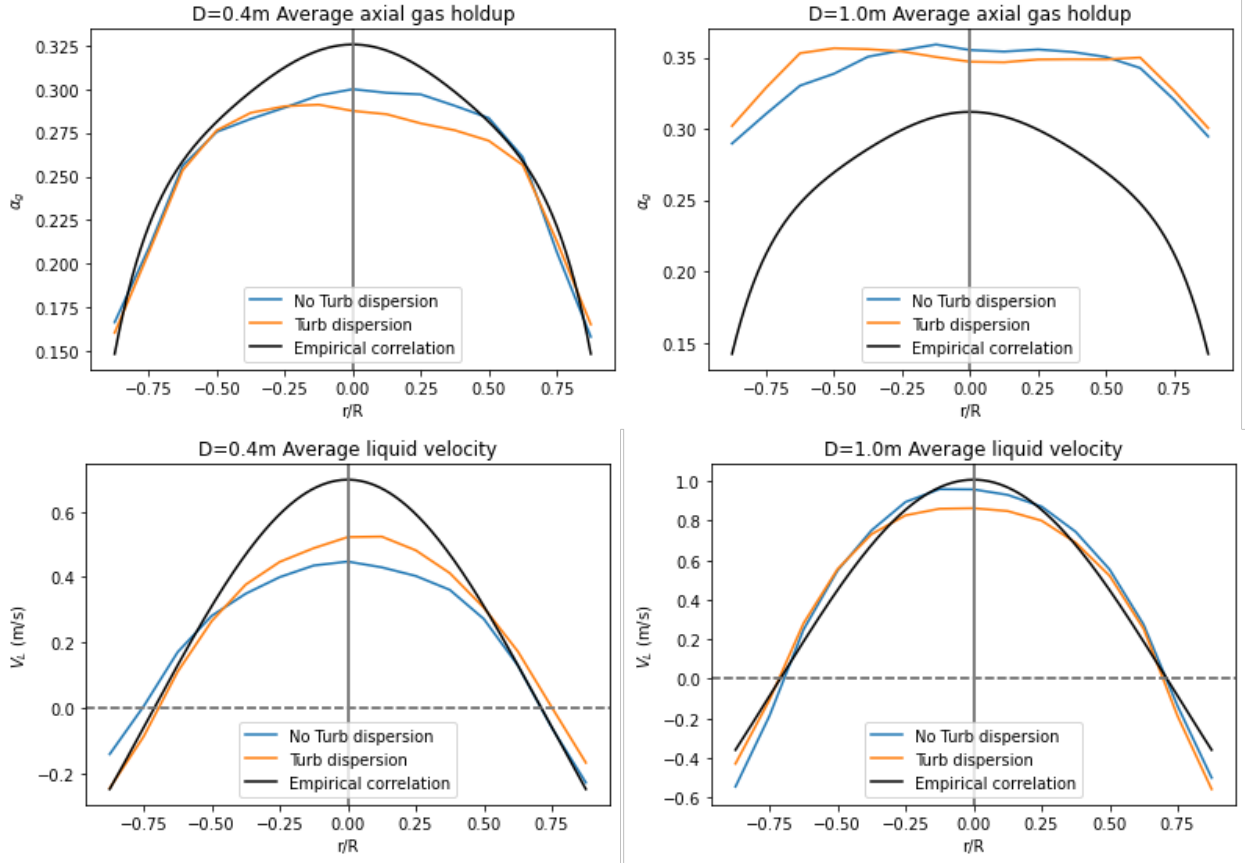


Figure 23: Comparison between the gas holdup (top) and liquid velocity (bottom) profiles of the Base model (including NITA and BIT) with and without turbulent dispersion for $D=0.4\text{m}$ (left) and $D=1.0\text{m}$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 no TD	26.7%	25.8%	0.016	0.698	0.447	0.140
D0.4 TD	26.7%	25.3%	0.021	0.698	0.522	0.099
D1.0 no TD	25.6%	33.7%	0.088	1.007	0.958	0.087
D1.0 TD	25.6%	34.1%	0.094	1.007	0.863	0.096

Table 19: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model (including NITA and BIT) with and without turbulent dispersion for $D=0.4\text{m}$ and $D=1.0\text{m}$.

Visually an improvement in the velocity profile for $D=0.4\text{m}$ can be seen, which is supported by the increase in central velocity to 0.522 m/s , reducing the difference between the model and the correlation by 0.075 m/s . This reduction in difference is also reflected in the decrease of RMSE by 0.041 . However, for $D=1.0\text{m}$, the inclusion of turbulent dispersion led to a slight deterioration, resulting in an increase in the difference between the correlation and the model. Consequently, there was an increase in the velocity RMSE by 0.009 . Examining the gas holdup profile, both $D=0.4\text{m}$ and $D=1.0\text{m}$ show an increase in RMSE of 0.005 and 0.006 .

Previous research had shown that the inclusion of the turbulent dispersion force in bubble column simulations improves agreement with experimental results [18][60]. It has also been observed that turbulent dispersion contributes to a more flat gas holdup profile [60]. However, in this study, no notable improvements were observed. Only a slight improvement was noticed in the gas holdup for $D=0.4\text{m}$. Upon closer examination of

the gas holdup profile, a slightly flatter profile for $D=1.0\text{m}$ was observed, which aligns with previous research findings [60].

Despite these ambiguous results, showing an improvement in $D=0.4\text{m}$ but not in $D=1.0\text{m}$, it was decided to include the turbulent dispersion force into our model. The rationale behind this decision lies in the fact that the existing literature on previous experiments clearly states that it improved their models[18][60][16].

4.6 Drag model

The base model has now been expanded to include the NITA iterative scheme, as well as BIT, a constant swarm modification and turbulent dispersion. The next step involves comparing different drag models. For this the Grace, and Tomiyama drag formulations were included. Results are shown in figure 24 and table 20

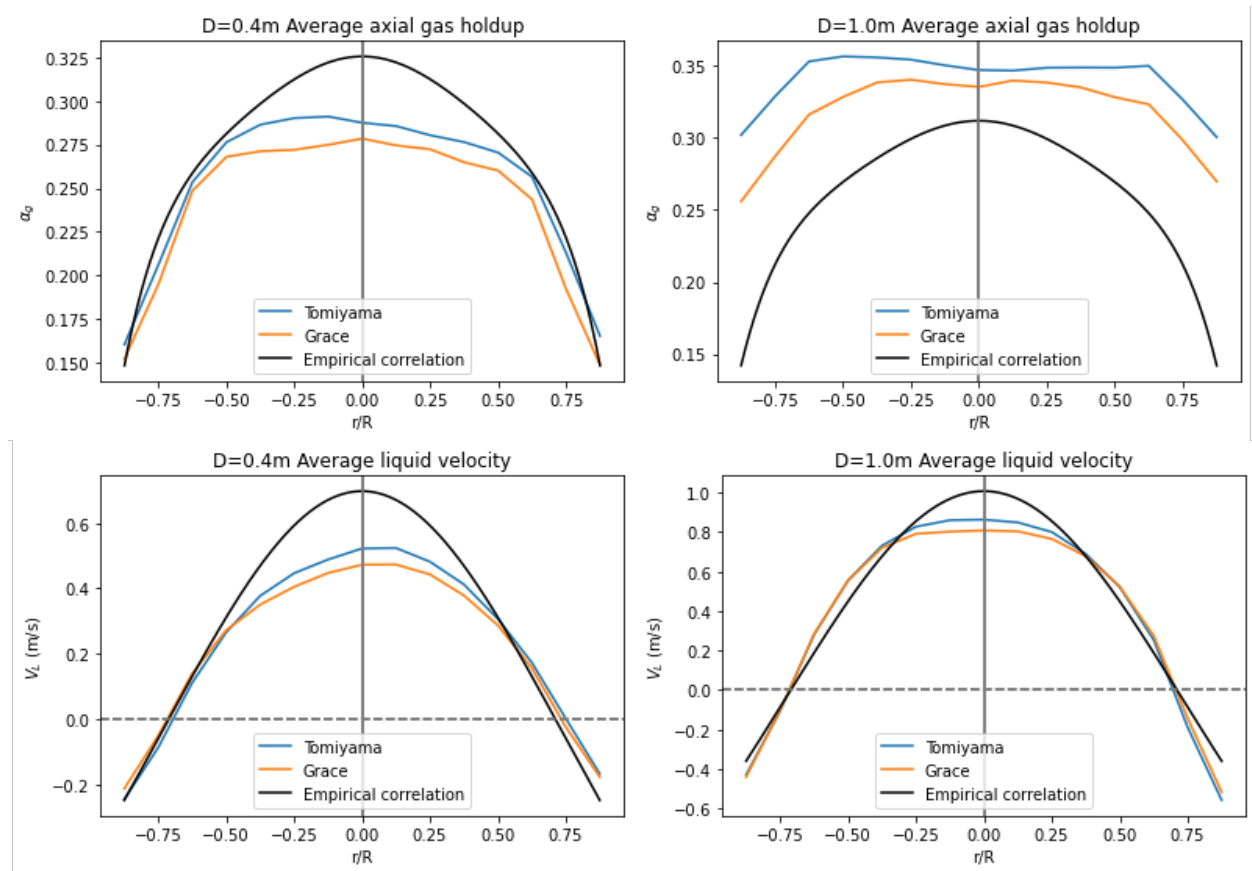


Figure 24: Comparison between the gas holdup (top) and liquid velocity (bottom) profiles of the Base model (including NITA, BIT, turbulent dispersion) with the Tomiyama drag formulation, and the Grace drag formulation for $D=0.4\text{m}$ (left) and $D=1.0\text{m}$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 Tomiyama	26.7%	25.3%	0.021	0.698	0.522	0.099
D0.4 Grace	26.7%	24.1%	0.031	0.698	0.473	0.124
D1.0 Tomiyama	25.6%	34.1%	0.094	1.007	0.863	0.096
D1.0 Grace	25.6%	31.8%	0.068	1.007	0.808	0.109

Table 20: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model (including NITA, BIT, turbulent dispersion) with the Tomiyama drag formulation, and the Grace drag formulation for $D=0.4m$ and $D=1.0m$.

Figure 24 shows a decrease in gas holdup and liquid velocity, for the grace drag formulation compared to the Tomiyama drag force. The decrease in liquid central velocity was 0.049 m/s and 0.055 m/s for $D=0.4m$ and $D=1.0m$. The gas holdup decreased by 1.2% respectively 2.3% for $D=0.4m$, and respectively $D=1.0m$. The implementation of the Grace drag formulation resulted to an increase in the liquid velocity RMSE for $D=0.4m$ and $D=1.0m$, as well as an increase in the gas holdup RMSE of $D=0.4m$. However, it is worth noting that the gas holdup RMSE for $D=1.0m$ showed a decrease, due to it's initial overestimation. The reduction in gas holdup, consequently resulted in an improved match with the experimental data, where the overestimation became smaller and thus decreasing the RMSE.

The observed reduction in both gas holdup and liquid velocity can be attributed to a consistent difference in the estimation of drag coefficients between the Tomiyama and Grace model. The Grace model predicts a lower drag coefficient compared to the Tomiyama model, resulting in a reduced bubble drag force and, consequently, a decrease in the liquid velocity, since the liquid doesn't get dragged along as much by the bubbles. Both drag coefficient models are a function of bubble diameter and slip velocity (equations 2.26 and 2.19). Even though both equations are a function of slip velocity and bubble diameter, assuming a fixed bubble diameter of 5.1mm, resulted in a constant difference of 0.14 between both models predicted drag coefficients (figure 25). Although this difference may vary at very small slip velocities (figure 25), it can be assumed that during the simulation, the difference between the drag coefficients remained unchanged. This assumption is justified by the fact that the bubble rise velocity falls within the range of 0.25 m/s for bubble diameters between 1 mm and 10 mm [10], very low slip velocity's are therefore not expected.

Based on the previously conducted validation studies by Fletcher and Ertekin[17][16], which demonstrated successes with the Grace model, we decided to implement this model in the following simulations. Although visually, it might seem like the Tomiyama model provides better results for the liquid and gas holdup profiles in this particular case, no notable differences between both models could be observed. Since there wasn't one model outperforming the other in our simulations, it was the rational decision to work in analogy with prior studies.

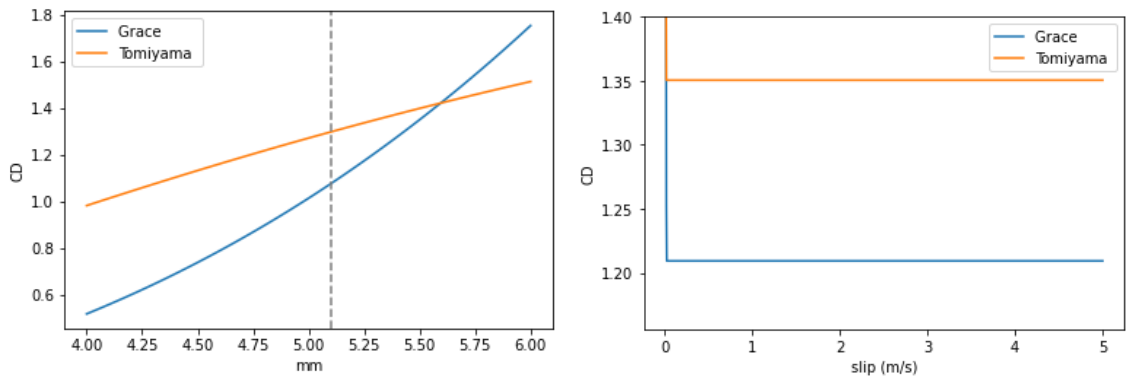


Figure 25: Modeled drag coefficient with a slip velocity of 2 m/s as a function of bubble diameter (left) and modeled drag coefficient with a constant bubble diameter of 5.1mm as a function of slip velocity (right).

4.7 Turbulence modeling

A comparison has been made to assess the impact of different turbulence models on the simulation results. For these simulations, the base model (table 15) was used, which now included the NITA iterative schema, constant drag modification, Yao and Morel BIT, Burns turbulent dispersion and Grace drag coefficient. Previous simulations had used the standard $k - \epsilon$ turbulence model. The comparison between different turbulence models involved evaluating both the standard the standard $k - \epsilon$ and the RNG $k - \epsilon$ turbulence models.

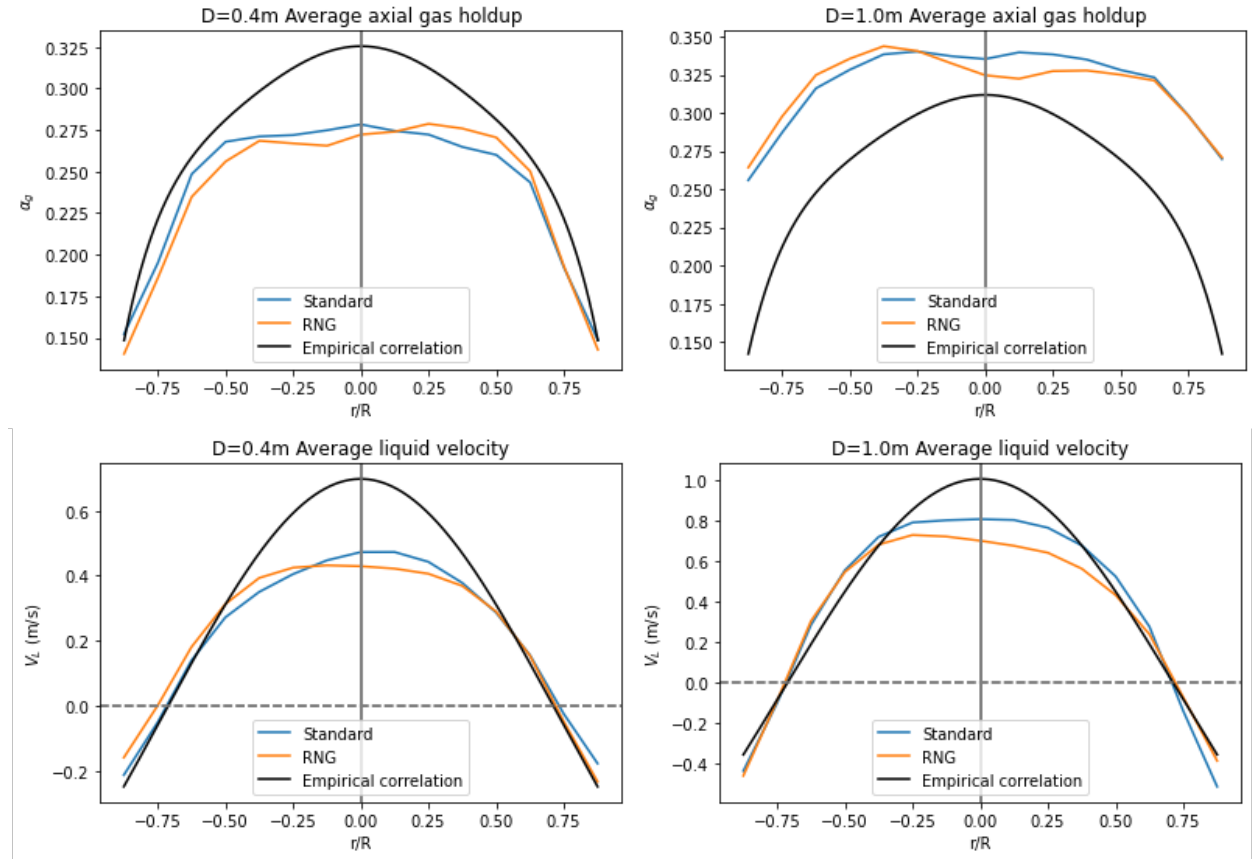


Figure 26: Comparison between the gas holdup (top) and liquid velocity (bottom) profiles of the Base model (including NITA, BIT, turbulent dispersion, Grace drag formulation) with the standard $k - \epsilon$ turbulence model, and the RNG $k - \epsilon$ turbulence model for $D=0.4m$ (left) and $D=1.0m$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 standard	26.7%	24.1%	0.031	0.698	0.473	0.124
D0.4 RNG	26.7%	23.8%	0.033	0.698	0.429	0.138
D1.0 Standard	25.6%	31.8%	0.068	1.007	0.808	0.109
D1.0 RNG	25.6%	31.7%	0.070	1.007	0.701	0.153

Table 21: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model (including NITA, BIT, turbulent dispersion, Grace drag formulation) with the standard $k - \epsilon$ turbulence model, and the RNG $k - \epsilon$ turbulence model for $D=0.4m$ and $D=1.0m$.

The inclusion of the RNG $k - \epsilon$ model, did not result in a notable difference in the model's ability to predict

the liquid velocity and gas holdup. However, it did predicted a lower average gas holdup for both cases, increasing the difference between the model and correlation from 2.6% to 2.9% for D=0.4m. The decrease for D=1.0m was only 0.1%. These reductions in gas holdup resulted in a minimal increase of both their RMSE of 0.002. Similarly, for the liquid velocity, an increase in the RMSE of 0.014 and 0.044 was observed for D=0.4 and D=1.0m. This increase was accompanied by a rise in difference between the model's central velocity and the correlation, with a difference of 0.044 m/s and 0.107 m/s for D=0.4 and D=1.0m.

The RNG $k - \epsilon$ was designed to include the turbulence generated by swirly flows, thereby improving accuracy in such flows[37]. However, the flow in a bubble column does not exhibit swirling flows within its domain. The inclusion of the RNG $k - \epsilon$ did therefore not lead to a better agreement with the correlation. As a result, the decision was made to exclude the RNG $k - \epsilon$ model from the following simulations and continue using the standard $k - \epsilon$ model. This choice aligns with the approach followed by Fletcher and Ertekin in their validation studies.

4.8 Inclusion of the ideal gas law

For all previous simulations, a constant gas density had been assumed instead of using the ideal gas law. This choice was motivated by stability concerns, as the inclusion of the ideal gas law was found to destabilize the Yao and Morel BIT model. To address this, a two-step simulation approach was adopted. Initially, the simulation was run for 20 seconds without incorporating the ideal gas law, ensuring stability. After which the ideal gas law was included.

Assuming a constant gas density is a great simplification, considering the hydrostatic pressure at the bottom of D=1.0m gives a density of 1.642 kg/m^3 , compared to the constant density assumption of 1.204 kg/m^3 , resulting in an increase of 36.4%. Instead of employing a constant value for all reactors, the selection should have been predicated on the average pressure, thereby reducing the density difference between both approaches. This is discussed in further detail in Chapter 6. For D=0.4, the increase in density was 13.5% due to the lower initial liquid height (1.6m instead of 4m in D=1.0m), resulting in a lower hydrostatic pressure.

In the following simulations, a comparison has been made between the base model (table 15) including; the NITA iterative schema, constant drag modification, Yao and Morel BIT, Burns turbulent dispersion and Grace drag coefficient, and the same model with the gas density modeled according to the ideal gas law. The simulation results are presented in Figure 27 and table 22.

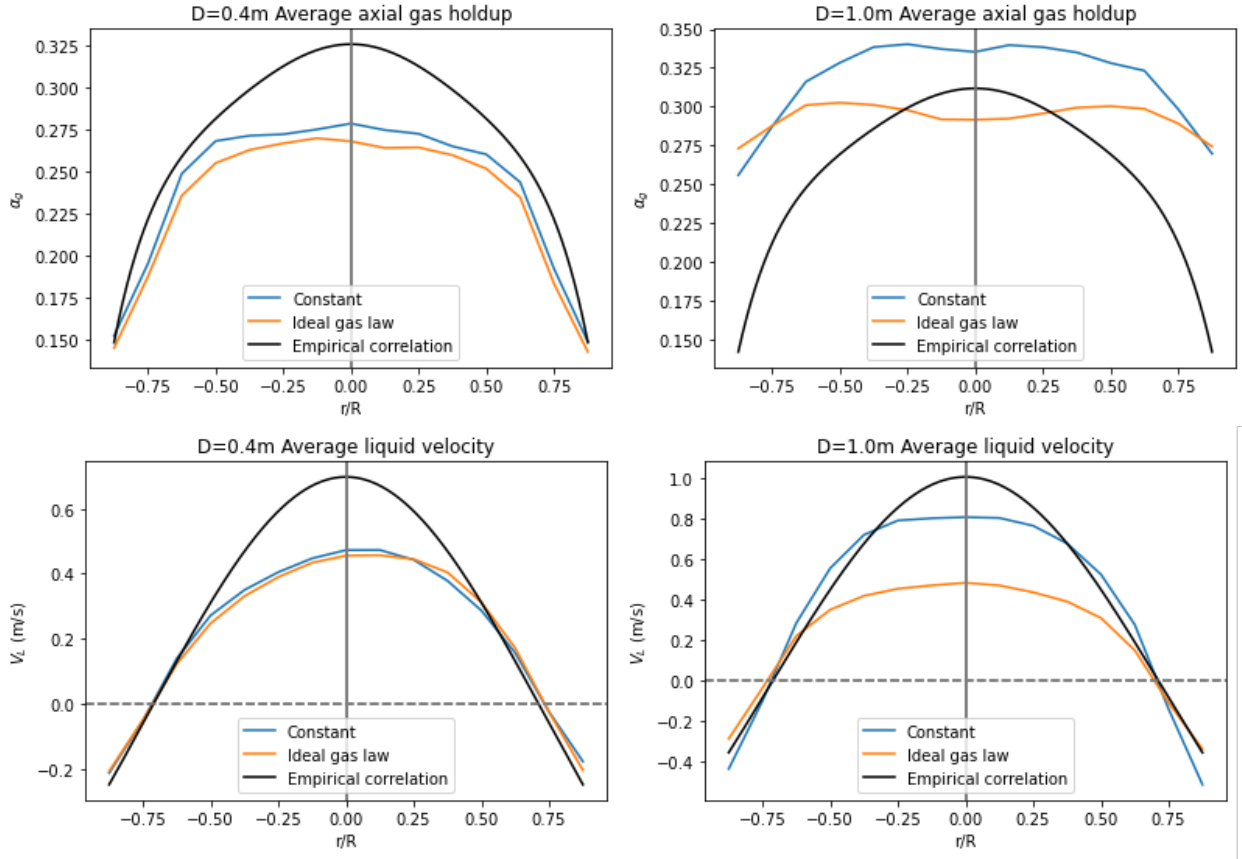


Figure 27: Comparison between the gas holdup (top) and liquid velocity (bottom) profiles of the Base model (including NITA, BIT, turbulent dispersion, Grace drag formulation) with and without the ideal gas law for $D=0.4\text{m}$ (left) and $D=1.0\text{m}$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 Constant	26.7%	24.1%	0.031	0.698	0.473	0.124
D0.4 Ideal gas	26.7%	23.3%	0.038	0.698	0.456	0.131
D1.0 Constant	25.6%	31.8%	0.038	1.007	0.808	0.170
D1.0 Ideal gas	25.6%	29.3%	0.061	1.007	0.482	0.290

Table 22: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model (including NITA, BIT, turbulent dispersion, Grace drag formulation) with and without the ideal gas law for $D=0.4\text{m}$ and $D=1.0\text{m}$.

If we analyse the results of both reactors, it becomes clear that the effect on $D=0.4\text{m}$ is relatively minor compared to that of $D=1.0\text{m}$. In $D=0.4$, the liquid velocity profile remained largely similar, with only a slight decrease in the central velocity of 0.017 m/s compared to the model with constant density. The gas holdup profile showed a decrease of 0.8% in the average gas holdup.

In contrast, the ideal gas law had a more bigger effect on $D=1.0\text{m}$, as expected, due to the 36.4% density increase compared to the 13.5% for $D=0.4\text{m}$. For $D=1.0\text{m}$, the gas holdup experienced a reduction of 2.5% and the RMSE increased by 0.023 . Additionally, the central velocity decreased by 0.326 m/s , accompanied by an increase in the RMSE by 0.120 .

The decrease in gas holdup can be attributed to the increased gas density resulting from the activation of

the ideal gas law. The gas holdup is calculated by dividing the gas volume by the total gas-liquid volume (equation 2.4). When the ideal gas law is applied and the gas density increases, the gas volume decreases, leading to a reduction in the gas holdup. This decrease in gas volume, also decreases the number of bubbles present, resulting in less bubbles exerting an upward force on the liquid. Consequently, the liquid velocity decreases as a result of this reduced upward force.

Despite the decrease in the model's ability to predict the gas holdup and liquid velocity caused by the activation of the ideal gas law, it should still be included in the simulation. The ideal gas law is a fundamental equation that relates pressure and volume. Its inclusion is therefore crucial for achieving more realistic simulations. Without incorporating the ideal gas law, the model would neglect this correlation between pressure and volume.

4.9 Improved model

Past efforts have resulted in the refinement of the base model, to an improved model. The specific settings for both models are summarized in table 23. In this chapter both models have been compared, in their ability to reproduce the experimental data. The comparison results, are displayed in figure 28 and table 24.

	Base model	Improved model
Iterative solver	ITA	NITA
Drag model	Tomiyama	Grace
$k - \epsilon$ turbulence model	Standard	Standard
Dispersed phase turbulence	Dispersed	Dispersed
Swarm modification	0.12	0.12
BIT	-	Yao and Morel
Turbulent dispersion	-	Burns
Gas density	1.204 kg/m^3	ideal gas law

Table 23: Base and improved model setting following a step wise optimization method

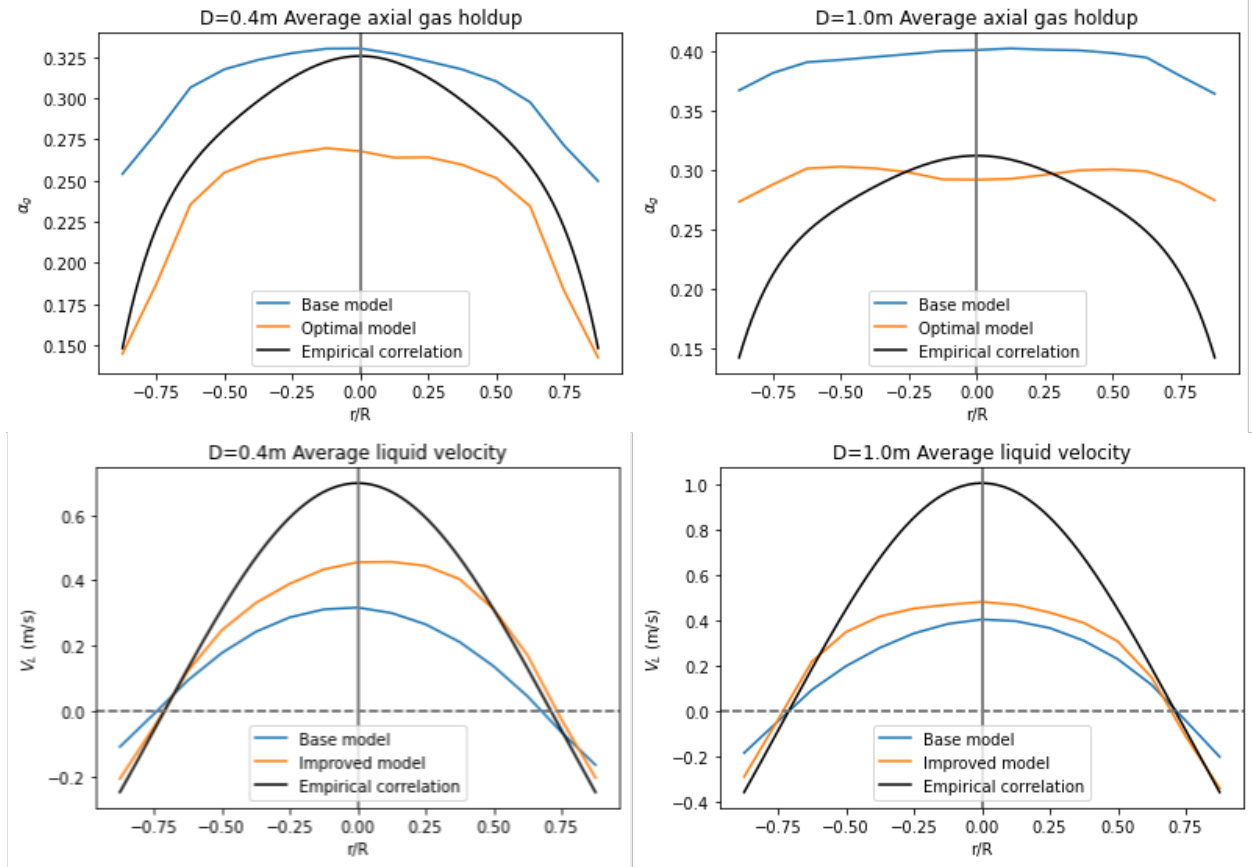


Figure 28: Comparison between the gas holdup (top) and liquid velocity (bottom) profiles of the Base model and the improved model for $D=0.4m$ (left) and $D=1.0m$ (right).

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
D0.4 Base model	26.7%	30.7%	0.048	0.698	0.317	0.233
D0.4 improved model	26.7%	23.3%	0.038	0.698	0.456	0.131
D1.0 Base model	25.6%	39.1%	0.141	1.007	0.405	0.366
D1.0 improved model	25.6%	29.3%	0.061	1.007	0.482	0.290

Table 24: Average gas holdup, liquid velocities, and RMSE values obtained from the Base model and the improved model for $D=0.4m$ and $D=1.0m$.

The optimized model has demonstrated to improve the ability to predict the gas holdup and liquid velocity for $D=0.4$ and $D=1.0m$. In the case of $D=0.4m$, the optimized model shows a better gas holdup gradient compared to the base model, which can be attributed to the inclusion of the BIT model, resulting in a RMSE reduction of 0.01. Similarly, for $D=1.0m$, the optimized model shows a improvement by reducing the average gas holdup by 9.8%, resulting in a 3.7% difference between the correlation and the simulation, which was originally 13.5%. Additionally, improvements in the liquid velocity profiles are observed, with the RMSE decreasing by 0.080 and 0.076 for $D=0.4$ and $D=1.0m$.

The improved model has demonstrated to be better able to capture the experimental gas holdup and velocity profiles, while maintaining computational stability for both $D=0.4m$ and $D=1.0m$.

4.10 Detailed model comparison

Previous model comparisons have primarily focused on scale-dependent profiles, neglecting the scale-independence and influence of measuring height. However, Raimundo did conduct an investigation into these aspects. His experimental findings demonstrated the scale-independence of the results, as shown in Figure 29 [15]. These findings align with Schweitzer’s research, which described that the gas holdup should follow equation 2.7, independent of its scale [34]. Raimundo’s work confirmed that the measuring height had a negligible effect on the gas holdup, which aligned with the observations made by Forret [26]. Therefore, the model was tested to validate its ability to reproduce both of these characteristics. This validation is crucial, as the model aims to serve as a valuable tool for assisting in the scale-up of bubble columns.

4.10.1 Scale independent gas holdup

To assess the improved model’s ability to predict the scale-independent characteristics of a bubble column, we compare the model results with the normalized gas holdup, equation: 2.7 and the experimental results from Raimundo [15].

In order to validate the model’s ability to capture the scale-up criteria, simulations were performed for all four reactor geometries using the improved model settings outlined in Table 23. The simulations for $D=0.15$ and $D=3.0$ were unfortunately not stable, and no meaningful results could be obtained from them. The simulation for $D=0.15$ m could only be run for a duration of 0.642 seconds, whereas $D=3.0$ m was able to run for the required 90 seconds. However, the results obtained from this simulation were considered invalid due to the presence of gas accumulation below the water surface at the measurement height (foaming). Such foaming regimes are uncommon in bubble columns [61] and were not observed by Raimundo, thus rendering the results unreliable. As a result, the scale-up criteria were tested based on the already known results obtained from $D=0.4$ m and $D=1.0$ m. Figure 29 shows the normalized gas holdup profiles for $D=0.4$ m and $D=1.0$ m as well as the empirical correlation.

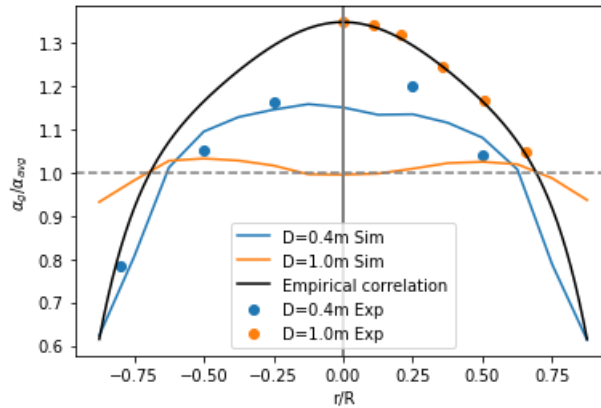


Figure 29: Comparison of non-dimensionalized gas holdup profiles of $D=0.4$ m, $D=1.0$ m, and the experimental correlation.

The analysis of the normalized gas holdup profile provided by Raimundo reveals that the outcomes obtained for $D=1.0$ comply to the scale independent correlation. For $D=0.4$ m, although the data appears to deviate from the normalized gas holdup, but as stated by Raimundo[15], this data follows the normalized gas holdup profile with an average error below 5%.

From figure 29, it is evident that current model is not able to accurately capture the scale-independence of the normalized gas holdup. This is evident from the different normalized gas holdup values obtained from

both reactors, implying that the current model is unsuitable for scaling up.

An important consideration should be taken into account regarding this comparison, the average gas holdup was only measured within the range of $r/R = -0.875$ to 0.875 . Consequently, the measured average gas holdup is higher compared to when the entire range is considered. Since a lower average gas holdup would yield higher normalized values, it would align the normalized gas holdup more closely with the normalized gas holdup correlation.

4.10.2 Measurement height

Raimundo's work noted that the measuring height had a negligible effect on the gas holdup profile. Therefore, the model was tested to validate its ability to reproduce this characteristic. For this comparison, only $D=0.4m$ was considered, as Raimundo had only provided this dataset. Visually analyzing the gas holdup profiles from the simulation, it becomes evident that the gas holdup is influenced by the measurement height, in contrast to the Raimundo data, as depicted in Figure 30. This observation implies that the simulation fails to capture the stable region where the gas holdup is independent of measuring height.

While Raimundo acknowledges the discrepancies in the middle of the column for the profiles measured at $H/D = 2.5$ and 3.75 , the difference between the simulated measurements at different heights demonstrate larger deviations compared to Raimundo's experimental observations. Hence, it is not possible to conclude that the model is able to capture a stable region.

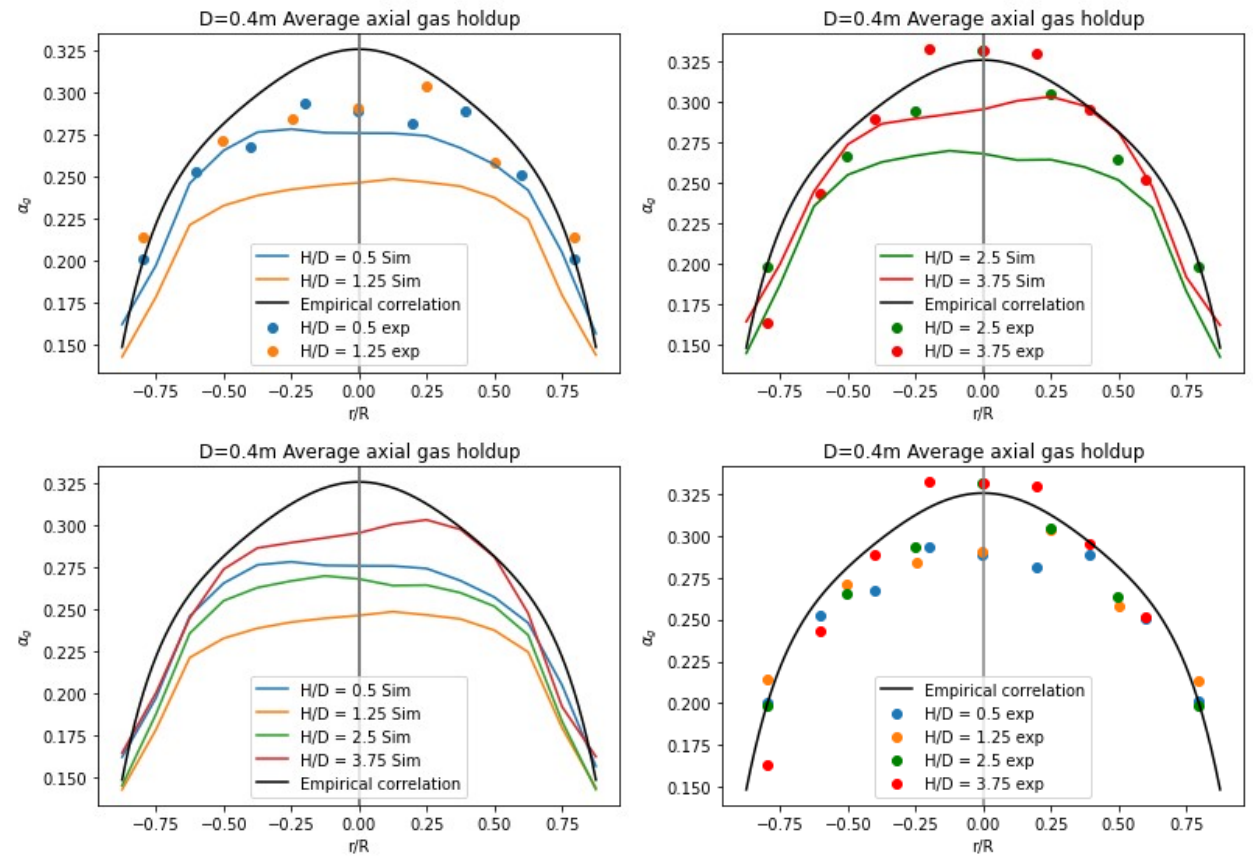


Figure 30: Comparison between the gas holdup profiles obtained at different measuring heights (top), the improved model gas holdup profiles (bottom left), and the experimental measured gas holdups (bottom right).

Looking at the RMSE value's from these simulations as well as the graphs in figure 30, it is evident that the gas holdup does not increase along the reactor height. The gas holdup measured at $H/D = 0.5$, doesn't follow the trend of increasing gas holdup along the reactor height, as is evident in the bottom left graph of in figure 30.

Upon analysis of the RMSE, and average gas holdup values derived from these simulations, as well as visual examination of the corresponding graphs, it is evident that the gas holdup does not exhibit a consistent increase along the vertical axis of the reactor. Notably, the gas holdup measured at $H/D = 0.5$ appears to be higher compared to the subsequent points, where a gradual increase can be observed.

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
$H/D = 0.5$	26.7%	27.1%	0.014	0.698	0.192	0.300
$H/D = 1.25$	26.7%	23.9%	0.036	0.698	0.358	0.194
$H/D = 2.5$	25.6%	24.1%	0.031	0.698	0.473	0.124
$H/D = 3.75$	25.6%	25.7%	0.017	0.698	0.558	0.067

Table 25: Average gas holdup and RMSE values obtained from the improved model at different heights in the reactor.

4.11 Mesh dependency

The mesh dependency comparison is a crucial step in CFD simulations, as it involves assessing the sensitivity of the solution to changes in mesh characteristics. In this study, the comparison focuses on a mesh containing 100,000 and 200,000 cells and the modeling of $D=1.0m$, as attempting to simulate other geometries with the finer mesh did not yield stable results.

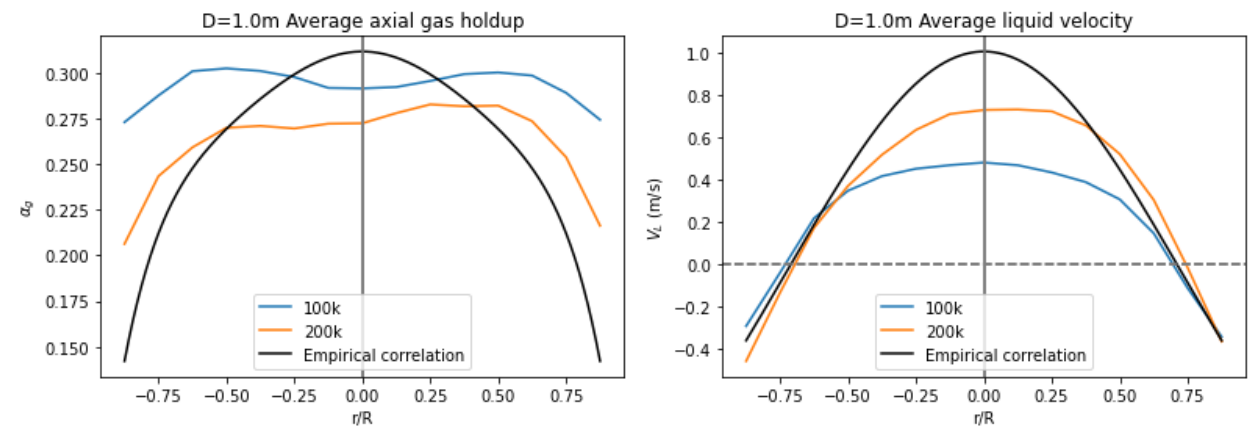


Figure 31: Comparison between the gas holdup (left) and liquid velocity (right) profiles of the improved model with 100k cells and with 200k cells for $D=1.0m$.

Figure 31, displays the results of the simulation, from which it is apparent that the mesh has an influence on the simulation results. This is not in agreement with mesh dependency studies performed by other researchers. The mesh used in the work of Ertekin used 80,100 cells, who used a finer mesh containing 211,960 cells had been used to perform a mesh independency study. Where it was concluded, that a coarser mesh did not influence the simulation results [16]. Other papers (not simulating Raimundo experiments), also suggested that a mesh containing 100,000 cells, was sufficient to not influence the results [18][54]. The main difference between these studies and this Study, is the cell types used. All above mentioned simulations

used hexahedral mesh. This mesh type has been simulated but was not able to run in a stable manner.

Figure 31 presents the simulation results, revealing a influence of the mesh on the simulation results. This observation is in contradicts previous mesh dependency studies conducted by other researchers. Ertekin also performed a mesh independence study. They increased the mesh from 80,100 cells, to 211,960 cells, after which they concluded that the results were not influenced by the coarse mesh. [16]. Similarly, other papers (not specifically simulating Raimundo experiments) suggested that a mesh containing 100,000 cells was sufficient to avoid influence from the mesh size on the results [18][54]. The key distinction between these studies and the one presented here lies in the cell types employed. All the other simulations employed a hexahedral mesh, which was also attempted in this study but could not be run in a stable manner.

4.12 Higher order discretization

Regarding the numerical discretization for bubble columns, it is recommended to incorporate higher order discretization schemes to mitigate discretization errors [62]. In the context of bubble column reactors, it is common to incorporate the Quick discretization scheme for the volume fraction and momentum transfer terms [16][17][62]. Building upon this established approach, the enhanced model has been extended to include the Quick discretization scheme for the volume fraction and momentum transfer terms.

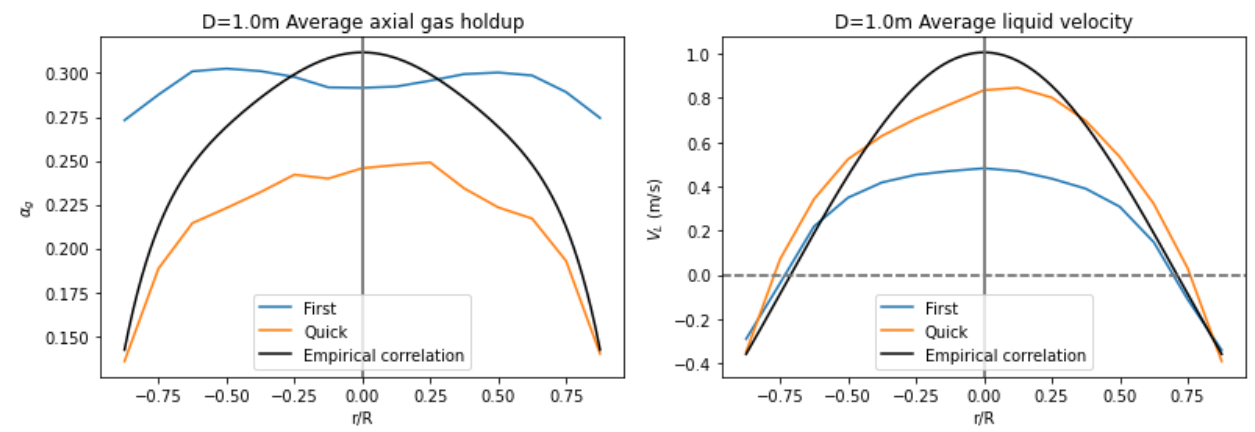


Figure 32: Comparison between the gas holdup (left) and liquid velocity (right) profiles of the improved model with and without Quick discretization for momentum transfer and volume fraction for $D=1.0m$.

Both improved models with diameters of 0.4m and 1.0m were expanded by including higher order discretization schemes. However, the model with a diameter of 0.4m could not be run in a stable manner. The simulation for $D=1.0m$ was successfully carried out. Nevertheless, the gas holdup graph presented in Figure 32 demonstrates a lack of consistency. This inconsistency arises due to the simulation failing to meet the convergence criteria of $5e-5$; instead, a residual of $1e-3$ was observed.

From Figure 32, it is evident that the inclusion of higher order discretization schemes leads to improved simulations. The higher-order schemes successfully captures the gas holdup gradients near the wall more accurately, resulting in a gas holdup profile that is less flat. This improvement is further supported by the reduced RMSE of the gas holdup, which decreased from 0.061 to 0.046.

In addition, the liquid velocity plot shows a more consistent trend, without any irregularities. Here, an improvement is observed as the central velocity increases from 0.482 m/s to 0.836 m/s, approaching the central velocity of the correlation, which was 1.007 m/s.

	Improved model	Improved model with Quick	Ertekin model
cell type	Poly-Hexcore	Poly-Hexcore	Hexahedral
Iterative solver	NITA	NITA	NITA
Drag model	Grace	Grace	Grace
$k - \epsilon$ turbulence model	Standard	Standard	Standard
Dispersed phase turbulence	Dispersed	Dispersed	Dispersed
Swarm modification	0.12	0.12	Simonnet
BIT	Yao and Morel	Yao and Morel	Yao and Morel
Turbulent disperion	Burns	Burns	Burns
Gas density	ideal gas law	ideal gas law	ideal gas law
Discretization			
Momentum	First order upwind	Quick	Quick
Volume fraction	First order upwind	Quick	Quick

Table 27: Overview of the model settings for the improved model, the improved model with Quick discretization for momentum and volume fraction, and Ertekins model settings.

These results align with the recommendation to incorporate higher-order discretization schemes for the momentum and volume fraction as an improvement in both graphs can be observed. However, it is crucial to consider that higher-order discretization schemes can potentially result in unstable simulations, as demonstrated in this particular case.

	Average gas holdup			Average Liquid velocity (m/s)		
	Exp	Sim	RMSE	Exp V_0	Sim V_0	RMSE
First	25.6%	29.3%	0.061	1.007	0.482	0.290
Quick	25.6%	21.5%	0.046	1.007	0.836	0.114

Table 26: Average gas holdup, liquid velocities, and RMSE values obtained from the improved model with and without Quick discretization for momentum transfer and volume fraction for $D=1.0m$.

4.13 Ertekin model

As the current study builds upon the model developed by Ertekin, a comparison between the simulation results of the two models has been conducted. For this comparison, $D=1.0$ was selected since Ertekin’s research did not include reactors smaller than this specific dimension [16]. Given that the $D=3.0m$ model in this work yielded unsatisfactory results, the decision was made to focus solely on $D=1.0m$ for the analysis. For the measurement of the liquid velocity profile, the location at $H/D = 3.75$ was chosen, as the results provided by Ertekin were limited to this height [16]. However, the gas holdup profile for this comparison was still obtained from $H/D = 2.5$, as those were available for this height.

For the comparison, three simulation results have been included: the improved model, the improved model with Quick scheme discretization (as discussed in Chapter 4.12), and the results obtained from Ertekin’s model, along with the correlation plot. The decision to incorporate the improved model with Quick discretization was based on its promising results. The model by Ertekin incorporates the Simonnet model, utilizes hexahedral cells instead of poly-hexcore cells, and employs the Quick discretization scheme. An overview of the settings for each of the models is presented in Table 27.

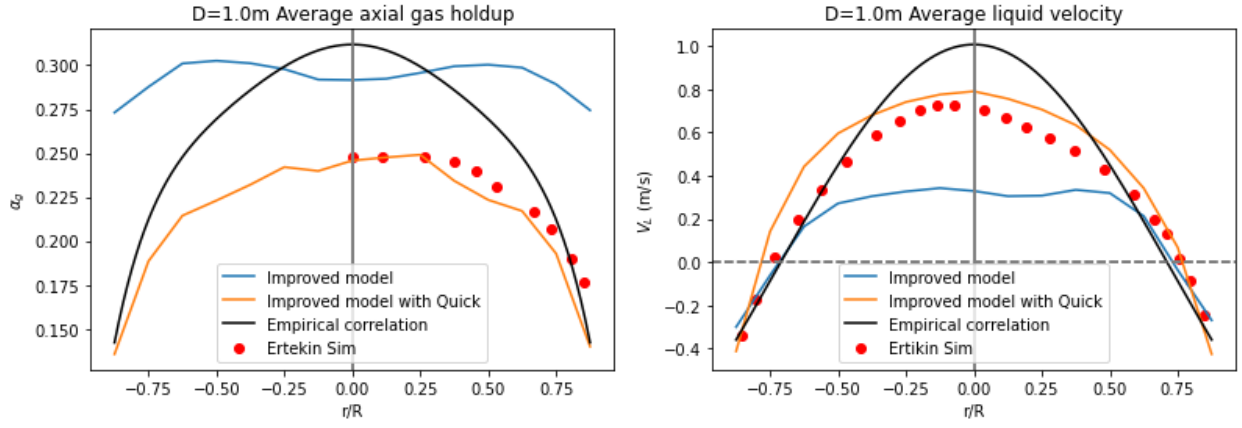


Figure 33: Comparison between the gas holdup profiles (left) and liquid velocity profiles (right) of the improved model with and without Quick discretization for momentum transfer and volume fraction, and Ertekin's model for $D=1.0m$. The liquid velocity is measured at $H/D = 3.5$, deviating from the standard measurement at 2.5.

Upon examining both graphs, it is evident that the improved model without the Quick scheme fails to comply with the simulation results presented by Ertekin. The gas holdup is overestimated, with a flattened profile, while the liquid velocity is underestimated, exhibiting a flat profile in the central region, as observed previously.

The results from the improved model with the higher order discretization show to be in good agreement with the results obtained by Ertekin. Where the only difference between both models has been drag modification, and cell type, for which Ertekin used the simonnet model and hexahedral cells and the improved model used a constant value of 0.12 as drag modification and Poly-Hexcore cells. From this we can conclude that the use of simonnet nor hexahedral cells have a influence on the results for $D=1.0$.

Ertiken claims that her simulation is capable of reproducing the experimental data [16]. However, when utilizing Raimundo's experimental correlation [15] instead of the experimental results used by Ertiken [16], it becomes evident that the simulation underestimates the gas holdup and liquid velocity, failing to replicate the correlation and thus concluding that the simulation is capable of reproducing the experimental setup. Ertiken's decision to directly use Raimundo's results, rather than incorporating the experimental correlation to validate her model, likely contributed to these differing observations, as further discussed in 6. It should be noted that a different measuring height (H/D of 3.5 instead of 2.5) was used to reach the conclusion that the liquid velocity reproduces the experimental results [16].

5 Conclusion

The aim of this project was to establish a single CFD model capable of accurately simulating gas holdup and liquid velocity as well as scale independent phenomena. This involved the establishment of a CFD model replicating experimental observations in bubble columns with diameters ranging from 0.15m to 3m.

Initially, a base model was established as the foundation for model evaluations. The base model exhibited a flat gas holdup profile across two different reactor diameters, $D=0.4\text{m}$ and $D=1.0\text{m}$. However, it was observed that the average gas holdup for $D=0.4\text{m}$ was overestimated by 4.0%, while for $D=1.0\text{m}$, the overestimation reached 13.5%. Furthermore, an analysis of the central liquid velocity revealed an underestimation of 0.381 m/s and 0.602 m/s for $D=0.4\text{m}$ and $D=1.0\text{m}$, respectively.

Subsequently, the base model was extended through the incorporation of diverse turbulence and drag formulations, higher order discretization methods, and an alternative iterative scheme. These were implemented in a step wise manner aimed to explore the effects of these modifications on the model's performance. The outcomes obtained from these enhancements are summarized below.

- The incorporation of the NITA iterative solver showed minimal influence on the resultant outcomes. However, it accelerated the run time by a factor of 3.7 and 5.3 for $D=0.4\text{m}$ and $D=1.0\text{m}$, respectively.
- The inclusion of the Grace drag modification with $n=4$ demonstrated an improvement in the simulation bringing it closer to the correlation value, and yielding a less flat profile. The stability of this model was however compromised upon the inclusion of the BIT model, and grace drag modification was therefore not included in further models.
- The incorporation of the Yao and Morel BIT model yielded a substantial improvement in the simulation results. As it reduced the RMSE value for the gas holdup and for the liquid velocity. These outcomes align with the previous validation efforts conducted by Ertekin and Fletcher, providing further substantiation for the efficacy of the Yao and Morel BIT model [16][17].
- The inclusion of turbulent dispersion did not improve the simulation results. Previous studies have shown that turbulent dispersion can improve the accuracy of simulations [18][60]. Nevertheless, due to its proven success in other simulations, turbulent dispersion was included in the base model.
- The Grace drag model exhibited a marginal influence on the simulation results compared to the Tomiyama drag formulation. With Grace the liquid velocity decreased for both reactor geometries, due to the lower drag coefficient estimation of the Grace model. The Grace drag model led to an overall reduction in the gas holdup, improving alignment with the experimental correlation for $D=1.0\text{m}$, while slightly compromising the agreement for $D=0.4\text{m}$.
- The RNG $k - \epsilon$ model has been compared to the standard $k - \epsilon$ model, which offers an additional term to account for swirl turbulence resulting in reduced turbulent viscosity. This reduction would decrease the hindrance of the liquid flow, consequently leading to higher liquid velocities in the central region. However, no notable increase in liquid velocity profile had been observed, and no other notable improvements were observed. The RNG $k - \epsilon$ model was therefore not included in further models.
- The inclusion of the ideal gas law in the base model had a notable impact on the model's ability to accurately predict the hydrodynamics of $D=1.0\text{m}$, because the density increased by 36.4% at the sparger for this geometry. The incorporation of the ideal gas law had minimal effect on the hydrodynamics prediction for $D=0.4\text{m}$. Considering the fundamental nature of the ideal gas law it was included in the base model.
- The inclusion of NITA, turbulent dispersion, BIT, the Grace drag formulation, and the ideal gas law in the base model yielded substantial improvements, particularly for $D=1.0\text{m}$.

- The detailed model comparison revealed that the improved model exhibited instability issues when simulating bubble columns with diameters of 0.15m and 3.0m. Additionally, The model failed to accurately capture the scale-independent gas holdup profile, as well as dependence on the measuring height, which contradicts experimental observations.
- The implementation of a finer mesh resolution revealed that the model's performance was sensitive to the mesh size.
- The utilization of higher-order discretization for $D=1.0\text{m}$ yielded a notable improvement in the simulation outcomes. This came at the cost of convergence, as the model failed to reach the convergence threshold of $5e-5$. Consequently, the gas holdup profile exhibited inconsistencies, indicating a lack of accurate predictions.
- The inclusion of higher order discretization has been shown to result in good agreement with the results obtained by Ertekin [16] for $D=1.0\text{m}$.

This study aimed to explore the possibility of developing a single CFD model capable of accurately capturing the gas holdup and liquid velocity profiles, as well as the scale-independence phenomena observed in bubble columns with diameters ranging from 0.15m to 3.0m. The objective was to assess whether a single model could effectively simulate these hydrodynamic behaviors across various column diameters, thereby providing valuable insights for bubble column design and optimization. The findings of this study revealed that although the current model successfully simulated $D=0.4\text{m}$ and $D=1.0\text{m}$, it fell short in accurately reproducing the gas holdup and liquid velocity profiles, as well as the scale-independence phenomena.

This study establishes an initial step in the development of a single CFD model for simulating bubble columns with water and air, laying the foundation for future research. By incorporating the discussed models and providing the necessary code, this thesis offers a well-established starting point for investigating the instabilities and limitations of the current model. Through these endeavors, the ultimate objective of accurately simulating bubble columns for syngas fermentation can be progressively approached.

6 Recommendations

6.1 Validation

The validation process involved utilizing equations 2.7 and 2.8 to evaluate the model's capability to replicate the experimental setup. Raimundo [15] performed a comparative analysis between his experimentally derived data and these correlations, revealing a satisfactory level of agreement. However, it was observed that it had an average error of 5% [15]. Conversely, our investigation, employing the same correlations, led to the conclusion that the model inadequately reproduced the experimental outcomes. Notably, our findings exhibited concurrence with Ertikin's results [16], where she noted a favorable correspondence between her outcomes and the experimental data. Ertikin's assessment was based on her utilization of the experimental results for deriving her conclusions. Given the discrepancy regarding the model's proficiency in replicating experimental results, it is imperative to conduct the comparison using the actual experimental data, as this doesn't contain an error of 5%.

6.2 Measuring range

The data had been measured for a range between $\frac{r}{R} = -0.875$ and 0.875 . This range was selected because the simulation measurements of gas holdup were available exclusively within this interval. However, since a simulation is not limited to a specific range, and an empirical correlation had been selected for validation, it is recommended to measure the data within the range of $\frac{r}{R} = -1$ to 1 . In addition to gaining a better understanding of the gas holdup profile, it would have been possible to determine the average gas holdup, which is currently based on the measured range between $\frac{r}{R} - 0.875$ and 0.875 .

6.3 Gas density

The initial models assumed a constant gas density that was not related to the average pressure in the reactor. Consequently, there was a density difference of 37% at the sparger for $D=1.0\text{m}$. The incorporation of the ideal gas law revealed a notable disparity between the constant density assumption for this geometry and the ability to accurately reproduce the experimental setup. However, previous simulations that included the drag law occasionally resulted in unstable simulations. In such cases, a constant density model may be deemed acceptable if the constant density is derived from the average gas density within the reactor. This approach is expected to yield satisfactory results, based on the fact that the smaller reactors were less influenced by the ideal gas law model due to their proximity to the assumed constant density.

6.4 Simonnet model instability

Previous research incorporated the Simonnet model to account for the drag reduction resulting from the swarming behavior of bubbles, which demonstrated improved model performance. However, in this study, attempts to achieve stable simulations using the Simonnet model were unsuccessful. Consequently, the model could not be implemented and applied for analysis. To mitigate this instability, the time step was reduced from 0.001 sec to 0.0001 sec. Despite this adjustment, the model continued to be unstable, accompanied by an increase in simulation time. The rapid variability of the drag reduction factor between consecutive time steps likely contributes to this instability, posing challenges for achieving convergence. To address this issue, Fluent introduced relaxation factors, a method that limits the magnitude of change for specific variables [57], thereby enhancing simulation stability. This methodology can potentially be applied to the Simonnet model as well. Initially, a UDF implementing a constant drag reduction could be employed to establish a stable simulation and determine an initial drag reduction factor. Once a stable simulation state is attained,

the Simonnet model can be introduced using appropriate relaxation factors. The selection of an appropriate under relaxation factor involves an iterative process due to the absence of a universally accepted standard value. A recommended starting point is 0.5. By systematically adjusting this value by 0.2, changes in model stability can be observed, aiding in the determination of the optimal relaxation factor.

6.5 Further model improvement

The improved model with Quick discretization demonstrated the ability to reproduce results obtained by Ertekin [16] and displayed the most potential. However, this model encountered two primary issues: it could only be run for $D=1.0\text{m}$, and it exhibited limited convergence. Prior to incorporating additional models into the current simulation, it is preferred to enhance the existing model to meet convergence criteria. This could be achieved by minimizing the number of models included, focusing solely on those that have an influence on the results, such as the BIT, ideal gas law, and Quick discretization. By reducing the model complexity to these key factors, it would be possible to gain insights into the sources of model instability.

If convergence issues persist, it is advisable to conduct simulations without the BIT model. If the simulation shows improvement without the BIT model, further investigation can be directed towards the maximum slip condition, which is currently set at 5 m/s. As this value is arbitrarily chosen, modifying it by either increasing or decreasing could potentially resolve instabilities. Additionally, implementing alternative approaches, such as defining a maximum output for the bubble-induced turbulence, may help mitigate model instability.

References

- [1] Hairong Yue, Xinbin Ma, and Jinlong Gong. An alternative synthetic approach for efficient catalytic conversion of syngas to ethanol. *Accounts of Chemical Research*, 47(5):1483–1492, 5 2014.
- [2] Xiao Sun, Hasan K Atiyeh, Raymond L Huhnke, and Ralph S Tanner. Syngas fermentation process development for production of biofuels and chemicals: A review. 2019.
- [3] Fayetteville James L. Gaddy. Gas Fermentation-A Flexible Platform for Commercial Scale Production of Low-Carbon-Fuels and Chemicals from Waste and Renewable Feedstocks. *Frontiers in Microbiology* — *www.frontiersin.org*, 1:694, 2016.
- [4] Mark E. Dry. The Fischer-Tropsch process: 1950-2000. *Catalysis Today*, 71(3-4):227–241, 1 2002.
- [5] Haris Nalakath Abubackar, María C Veiga, and Christian Kennes Biofuels. Biological conversion of carbon monoxide: rich syngas or waste gases to bioethanol.
- [6] Konstantinos Asimakopoulos, Hariklia N. Gavala, and Ioannis V. Skiadas. Reactor systems for syngas fermentation processes: A review, 9 2018.
- [7] Marshall D. Bredwell, Prashant Srivastava, and R. Mark Worden. Reactor design issues for synthesis-gas fermentations. *Biotechnology Progress*, 15(5):834–844, 9 1999.
- [8] James J. Orgill, Hasan K. Atiyeh, Mamatha Devarapalli, John R. Phillips, Randy S. Lewis, and Raymond L. Huhnke. A comparison of mass transfer coefficients between trickle-bed, Hollow fiber membrane and stirred tank reactors. *Bioresource Technology*, 133:340–346, 2013.
- [9] Muhammad Yasin, Yeseul Jeong, Shinyoung Park, Jiyeong Jeong, Eun Yeol Lee, Robert W. Lovitt, Byung Hong Kim, Jinwon Lee, and In Seop Chang. Microbial synthesis gas utilization and ways to resolve kinetic and mass-transfer limitations. *Bioresource Technology*, 177:361–374, 2 2015.
- [10] J. J. Heijnen and K. Van't Riet. Mass transfer, mixing and heat transfer phenomena in low viscosity bubble column reactors. *The Chemical Engineering Journal*, 28(2), 1984.
- [11] Giorgio Besagni, Fabio Inzoli, Giorgia De Guido, and Laura Annamaria Pellegrini. Experimental investigation on the influence of ethanol on bubble column hydrodynamics. *Chemical Engineering Research and Design*, 112:1–15, 8 2016.
- [12] Lars Puiman, Marina P. Elisiário, Lilo M.L. Crasborn, Liselot E.C.H. Wagenaar, Adrie J.J. Straathof, and Cees Haringa. Gas mass transfer in syngas fermentation broths is enhanced by ethanol. *Biochemical Engineering Journal*, 185:108505, 7 2022.
- [13] Gunter Keitel and Ulfert Onken. Inhibition of bubble coalescence by solutes in air/water dispersions. *Chemical Engineering Science*, 37(11):1635–1638, 1982.
- [14] M. Jamialahmadi and H. Müller-Steinhagen. Effect of alcohol, organic acid and potassium chloride concentration on bubble size, bubble rise velocity and gas hold-up in bubble columns. *The Chemical Engineering Journal*, 50(1):47–56, 1992.
- [15] P. Maximiano Raimundo, A. Cloupet, A. Cartellier, D. Beneventi, and F. Augier. Hydrodynamics and scale-up of bubble columns in the heterogeneous regime: Comparison of bubble size, gas holdup and liquid velocity measured in 4 bubble columns from 0.15m to 3m in diameter. *Chemical Engineering Science*, 198:52–61, 4 2019.
- [16] Ege Ertekin, John M. Kavanagh, David F. Fletcher, and Dale D. McClure. Validation studies to assist in the development of scale and system independent CFD models for industrial bubble columns. *Chemical Engineering Research and Design*, 171:1–12, 7 2021.

- [17] David F. Fletcher, Dale D. McClure, John M. Kavanagh, and Geoffrey W. Barton. CFD simulation of industrial bubble columns: Numerical challenges and model validation successes. *Applied Mathematical Modelling*, 44:25–42, 4 2017.
- [18] Dale D. McClure, John M. Kavanagh, David F. Fletcher, and Geoffrey W. Barton. Development of a CFD model of bubble column bioreactors: Part two - comparison of experimental data and CFD predictions. *Chemical Engineering and Technology*, 37(1):131–140, 1 2014.
- [19] I. Katharina Stoll, Nikolaos Boukis, and Jörg Sauer. Syngas Fermentation to Alcohols: Reactor Technology and Application Perspective, 1 2020.
- [20] Mark R. Wilkins and Hasan K. Atiyeh. Microbial production of ethanol from carbon monoxide, 6 2011.
- [21] Dinesh K. A. Ko C. W. phillips J. R. Basu R. Wilkstrom C. V. et al. Gaddy, j.L. Methods for increasing the production of ethanol from microbial fermentation. *US7285402. Washington, DC: U.S. Patent and Trademark Office*, 2007.
- [22] Ánxela Fernández-Naveira, María C. Veiga, and Christian Kennes. H-B-E (hexanol-butanol-ethanol) fermentation for the production of higher alcohols from syngas/waste gas, 4 2017.
- [23] Daniele Pugliesi. Bubble column reactor, 7 2012.
- [24] Giorgio Besagni and Fabio Inzoli. The effect of liquid phase properties on bubble column fluid dynamics: Gas holdup, flow regime transition, bubble size distributions and shapes, interfacial areas and foaming phenomena. *Chemical Engineering Science*, 170:270–296, 10 2017.
- [25] R. Krishna and J. M. Van Baten. Scaling up bubble column reactors with the aid of CFD. *Chemical Engineering Research and Design*, 79(3):283–309, 2001.
- [26] A. Forret, J. M. Schweitzer, T. Gauthier, R. Krishna, and D. Schweich. Influence of scale on the hydrodynamics of bubble column reactors: An experimental study in columns of 0.1, 0.4 and 1 m diameters. *Chemical Engineering Science*, 58(3-6):719–724, 2003.
- [27] R Krishna and S T Sie. Design and scale-up of the Fischer-Tropsch bubble column slurry reactor. Technical report, 2000.
- [28] Robert F. Mudde. Gravity-driven bubbly flows. *Annual Review of Fluid Mechanics*, 37:393–423, 2005.
- [29] Xiangang Li, Derek Griffin, Xueliang Li, and Michael A. Henson. Incorporating hydrodynamics into spatiotemporal metabolic models of bubble column gas fermentation. *Biotechnology and Bioengineering*, 116(1):28–40, 1 2019.
- [30] Graham B Wallis. *The terminal speed of single drops or bubbles in an infinite medium*, volume ! Pergamon Press, 1974.
- [31] J.R. Grace. Shapes and velocities of bubbles rising in infinite liquids. *Transactions of the Institution of Chemical Engineers*, pages 116–120, 1973.
- [32] E. Dinesh Kumar, S. A. Sannasiraj, and V. Sundar. Phase field lattice Boltzmann model for air-water two phase flows. *Physics of Fluids*, 31(7), 7 2019.
- [33] Sung Hoon Park, Changhwan Park, Jin Yong Lee, and Byungchul Lee. A Simple Parameterization for the Rising Velocity of Bubbles in a Liquid Pool. *Nuclear Engineering and Technology*, 49(4):692–699, 6 2017.
- [34] J. M. Schweitzer, J. Bayle, and T. Gauthier. Local gas hold-up measurements in fluidized bed and slurry bubble column. *Chemical Engineering Science*, 56(3):1103–1110, 2 2001.
- [35] Peter M. Wilkinson, Arie P. Spek, and Laurent L. van Dierendonck. Design parameters estimation for scale-up of high-pressure bubble columns. *AIChE Journal*, 38(4):544–554, 1992.

- [36] Giorgio Besagni, Lorenzo Gallazzini, and Fabio Inzoli. On the scale-up criteria for bubble columns. *Petroleum*, 5(2):114–122, 6 2019.
- [37] ANSYS Fluent Theory Guide 15.
- [38] Roland Rzehak and Sebastian Kriebitzsch. Multiphase CFD-simulation of bubbly pipe flow: A code comparison. *International Journal of Multiphase Flow*, 68:135–152, 1 2015.
- [39] Mohan R. Rampure, Amol A. Kulkarni, and Vivek V. Ranade. Hydrodynamics of bubble column reactors at high gas velocity: Experiments and computational fluid dynamics CFD simulations. In *Industrial and Engineering Chemistry Research*, volume 46, pages 8431–8447, 12 2007.
- [40] Akio Tomiyama, Isao Kataoka, Iztok Zun, and Tadashi Sakaguchi. Drag coefficients of single bubbles under normal and micro gravity conditions. *JSME International Journal, Series B: Fluids and Thermal Engineering*, 41(2):472–479, 1998.
- [41] M. Simonnet, C. Gentric, E. Olmos, and N. Midoux. CFD simulation of the flow field in a bubble column reactor: Importance of the drag force formulation to describe regime transitions. *Chemical Engineering and Processing: Process Intensification*, 47(9-10):1726–1737, 2008.
- [42] E. Olmos, C. Gentric, and N. Midoux. Numerical description of flow regime transitions in bubble column reactors by a multiple gas phase model. *Chemical Engineering Science*, 58(10):2113–2121, 2003.
- [43] Alan D Burns, Thomas Frank, Ian Hamill, and Jun-Mei Shi. The Favre Averaged Drag Model for Turbulent Dispersion in Eulerian Multi-Phase Flows. Technical report.
- [44] Launder and Spalding. The mathematical modelling of turbulent flows. 1985.
- [45] Cédric Laborde-Boutet, Faïçal Larachi, Nicolas Dromard, Olivier Delsart, and Daniel Schweich. CFD simulation of bubble column flows: Investigations on turbulence models in RANS approach. *Chemical Engineering Science*, 64(21):4399–4413, 11 2009.
- [46] Y Sato, M Sadatomi, and K Sekoguchi. Momentum and heat transfer in two-phase bubble flow. Technical report, 1981.
- [47] D Peger and S Becker. Modelling and simulation of the dynamic flow behaviour in a bubble column. Technical report, 2001.
- [48] M. Lopez de Bertodano, R. T. Lahey Jr, and O. C. Jones. Phase distribution in bubbly two-phase flow. *International Journal of Multiphase Flow*, 20(5):805–818, 1994.
- [49] L. Kataoka and A. Serizawa. Basic equations of turbulence in gas-liquid two-phase flow. *International Journal of Multiphase Flow*, 15(5):843–855, 1989.
- [50] A A Troshko and Y A Hassan. A two-equation turbulence model of turbulent bubbly flows. Technical report, 2001.
- [51] Wei Yao and Christophe Morel. Volumetric interfacial area prediction in upward bubbly two-phase flow. *International Journal of Heat and Mass Transfer*, 47(2):307–328, 2004.
- [52] Yohana Ezzat and Ahmed A. Abdel-Rehim. Numerical modelling of lauric acid phase change material using iterative and non-iterative time-advancement schemes. *Journal of Energy Storage*, 53, 9 2022.
- [53] ANSYS Fluent Mosaic Technology Automatically Combines Disparate Meshes with Polyhedral Elements for Fast, Accurate Flow Resolution. Technical report, 2018.
- [54] Dale D. McClure, Hannah Norris, John M. Kavanagh, David F. Fletcher, and Geoffrey W. Barton. Validation of a computationally efficient computational fluid dynamics (CFD) model for industrial bubble column bioreactors. *Industrial and Engineering Chemistry Research*, 53(37):14526–14543, 8 2014.

- [55] R. Clift, J. R. Grace, and M. E. Weber. Bubbles, Drops and Particles. *Academic Press: New York*, 1978.
- [56] S Becker, H De Bie, and J Sweeney. Dynamic flow behaviour in bubble columns. *Chemical Engineering Science*, 54:10591–6714, 1999.
- [57] Ansys Fluent Customization Manual. Technical report, 2021.
- [58] Bengt Andersson. *Computational fluid dynamics for engineers*. Cambridge University Press, 2011.
- [59] B. P. Welford. Note on a Method for Calculating Corrected Sums of Squares and Products. *Technometrics*, 4(3):419–420, 1962.
- [60] Mandar V. Tabib, Swarnendu A. Roy, and Jyeshtharaj B. Joshi. CFD simulation of bubble column-An analysis of interphase forces and turbulence models. *Chemical Engineering Journal*, 139(3):589–614, 6 2008.
- [61] Nigar Kantarci, Fahir Borak, and Kutlu O. Ulgen. Bubble column reactors, 6 2005.
- [62] Giorgio Besagni, Nicolò Varallo, and Riccardo Mereu. Computational Fluid Dynamics Modelling of Two-Phase Bubble Columns: A Comprehensive Review. *Fluids*, 8(3):91, 3 2023.

Appendix A - Experimental validation Raimundo

This section contains the experimental scale-independent gas holdup profiles from Raimundo[15], from which the empirical gas holdup correlation by Schweitzer has been validated [34]. As well as the validation of the liquid velocity scale-independent correlation by Forret [26].

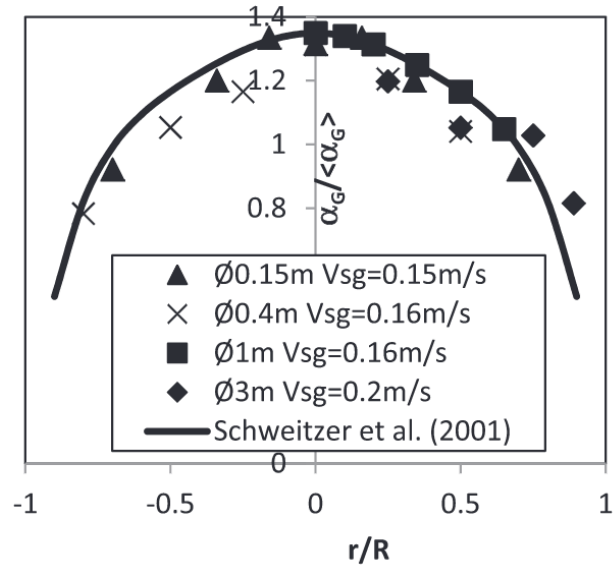


Figure 34: Gas holdup profiles at $H/D = 2.5$ obtained by Raimundo, normalized by the average gas holdup with the empirical correlation (Equation 2.7)

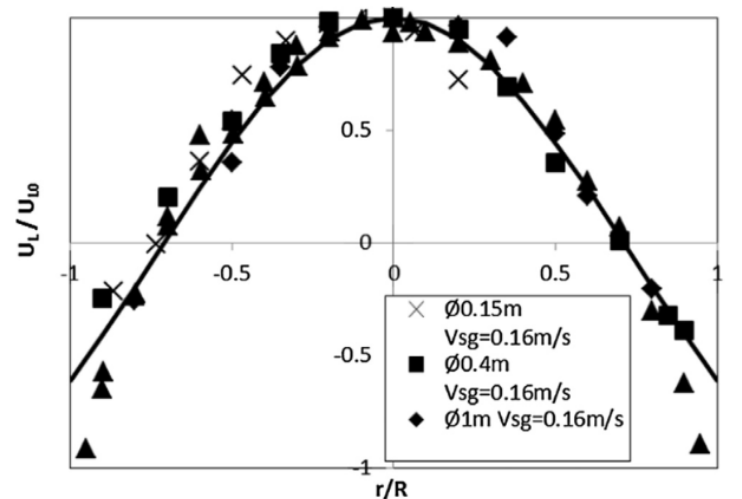


Figure 35: Normalized liquid velocity profile obtained by Raimundo and the empirical normalized liquid velocity profile (equation 2.8)

Appendix B: Overview of the ITA and NITA iterative schemes

This section contains an overview of both iterative schemes employed in this study.

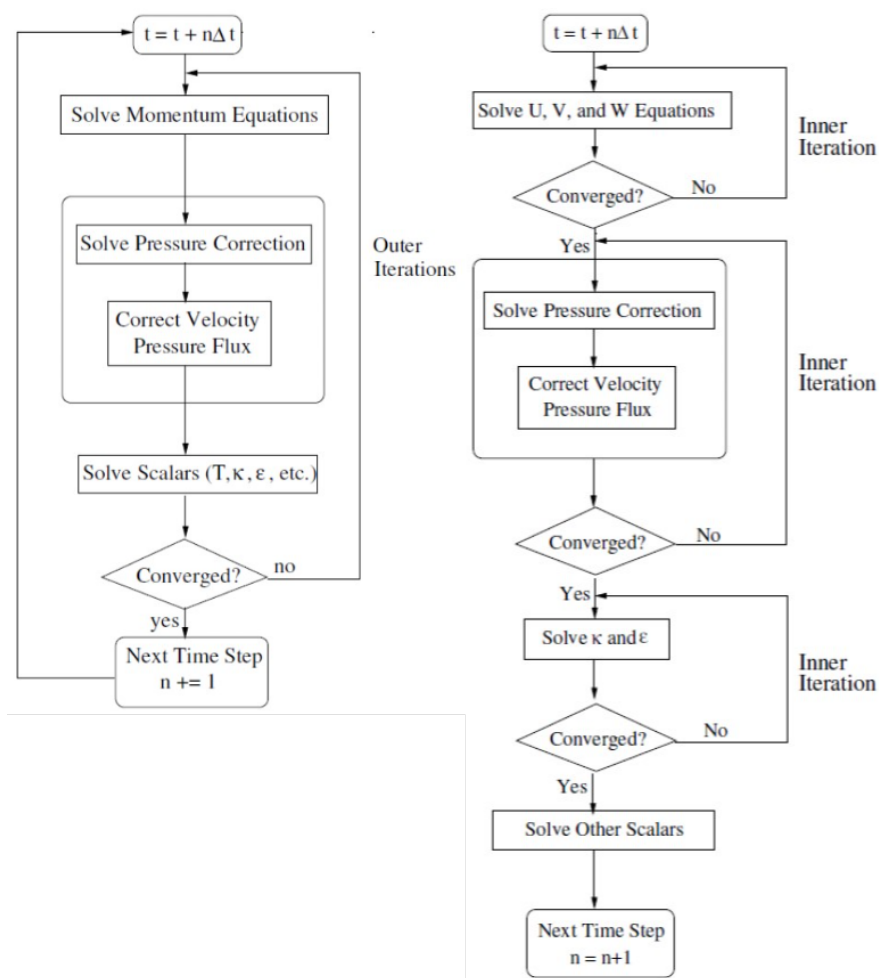


Figure 36: Overview of the ITA and NITA iterative schemes [37].

Appendix C: Reactor geometry

This section displays the geometry of the reactor with a diameter of 1m.

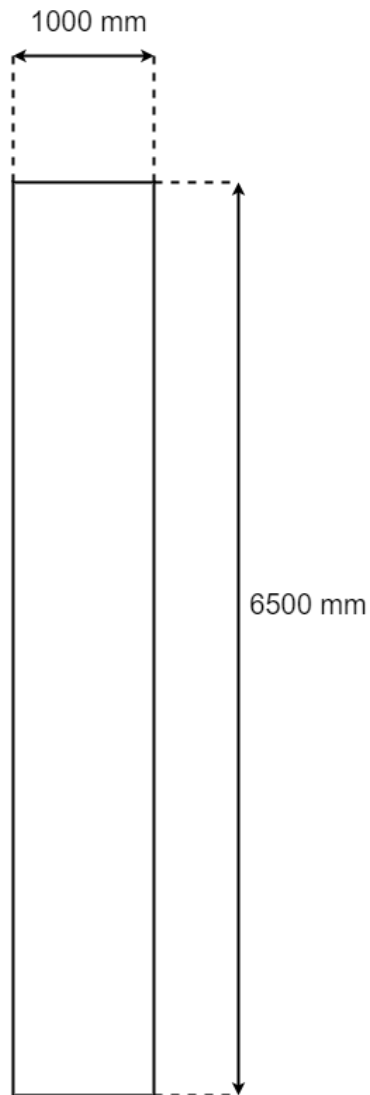


Figure 37: Schematic overview of the reactor with a diameter of 1m.

Appendix D: Meshing

This section contains the meshes used to model $D=1.0\text{m}$ and $D=0.4\text{m}$ as well as the mesh of $D=1.0\text{m}$ used in the mesh independence study.

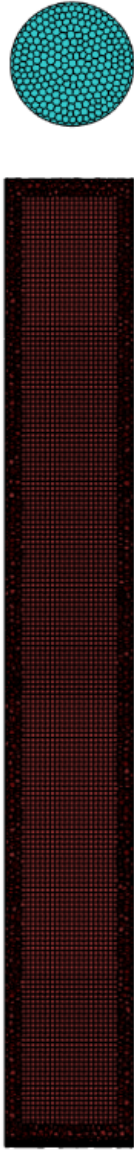
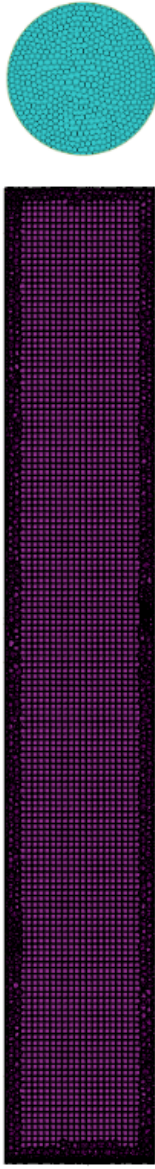
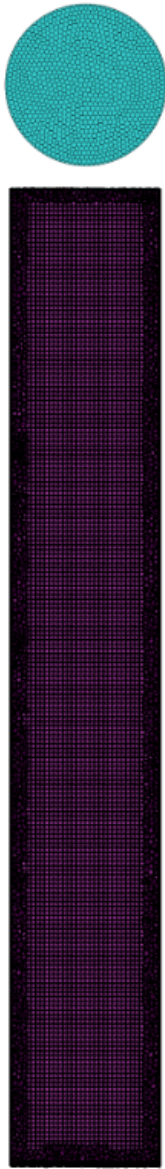
	D=0.4	Coarser D=1.0m	Finer D=1.0m
Mesh			
Number of cells	118577	96752	207651
Mesh quality	0.457	0.483	0.463

Figure 38: Overview of the meshes for $D=0.4$ and $D=1.0$, as well as the mesh used for the mesh independancy study

Appendix E: User-Defined functions

This section introduces the UDFs employed in this study, along with the locations of the UDMIs (table 28). The provided codes are; the Simonnet drag reduction factor UDF, the time-averaging UFD, and the BIT UDFs.

UDMI

UDMI	Phase	Var	description	unit	UDF
0	Liquid	u velocity	average	m/s	time_avg
1	Liquid	v velocity	average	m/s	time_avg
2	Liquid	w velocity	average	m/s	time_avg
3	Gas	u velocity	average	m/s	time_avg
4	Gas	v velocity	average	m/s	time_avg
5	Gas	w velocity	average	m/s	time_avg
6	Liquid	u velocity	stdev	m/s	time_avg
7	Liquid	v velocity	stdev	m/s	time_avg
8	Liquid	w velocity	stdev	m/s	time_avg
9	Gas	u velocity	stdev	m/s	time_avg
10	Gas	v velocity	stdev	m/s	time_avg
11	Gas	w velocity	stdev	m/s	time_avg
12	Liquid	Turb kin energy	average	m ² /s ²	time_avg
13	Liquid	Turb energy dissipation	average	m ² /s ³	time_avg
14	Gas	Volume fraction	average		time_avg
15	Liquid	Turb kin energy	average	m ² /s ²	time_avg
16	Liquid	Turb kin energy	stdev	m ² /s ³	time_avg
17	Gas	Volume fraction	stdev		time_avg
18	liquid & gas	Slip velocity	average	m/s	time_avg
19	liquid & gas	Slip velocity	instantaneous	m/s	time_avg
20					
21					
22	liquid	K_source	instantaneous		eps_source
23	liquid	eps_source	instantaneous		eps_source
24	liquid	CD_swarm	instantaneous		eps_source
25	liquid	CD	instantaneous		eps_source
26	liquid	CD_factor	instantaneous		eps_source
27	liquid	F_drag	instantaneous		eps_source
28	liquid	tau	instantaneous		eps_source
29	liquid	Rey	instantaneous		eps_source

Table 28: Overview of the UDMI memory locations used for the time-averaging, Simonnet drag reduction factor and the BIT UDFs.

Simonnet drag reduction factor

```
1  /*****
2  UDF for Simonnet's drag modification in the heterogeneous regime
3  *****/
4
5  #include "udf.h"
6  #include "math.h"
7  #include "mem.h"
8
9  DEFINE_EXCHANGE_PROPERTY(Simonnet_modification, cell, mix_thread, s_col, f_col)
10 {
11
12  /* Variables */
13  Thread *thread_g;          /* Gas and liquid threads */
14  real CD_correction;
15
16  /* Get the Threads */
17  thread_g = THREAD_SUB_THREAD(mix_thread, f_col); /* Gas phase */
18
19  /* gas holdup */
20  real alpha_g = C_VOF(cell, thread_g);
21
22  /* exponent */
23  real m = 25.0;
24
25  /* CD correction term */
26  real C1 = 0.8;
27
28  /*model stability*/
29  alpha_g = MAX(0.01, alpha_g);
30
31  /* Compute drag coefficient correction */
32  real CD_factor = (1-alpha_g)* pow((pow((1-alpha_g),m)+pow((4.8*(alpha_g/(1-alpha_g))),m))
33  ,(-2/m));
34
35  /* correct CD_correction */
36  if (CD_factor < 1)
37  {
38    CD_correction = C1 * CD_factor;
39  }
40  else
41  {
42    CD_correction = 1;
43  }
44
45  return CD_correction;
46 }
```

Listing 1: modified Simonnet drag modification

Time-averaging UDF

```
1 /*function to compute a rolling average in a multi-phase process*/
2
3 #include "udf.h"      /* these are the packages to be imported */
4 #include "dpm.h"     /* these are the packages to be imported */
5 #include "mem.h"     /* these are the packages to be imported */
6
7 /*definitions */
8 int timer = 1;      /* number of timesteps, initialize at 1 */
9
10 /* executes at end of every timestep */
11 DEFINE_EXECUTE_AT_END(Time_average_std)
12 {
13     Domain *d;          /* Load domain */
14     cell_t cell;       /* will loop over all cells c */
15     Thread *tr;        /* will loop over all threads tr */
16     double BU, M2;     /* backup variable for computation */
17     double slip,slip_U,slip_V,slip_W; /* variable for slip velocity*/
18
19     /* if multiphase specify liquid and gas thread. */
20     Thread *SUBT_Liq;  /* sub-thread for liquid */
21     Thread *SUBT_Gas;  /* sub-thread for gas */
22     d = Get_Domain(1);
23
24     /* loop over all cells to get average and std data per cell */
25     thread_loop_c (tr,d)
26     {
27
28         SUBT_Liq = THREAD_SUB_THREAD(tr,0); /* set liquid thread */
29         SUBT_Gas = THREAD_SUB_THREAD(tr,1); /* set gas thread */
30
31         /* this list can be made arbitrarily long. Make sure to refer to right thread (for
32         epsilon,
33         in mixture k-epsilon mode this is the mixture level thread tr, in dispersed mode this is
34         the liquid thread)
35         make sure to include UDM in simulation! */
36
37         begin_c_loop (cell,tr)
38         {
39             /* velocity averages & stdev, liquid */
40             BU = C_UDMI(cell,tr,0);
41             C_UDMI(cell,tr,0) = (C_UDMI(cell,tr,0)*(timer-1) + C_U(cell,SUBT_Liq))/(timer) ;
42             /* average liquid_vel u direction */
43             M2 = C_UDMI(cell,tr,6)+((C_U(cell,SUBT_Liq) - BU)*(C_U(cell,SUBT_Liq)-C_UDMI(cell,tr,0))
44             ); /* standard deviation */
45             C_UDMI(cell,tr,6) = M2/timer;
46
47             BU = C_UDMI(cell,tr,1);
48             C_UDMI(cell,tr,1) = (C_UDMI(cell,tr,1)*(timer-1) + C_V(cell,SUBT_Liq))/(timer) ;
49             /* liquid_vel v */
50             M2 = C_UDMI(cell,tr,7)+((C_V(cell,SUBT_Liq) - BU)*(C_V(cell,SUBT_Liq)-C_UDMI(cell,tr,1))
51             );
52             C_UDMI(cell,tr,7) = M2/timer;
53
54             BU = C_UDMI(cell,tr,2);
55             C_UDMI(cell,tr,2) = (C_UDMI(cell,tr,2)*(timer-1) + C_W(cell,SUBT_Liq))/(timer) ;
56             /* liquid_vel w */
57             M2 = C_UDMI(cell,tr,8)+((C_W(cell,SUBT_Liq) - BU)*(C_W(cell,SUBT_Liq)-C_UDMI(cell,tr,2))
58             );
59             C_UDMI(cell,tr,8) = M2/timer;
60
61             /* velocity averages & stdev, gas */
62             BU = C_UDMI(cell,tr,3);
63             C_UDMI(cell,tr,3) = (C_UDMI(cell,tr,3)*(timer-1) + C_U(cell,SUBT_Gas))/(timer) ;
64             /* average gas_vel u */
65             M2 = C_UDMI(cell,tr,9)+((C_U(cell,SUBT_Gas) - BU)*(C_U(cell,SUBT_Gas)-C_UDMI(cell,tr,3))
66             ); /* std dev */
```

```

57 C_UDMI(cell, tr, 9) = M2/timer;
58
59 BU = C_UDMI(cell, tr, 4);
60 C_UDMI(cell, tr, 4) = (C_UDMI(cell, tr, 4)*(timer-1) + C_V(cell, SUBT_Gas))/(timer) ;
61 /* gas vel v */
62 M2 = C_UDMI(cell, tr, 10)+((C_V(cell, SUBT_Gas) - BU)*(C_V(cell, SUBT_Gas)-C_UDMI(cell, tr, 4)
63 ));
64 C_UDMI(cell, tr, 10) = M2/timer;
65
66 BU = C_UDMI(cell, tr, 5);
67 C_UDMI(cell, tr, 5) = (C_UDMI(cell, tr, 5)*(timer-1) + C_W(cell, SUBT_Gas))/(timer) ;
68 /* gas vel w */
69 M2 = C_UDMI(cell, tr, 11)+((C_W(cell, SUBT_Gas) - BU)*(C_W(cell, SUBT_Gas)-C_UDMI(cell, tr, 5)
70 ));
71 C_UDMI(cell, tr, 11) = M2/timer;
72
73 /* turbulence averages & stdev, gas */
74 BU = C_UDMI(cell, tr, 12);
75 C_UDMI(cell, tr, 12) = (C_UDMI(cell, tr, 12)*(timer-1) + C_K(cell, SUBT_Liq))/(timer) ;
76 /* turb kinetic energy - average */
77 M2 = C_UDMI(cell, tr, 15)+((C_K(cell, SUBT_Liq) - BU)*(C_K(cell, SUBT_Liq)-C_UDMI(cell, tr
78 , 12))); /* turb kinetic energy - stdev */
79 C_UDMI(cell, tr, 15) = M2/timer;
80
81 BU = C_UDMI(cell, tr, 13);
82 C_UDMI(cell, tr, 13) = (C_UDMI(cell, tr, 13)*(timer-1) + C_D(cell, SUBT_Liq))/(timer) ;
83 /* Turb energy dissipation rate - average */
84 M2 = C_UDMI(cell, tr, 16)+((C_D(cell, SUBT_Liq) - BU)*(C_D(cell, SUBT_Liq)-C_UDMI(cell, tr
85 , 13))); /* Turb energy dissipation rate - stdev */
86 C_UDMI(cell, tr, 16) = M2/timer;
87
88 BU = C_UDMI(cell, tr, 14);
89 C_UDMI(cell, tr, 14) = (C_UDMI(cell, tr, 14)*(timer-1) + C_VOF(cell, SUBT_Gas))/(timer) ;
90 /* Gas fraction */
91 M2 = C_UDMI(cell, tr, 17)+((C_VOF(cell, SUBT_Gas) - BU)*(C_VOF(cell, SUBT_Gas)-C_UDMI(cell,
92 tr, 14)));
93 C_UDMI(cell, tr, 17) = M2/timer;
94
95 /*slip velocity*/
96 slip_U = C_U(cell, SUBT_Gas) - C_U(cell, SUBT_Liq);
97 slip_V = C_V(cell, SUBT_Gas) - C_V(cell, SUBT_Liq);
98 slip_W = C_W(cell, SUBT_Gas) - C_W(cell, SUBT_Liq);
99 slip = pow(pow(slip_U, 2.)+pow(slip_V, 2.)+pow(slip_W, 2.)), (1./2.);
100
101 /*averaging & t slip*/
102 BU = C_UDMI(cell, tr, 18);
103 C_UDMI(cell, tr, 18) = (C_UDMI(cell, tr, 18)*(timer-1) + slip)/(timer);
104 C_UDMI(cell, tr, 19) = slip;
105
106 }
107 end_c_loop (cell, tr)
108 }
109 timer = timer+1;
110 }

```

Listing 2: Time-averaging UDF

BIT UDF

The BIT model includes the modeling of the momentum transfer term, which has been implemented modeled with two drag formulations and two drag modifications resulting in four distinct codes. For redundancy purposes, only two codes have been added: BIT with Tomiyama and a constant drag formulation, and BIT with Grace drag formulation and Simonnet drag modification. The remaining two codes can be derived from these.

BIT with Tomiyama and constant drag correction

```
1  /*****
2   UDF modeling BIT using source terms
3   *****/
4
5  #include "udf.h"
6  #include "math.h"
7  #include "mem.h"
8  #include "stdlib.h"
9  #include "sg_mphase.h"
10
11 DEFINE_SOURCE(k_BIT, cell, phase_thread, dS, eqn)
12 {
13     /* variables */
14     Thread *Thread_GL;          /*thread to receive all phase threads*/
15     Thread *Thread_L;          /* sub-thread for liquid */
16     Thread *Thread_G;          /* sub-thread for gas */
17
18     /*position variable*/
19     real alpha_g, Rey, CD, Eo;
20
21     /*momentum transfer*/
22     real F_drag;
23
24     /*velocity vector*/
25     real slip, slip_U, slip_V, slip_W;
26
27     /*properties*/
28     real rho_g, rho_l;
29     real mu_L, diab, surf_ten;
30
31     /*storage*/
32     real min1, min2, max1, max2;
33
34     /*simonnet drag correction*/
35     real CD_correction, CD_swarm;
36
37     /*result*/
38     real k_source;
39
40     /*constants*/
41     real C1 = 1.44;
42     real max_slip = 5.;
43
44     /*gas & liquid threads*/
45     Thread_GL = THREAD_SUPER_THREAD(phase_thread); /*liquid & gas thread*/
46     Thread_L = THREAD_SUB_THREAD(Thread_GL, 0); /* set liquid thread */
47     Thread_G = THREAD_SUB_THREAD(Thread_GL, 1); /* set gas thread */
48
49     /*properties*/
50     rho_l = C_R(cell, Thread_L);
51     rho_g = C_R(cell, Thread_G);
52     mu_L = C_MU_L(cell, Thread_L);
53     alpha_g = C_VOF(cell, Thread_G);
54     diab = 5.1e-3; /*m*/
55     surf_ten = 0.072; /*N/m*/
56 }
```

```

57  /*slip velocity*/
58  slip_U = C_U(cell,Thread_G) - C_U(cell,Thread_L);
59  slip_V = C_V(cell,Thread_G) - C_V(cell,Thread_L);
60  slip_W = C_W(cell,Thread_G) - C_W(cell,Thread_L);
61
62  /*vector velocity*/
63  slip    = pow(pow(slip_U,2.)+pow(slip_V,2.)+pow(slip_W,2.),(1./2.));
64  slip    = MIN(slip,max_slip);
65
66  /*Tomiya drag model*/
67  Rey = (rho_l * fabs(slip) * diab) / mu_L;
68  Eo = (9.81*(rho_l-rho_g)*pow(diab,2.))/surf_ten ;
69
70  /*Drag coefficient tomyama*/
71  min1 = (24/Rey)*(1+0.15*pow(Rey,0.687));
72  min2 = (72/Rey);
73  max1 = (8./3.)*(Eo/(Eo+4));
74
75  max2 = MIN(min1,min2);
76  CD    = MAX(max1,max2);
77
78  /*Constant drag modification*/
79  CD_correction = 0.12;
80
81  /*correcting drag*/
82  CD_swarm = CD*CD_correction;
83
84  /*drag force - interfacial momentum transfer */
85  F_drag = (3./4.) * (CD_swarm/diab) * alpha_g * rho_l *(slip)*fabs(slip);
86
87  /* turbulent dissipation energy epsilon source term*/
88  k_source = (1.-alpha_g)*C1*fabs(F_drag)*fabs(slip);
89
90  /*force explicit solution*/
91  dS[eqn] = 0.;
92
93  return k_source;
94 }

```

Listing 3: BIT kinetic energy source term with Tomiyama and a constant drag modification

```

1  /*****
2   UDF modeling BIT using source terms
3   *****/
4
5  #include "udf.h"
6  #include "math.h"
7  #include "mem.h"
8  #include "stdlib.h"
9  #include "sg_mphase.h"
10
11 DEFINE_SOURCE(eps_BIT, cell, phase_thread, dS, eqn)
12 {
13     /* variables */
14     Thread *Thread_GL; /*thread to receive all phase threads*/
15     Thread *Thread_L; /* sub-thread for liquid */
16     Thread *Thread_G; /* sub-thread for gas */
17
18     /*position variable*/
19     real tau, alpha_g, Rey, CD, Eo, eps_B, k_source;
20
21     /*momentum transfer*/
22     real F_drag;
23
24     /*velocity vector*/
25     real slip, slip_U, slip_V, slip_W;
26
27     /*properties*/
28     real rho_g, rho_l;

```

```

28 real mu_L, diab, surf_ten;
29
30 /*storage*/
31 real min1,min2,max1,max2;
32
33 /*simonnet drag correction*/
34 real CD_correction, CD_swarm;
35
36 /*result*/
37 real eps_source;
38
39 /*constants*/
40 real C1      = 1.44;
41 real C3      = 1.;
42 real max_slip = 5.;
43
44 /*gas & liquid threads*/
45 Thread_GL = THREAD_SUPER_THREAD(phase_thread); /*liquid & gas thread*/
46 Thread_L   = THREAD_SUB_THREAD(Thread_GL,0);   /* set liquid thread */
47 Thread_G   = THREAD_SUB_THREAD(Thread_GL,1);   /* set gas thread */
48
49 /*properies*/
50 rho_l      = C_R(cell,Thread_L);
51 rho_g      = C_R(cell,Thread_G);
52 mu_L       = C_MU_L(cell,Thread_L);
53 eps_B      = C_UDMI(cell,Thread_GL,23);
54 alpha_g    = C_VOF(cell,Thread_G);
55 diab       = 5.1e-3; /*m*/
56 surf_ten   = 0.072; /*N/m*/
57
58 /*check if BIT has initial value*/
59 eps_B      = MAX(eps_B,1.185928e-09);
60
61 /*slip velocity*/
62 slip_U     = C_U(cell,Thread_G) - C_U(cell,Thread_L);
63 slip_V     = C_V(cell,Thread_G) - C_V(cell,Thread_L);
64 slip_W     = C_W(cell,Thread_G) - C_W(cell,Thread_L);
65
66 /*vector velocity*/
67 slip       = pow(pow(slip_U,2.)+pow(slip_V,2.)+pow(slip_W,2.),(1./2.));
68 slip       = MIN(max_slip,slip);
69
70 /*Tomiya drag model*/
71 Rey        = (rho_l * fabs(slip) * diab) / mu_L;
72 Eo         = (9.81*(rho_l-rho_g)*pow(diab,2.))/surf_ten ;
73
74 /*Drag coefficient tomyama*/
75 min1       = (24/Rey)*(1+0.15*pow(Rey,0.687));
76 min2       = (72/Rey);
77 max1       = (8./3.)*(Eo/(Eo+4));
78
79 max2       = MIN(min1,min2);
80 CD         = MAX(max1,max2);
81
82 /*Constant drag modification*/
83 CD_correction = 0.12;
84
85 /*correcting drag*/
86 CD_swarm   = CD*CD_correction;
87
88 /*drag force - interfacial momentum transfer */
89 F_drag     = (3./4.) * (CD_swarm/diab) * alpha_g * rho_l *(slip)*fabs(slip);
90
91 /* timescale dissipation */
92 tau        = pow((pow(diab,2)/(eps_B)),(1./3.));
93
94 /*k_source*/
95 k_source   = (1.-alpha_g)*C1*fabs(F_drag)*fabs(slip);

```

```

96
97  /* turbulent dissipation energy epsilon source term*/
98  eps_source = (C3/tau)*k_source;
99  eps_source = MIN(eps_source,1e8);
100
101  /*storage*/
102  C_UDMI(cell,Thread_GL,22) = k_source;
103  C_UDMI(cell,Thread_GL,23) = eps_source;
104  C_UDMI(cell,Thread_GL,24) = CD_swarm;
105  C_UDMI(cell,Thread_GL,25) = CD;
106  C_UDMI(cell,Thread_GL,26) = CD_correction;
107  C_UDMI(cell,Thread_GL,27) = F_drag;
108  C_UDMI(cell,Thread_GL,28) = tau;
109  C_UDMI(cell,Thread_GL,29) = Rey;
110
111  /*force explicit solution*/
112  dS[eqn] = 0.;
113
114  return eps_source;
115 }

```

Listing 4: BIT energy dissipation source term with Tomiyama and a constant drag modification

BIT with Grace and Simonnet drag correction

```

1  /*****
2  UDF modeling BIT using source terms
3  *****/
4
5  #include "udf.h"
6  #include "math.h"
7  #include "mem.h"
8  #include "stdlib.h"
9  #include "sg_mphase.h"
10
11 DEFINE_SOURCE(k_BIT, cell, phase_thread, dS, eqn)
12 {
13     /* variables */
14     Thread *Thread_GL;          /*thread to receive all phase threads*/
15     Thread *Thread_L;          /* sub-thread for liquid */
16     Thread *Thread_G;          /* sub-thread for gas */
17
18     /*position variable*/
19     real alpha_g, Rey, CD, Eo, Mo, H, J, Ut;
20     real CD_cap, CD_ell, CD_sph;
21
22     /*momentum transfer*/
23     real F_drag;
24
25     /*velocity vector*/
26     real slip, slip_U, slip_V, slip_W;
27
28     /*properties*/
29     real rho_g, rho_l;
30     real mu_L, diab, surf_ten;
31
32     /*storage*/
33     real max2;
34
35     /*simonnet drag correction*/
36     real CD_factor, CD_swarm;
37     real m = 25.;
38
39     /*result*/
40     real k_source;
41
42     /*constants*/
43     real C1 = 1.44;

```

```

44 real max_slip = 5.;
45
46 /*gas & liquid threads*/
47 Thread_GL = THREAD_SUPER_THREAD(phase_thread); /*liquid & gas thread*/
48 Thread_L = THREAD_SUB_THREAD(Thread_GL,0); /* set liquid thread */
49 Thread_G = THREAD_SUB_THREAD(Thread_GL,1); /* set gas thread */
50
51 /*properies*/
52 rho_l = C_R(cell,Thread_L);
53 rho_g = C_R(cell,Thread_G);
54 mu_L = C_MU_L(cell,Thread_L);
55 alpha_g = C_VOF(cell,Thread_G);
56 diab = 5.1e-3; /*m*/
57 surf_ten= 0.072; /*N/m*/
58
59 /*slip velocity*/
60 slip_U = C_U(cell,Thread_G) - C_U(cell,Thread_L);
61 slip_V = C_V(cell,Thread_G) - C_V(cell,Thread_L);
62 slip_W = C_W(cell,Thread_G) - C_W(cell,Thread_L);
63
64 /*vector velocity*/
65 slip = pow(pow(slip_U,2.)+pow(slip_V,2.)+pow(slip_W,2.),(1./2.));
66 slip = MIN(slip,max_slip);
67
68 /*reynolds*/
69 Rey = (rho_l * fabs(slip) * diab) / mu_L;
70
71 /*Eo*/
72 Eo = (9.81*(rho_l-rho_g)*pow(diab,2.))/surf_ten ;
73
74 /*Mo*/
75 Mo = (pow(mu_L,4.)*9.81*(rho_l-rho_g))/(pow(rho_l,2.)*pow(surf_ten,3.));
76
77 /*H*/
78 H = (4./3.)*Eo*pow(Mo,-0.149)*pow((mu_L/0.0009),-0.14);
79
80 /*J*/
81 if (H > 2. && H <= 59.2)
82 {
83     J = 0.94 * pow(H,0.757);
84 }
85 else if(H>59.2)
86 {
87     J = 3.42 * pow(H,0.441);
88 }
89 else
90 {
91     Message("Error H<2");
92 }
93
94 /*terminal rise velocity*/
95 Ut = (mu_L/(diab*rho_l))*pow(Mo,-0.149)*(J-0.857);
96
97 /*Drag coefficient grace*/
98 if (Rey <0.01)
99 {
100     CD_sph = 24/Rey;
101 }
102 else
103 {
104     CD_sph = (24/Rey)*(1+0.15*pow(Rey,0.687));
105 }
106
107 CD_ell = (4./3.) * ((9.81*diab)/pow(Ut,2))*((rho_l-rho_g)/rho_l);
108
109 CD_cap = (8./3.);
110
111 /*taking minimum and maximum*/

```

```

112     max2      = MIN(CD_ell,CD_cap);
113     CD        = MAX(CD_sph,max2);
114
115     /* Simonnet drag coefficient correction */
116     CD_factor = (1.-alpha_g)* pow((pow((1.-alpha_g),m)+pow((4.8*(alpha_g/(1-alpha_g))),m))
117         ,(-2./m));
118
119     /* correct CD_correction */
120     if (CD_factor < 1)
121     {
122         CD_swarm = 0.8 * CD_factor*CD;
123     }
124     else
125     {
126         CD_swarm = CD;
127     }
128
129     /*drag force - interfacial momentum transfer */
130     F_drag = (3./4.) * (CD_swarm/diab) * alpha_g * rho_l *(slip)*fabs(slip);
131
132     /* turbulent dissipation energy epsilon source term*/
133     k_source = (1.-alpha_g)*C1*fabs(F_drag)*fabs(slip);
134
135     /*force explicit solution*/
136     dS[eqn] = 0.;
137
138     return k_source;
139 }

```

Listing 5: BIT kinetic energy source term with Grace and simonnet drag correction term

```

1  /*****
2  UDF modeling BIT using source terms
3  *****/
4
5  #include "udf.h"
6  #include "math.h"
7  #include "mem.h"
8  #include "stdlib.h"
9  #include "sg_mphase.h"
10
11 DEFINE_SOURCE(eps_BIT, cell, phase_thread, dS, eqn)
12 {
13     /* variables */
14     Thread *Thread_GL; /*thread to receive all phase threads*/
15     Thread *Thread_L; /* sub-thread for liquid */
16     Thread *Thread_G; /* sub-thread for gas */
17
18     /*position variable*/
19     real tau, alpha_g, Rey, CD, Eo, eps_B, k_source, Mo, H, J, Ut;
20     real CD_cap, CD_ell, CD_sph;
21
22     /*momentum transfer*/
23     real F_drag;
24
25     /*velocity vector*/
26     real slip, slip_U, slip_V, slip_W;
27
28     /*properties*/
29     real rho_g, rho_l;
30     real mu_L, diab, surf_ten;
31
32     /*storage*/
33     real max2;
34
35     /*simonnet drag correction*/
36     real CD_factor, CD_swarm;
37     real m = 25.;

```



```

37
38  /*result*/
39  real eps_source;
40
41  /*constants*/
42  real C1          = 1.44;
43  real C3          = 1.;
44  real max_slip   = 5.;
45
46  /*gas & liquid threads*/
47  Thread_GL = THREAD_SUPER_THREAD(phase_thread); /*liquid & gas thread*/
48  Thread_L  = THREAD_SUB_THREAD(Thread_GL,0);   /* set liquid thread */
49  Thread_G  = THREAD_SUB_THREAD(Thread_GL,1);   /* set gas thread */
50
51  /*properties*/
52  rho_l     = C_R(cell,Thread_L);
53  rho_g     = C_R(cell,Thread_G);
54  mu_L      = C_MU_L(cell,Thread_L);
55  eps_B     = C_UDMI(cell,Thread_GL,23);
56  alpha_g   = C_VOF(cell,Thread_G);
57  diab      = 5.1e-3; /*m*/
58  surf_ten  = 0.072; /*N/m*/
59
60  /*check if BIT has initial value*/
61  eps_B = MAX(eps_B,1.185928e-09);
62
63  /*slip velocity*/
64  slip_U = C_U(cell,Thread_G) - C_U(cell,Thread_L);
65  slip_V = C_V(cell,Thread_G) - C_V(cell,Thread_L);
66  slip_W = C_W(cell,Thread_G) - C_W(cell,Thread_L);
67
68  /*vector velocity*/
69  slip    = pow(pow(slip_U,2.)+pow(slip_V,2.)+pow(slip_W,2.),(1./2.));
70  slip    = MIN(max_slip,slip);
71
72  /*reynolds*/
73  Rey = (rho_l * fabs(slip) * diab) / mu_L;
74
75  /*Eo*/
76  Eo = (9.81*(rho_l-rho_g)*pow(diab,2.))/surf_ten ;
77
78  /*Mo*/
79  Mo = (pow(mu_L,4.)*9.81*(rho_l-rho_g))/(pow(rho_l,2.)*pow(surf_ten,3.));
80
81  /*H*/
82  H = (4./3.)*Eo*pow(Mo,-0.149)*pow((mu_L/0.0009),-0.14);
83
84  /*J*/
85  if (H > 2. && H <= 59.2)
86  {
87      J = 0.94 * pow(H,0.757);
88  }
89  else if(H >59.2)
90  {
91      J = 3.42 * pow(H,0.441);
92  }
93
94  /*terminal rise velocity*/
95  Ut = (mu_L/(diab*rho_l))*pow(Mo,-0.149)*(J-0.857);
96
97  /*Drag coefficient grace*/
98  if (Rey <0.01)
99  {
100     CD_sph = 24/Rey;
101  }
102  else
103  {
104     CD_sph = (24/Rey)*(1+0.15*pow(Rey,0.687));

```

```

105     }
106
107     CD_ell = (4./3.) * ((9.81*diab)/pow(Ut,2))*((rho_l-rho_g)/rho_l);
108
109     CD_cap = (8./3.);
110
111     /*taking minimum and maximum*/
112
113     max2 = MIN(CD_ell,CD_cap);
114     CD = MAX(CD_sph,max2);
115
116     /* Simmonet_CD_Swarm */
117     /* Compute drag coefficient correction */
118
119     CD_factor = (1-alpha_g) * pow((pow((1-alpha_g),m)+pow((4.8*(alpha_g/(1-alpha_g))),m)),(-2/
120     m));
121
122     /* correct CD_correction */
123     if (CD_factor < 1)
124     {
125         CD_swarm = 0.8 * CD_factor*CD;
126     }
127     else
128     {
129         CD_swarm = 1*CD;
130     }
131
132     /*drag force - interfacial momentum transfer */
133     F_drag = (3./4.) * (CD_swarm/diab) * alpha_g * rho_l *(slip)*fabs(slip);
134
135     /* timescale dissipation */
136     tau = pow((pow(diab,2)/(eps_B)),(1./3.));
137
138     /*k_source*/
139     k_source = (1.-alpha_g)*C1*fabs(F_drag)*fabs(slip);
140
141     /* turbulent dissipation energy epsilon source term*/
142     eps_source = (C3/tau)*k_source;
143     eps_source = MIN(eps_source,1e8);
144
145     /*storage*/
146     C_UDMI(cell,Thread_GL,22) = k_source;
147     C_UDMI(cell,Thread_GL,23) = eps_source;
148     C_UDMI(cell,Thread_GL,24) = CD_swarm;
149     C_UDMI(cell,Thread_GL,25) = CD;
150     C_UDMI(cell,Thread_GL,26) = CD_factor;
151     C_UDMI(cell,Thread_GL,27) = F_drag;
152     C_UDMI(cell,Thread_GL,28) = tau;
153     C_UDMI(cell,Thread_GL,29) = Rey;
154
155     /*force explicit solution*/
156     dS[eqn] = 0.;
157
158     return eps_source;
159 }

```

Listing 6: BIT energy dissipation source term with Grace and simonnet drag correction term

Appendix F: Python codes

The data analysis involved the use of two Python codes: one for the setting up the gas holdup profiles and another for the liquid velocity profiles. Furthermore, a supplementary code was developed to generate a text file containing the coordinates of each measuring point, facilitating straightforward implementation.

Data processing python code

Each simulation is assigned a unique code (Appendix G) that corresponds to specific mesh and simulation settings. This code can be used as an input for the Python code to configure the corresponding profiles accordingly. The codes then read the simulation data from a text file in the same folder as the python code to set up the profiles.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Oct 28 14:59:00 2022
4
5 @author: bramb
6 """
7 import numpy as np
8 from scipy.integrate import quad
9 import matplotlib.pyplot as plt
10 import pandas as pd
11 import os
12 from scipy.stats import sem
13
14 "non dimensional plot"
15 ndim_plot = 'N'
16
17 "what to plot?"
18 plot_exp_correlation = 'Y' #Y/N
19 plot_only_mes_range = 'N' #Y/N
20 plot_sim_planes = 'N' #Y/N
21 plot_sim_avg_plane = 'Y'
22 plot_half_rR = 'N'
23 plot_t_step = 'N'
24 plot_err_bar = 'N' #N,PL,M
25 plot_t_legend = 'N'
26 plot_each_case = 'Y'
27 plot_label = 'Y' #not available when error bar enabled
28
29 case = [2,3] # [1,2,3,4] -> D = [ 0.15 , 0.4 , 1 , 3 ] m
30 run_num = [11.381,11.382]
31 run_label = ['Improved model','Improved model with Quick']
32 t_steps = [10,20,30,40,50] #sec
33
34 coll = ['green','red']
35
36 case_all = [3]
37 plot_all = 'N'
38
39 "limiting graph size" #not yet working
40 graph_lim = ['N','N'] # [ylim,xlim]
41 y_lim = 0.225,0.33
42 x_lim = 0,1
43
44 "number of points per line used"
45 ppl = 15
46 n_pl = 4
47 r_R = np.linspace(-0.875,0.875,1000) #mes = -0.875,0.875
48 run_num = np.array(run_num)
49
50 """reactor set up"""
51 R_set = np.array([[0.15,0.4,1.0,3.0], #diameter
52                 [0.15,0.16,0.16,0.2], #vel
```

```

53         [0.6,1.6,4,9]]) #H0
54
55
56 "compiling answers"
57 show = [plot_sim_planes,plot_sim_avg_plane,plot_half_rR,plot_t_step]
58
59 def func_Bd (r_R):
60     d32 = 5.8-2*r_R**2
61     return d32
62
63 def avg_gas_h (case):
64
65     set_up = case-1
66     gas_h = 0.49*(R_set [1,set_up]**(0.41))*R_set [0,set_up]**(-0.047)
67     return gas_h
68
69 def ax_gas_h (case,r_R,ndim_plot):
70     if ndim_plot == 'Y':
71         gas_h = (-1.638*((r_R**6)-1)+1.228*((r_R**4)-1)-0.939*((r_R**2)-1))
72     else:
73         gas_h = avg_gas_h(case)*(-1.638*((r_R**6)-1)+1.228*((r_R**4)-1)-0.939*((r_R**2)-1))
74     return gas_h
75
76 "plot title, axis, legend"
77 "dim"
78 def plt_gh(set_up):
79     plt.axvline(0,color='grey')
80     plt.xlabel('r/R')
81     plt.ylabel('\u03B1_g$')
82     plt.legend(loc=8)
83     return
84
85 def plt_gh_ndim(set_up):
86     plt.axvline(0,color='grey')
87     plt.xlabel('r/R')
88     plt.ylabel('\u03B1_g$/\u03B1_{avg}$')
89     plt.legend(loc=8)
90     plt.axhline(y=1,linestyle='--',color='grey')
91     return
92
93 "takes average and std over t&a data, and last rolling gh data"
94 def gem_std(data,ppl,n_pl):
95     gh = data[-1,1:ppl*n_pl+1]
96     gh = gh.reshape(-1,ppl)
97     gh_M.append(np.mean(gh,axis=0))
98
99     "determine errors of value's"
100     while plot_err_bar != 'N':
101         if plot_err_bar == 't':
102             gh_e = data[:,ppl*n_pl+1:-1] #t data
103         else:
104             gh_e = data[:,1:ppl*n_pl+1] #a data
105         gh_e = gh_e.reshape(-1,15)
106         ghe_M.append(sem(gh_e))
107         break
108
109     return()
110
111 "checks if cases requested exist"
112 def check_file_exists(filename):
113     directory = os.getcwd()
114     for path, dirs, files in os.walk(directory):
115         if filename in files:
116             return True
117         print('filename %s doesnt exist' %filename )
118     return False
119
120 "memory"

```

```

121 run_M, gh_M, ghe_M, gha_M, t_M, file_M, M_mean, M_ndim, M_ghr, cases, M_var, M_std, M_err,
    M_std_pl, M_sim_pl = [], [], [], [], [], [], [], [], [], [], [], [], [], [], []
122 rmse, rmse2, err = [], [], []
123 abb = []
124
125 "checks if data is available"
126 file_M = []
127 for i in range(len(case)):
128     for ii in range(len(run_num)):
129         file_name = 'd%s_%g_gha.txt'%(R_set[0, case[i]-1], run_num[ii])
130         exist = check_file_exists(file_name)
131         if exist:
132             cases.append(case[i]-1)
133             file_M.append([file_name, case[i]-1, run_num[ii]])
134
135 "which size data is being analyzed"
136 cases = set(cases)
137
138 "graph labels"
139 if plot_label == 'Y':
140     label = run_label
141     for i in range(int(len(file_M)/len(run_label))-1):
142         run_label = run_label + label
143
144
145 "x-value"
146 x_axis = np.zeros(pp1)
147 x = -1 #starting on the left hand side
148 for i in range(pp1):
149     x = x +(2/(pp1+1))
150     x_axis[i] = x
151
152 "putting plane data"
153 if len(file_M) == 0:
154     print('there is no data for these settings')
155
156 for i in range(len(file_M)):
157     da_set = file_M[i]
158     data = np.loadtxt(da_set[0])
159     test = data[:, 60:-2].reshape(-1, pp1)
160     ght_pl = np.loadtxt('d%s_%g_gha_pl.txt'%(R_set[0, da_set[1]], da_set[2]))
161     M_ghr.append(np.mean(data[-1, 1: pp1*n_pl+1].reshape(-1, pp1), axis=0))
162     M_mean.append(np.mean(data[:, 60:-2].reshape(-1, pp1)))
163     M_ndim.append(M_ghr[-1]/M_mean[-1])
164     M_sim_pl.append(data[-1, 1: pp1*n_pl+1].reshape(-1, pp1))
165     M_std.append(np.std(data[:, 60:-2].reshape(-1, pp1), axis=0))
166     M_std_pl.append(np.std(data[-1, 1: pp1*n_pl+1].reshape(-1, pp1), axis=0))
167     M_err.append(np.mean(abs((np.mean(data[-1, 1: pp1*n_pl+1].reshape(-1, pp1), axis=0) - ax_gas_h
        (da_set[1]+1, x_axis, 'N'))/ax_gas_h(da_set[1]+1, x_axis, 'N'))))
168     E = np.mean(data[-1, 1: pp1*n_pl+1].reshape(-1, pp1), axis=0)
169     T = ax_gas_h(da_set[1]+1, x_axis, 'N')
170     et = abs(E-T) / (abs(E)+abs(T))
171     RRMSE = np.sqrt(np.mean((E-T)**2))/sum(T)
172     abb.append(et)
173
174     RMS = np.sqrt(sum((E-T)**2)/len(E))
175     print('MREA =\t %.3f \t SMAPE =\t %.3f \t RME =\t %.3f for %s with %s'%(M_err[-1], np.
        mean(et), RMS, R_set[0, da_set[1]], run_label[i]))
176
177 "Root mean square"
178 rmse.append(np.sqrt(sum(sum(row) for row in (data[-1, 1: pp1*n_pl+1].reshape(-1, pp1) -
        ax_gas_h(da_set[1]+1, x_axis, 'N'))**2)/len(data[-1, 1: pp1*n_pl+1])))
179 rmse2.append(np.sqrt(sum((M_ghr[-1] - ax_gas_h(da_set[1]+1, x_axis, 'N'))**2)/len(M_ghr
        [-1])))
180 err.append(np.mean(M_ghr[-1] - ax_gas_h(da_set[1]+1, x_axis, 'N')))
181
182
183 "time correction"

```

```

184 data[:,-1] -= data[0,-1] #all times - t end because t doesnt start at 0
185 t_M.append(data[-1,-1])
186
187 "extracting std,avg"
188 gem_std_data = gem_std(data, ppl, n_pl)
189
190 "take valuas at other t if asked"
191 if show[3] == 'Y':
192     dt = data[2,-1]-data[1,-1]
193     plot_t_legend = 'Y'
194
195     for i in t_steps:
196         if i > data[-1,-1]:
197             print('no data at t= %.f'%(i))
198         else:
199             plot_t_legend = 'Y'
200             t_M.append(data[int(i/dt),-1])
201             gha_M.append(np.mean(data[int(i/dt),1:ppl*n_pl+1]))
202             data2 = data[:int(i/dt),:]
203             gem_std_data = gem_std(data2, ppl, n_pl)
204
205 "plot each plane data"
206 if plot_sim_planes == 'Y':
207     print('hello')
208     da_set = file_M[i]
209     for i in range(len(file_M)):
210         sim_pl = M_sim_pl[i] #get 4 planes from current measurement
211         plt.figure('All sim for D=%.2f'%R_set[0,da_set[1]])
212         for ii in range(n_pl):
213             plt.plot(x_axis, sim_pl[ii,:], label='Plane %i'%(ii+1), linestyle='--')
214         plt.plot(x_axis, M_ghr[i], label='Avarage profile', color='black')
215     plt_gh(1)
216
217
218 "now plotten"
219 "plot data per case"
220 if ndim_plot == 'N':
221     for i in range(len(file_M)):
222         da_set = file_M[i]
223         if plot_each_case == 'Y':
224             plt.figure('gh_%s' %(R_set[0,da_set[1]]))
225             plt.title('D=%sm Average axial gas holdup'%(R_set[0,da_set[1]]))
226             if plot_label == 'Y':
227                 plt.plot(x_axis, gh_M[i], label=run_label[i])
228                 plt_gh(da_set[1])
229                 print('The gas avg holdup for D=%s is %.3f for label: %s \\'%(R_set[0,da_set
[1]],M_mean[i],run_label[i]))
230                 continue
231
232             if plot_err_bar == 'M':
233                 plt.errorbar(x_axis, gh_M[i], M_std[i], capsize=5, elinewidth = 1, label='D=%.2
fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
234                 print('The error for D=%.2fm r=%.3f is %.3f'%(R_set[0,da_set[1]],da_set[2],
M_err[i]))
235                 continue
236
237             if plot_err_bar == 'PL':
238                 plt.errorbar(x_axis, gh_M[i], M_std_pl[i], capsize=5, elinewidth = 1, label='D
=%.2fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
239                 print('The error for D=%.2fm r=%.3f is %.3f'%(R_set[0,da_set[1]],da_set[2],
M_err[i]))
240                 continue
241
242             plt.plot(x_axis, gh_M[i], label='D=%.2fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
243             plt_gh(da_set[1])
244             print('The error for D=%.2fm r=%.3f is %.3f'%(R_set[0,da_set[1]],da_set[2],M_err
[i]))
245

```

```

246 plt.figure('ndim')
247 if ndim_plot== 'Y':
248     plot_each_case = 'N'
249     print('Plot all cases not available for ndim plots, plot all cases changed to No')
250     for i in range(len(file_M)):
251         da_set = file_M[i]
252         plt.figure('ndim')
253         plt.title('Normalized gas holdup'%(R_set[0,da_set[1]]))
254         if plot_label == 'Y':
255             run_label[i]='D=%sm %s'%(R_set[0,da_set[1]],run_label[i])
256             plt.plot(x_axis,M_ndim[i],label=run_label[i])
257             plt_gh(da_set[1])
258             continue
259         plt.plot(x_axis,M_ndim[i],label='D=%.2fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
260         plt_gh_ndim(da_set[1])
261
262
263 if plot_exp_correlation == 'Y':
264     for i in cases:
265         if plot_each_case == 'Y':
266             plt.figure('gh_%s' %(R_set[0,i]))
267             plt.plot(r_R,ax_gas_h(i+1,r_R,ndim_plot),color='black', label = 'Empirical
268 correlation'%(R_set[0,i]))
269             plt_gh(da_set[1])
270         if plot_each_case == 'N':
271             plt.plot(r_R,ax_gas_h(1,r_R,ndim_plot),color='black', label = 'Empirical
272 correlation'%(R_set[0,1]))
273             plt_gh(da_set[1])
274
275 "changing graph limits"
276 if graph_lim[0] == 'Y':
277     plt.ylim(y_lim)
278 if graph_lim[1] == 'Y':
279     plt.xlim(x_lim)

```

Listing 7: Python code to set up gas holdup profiles

```

1 # -*- coding: utf-8 -*-
2 """
3 Created on Fri Oct 28 14:59:00 2022
4
5 @author: bramb
6 """
7 import numpy as np
8 from scipy.integrate import quad
9 import matplotlib.pyplot as plt
10 import pandas as pd
11 import os
12 from scipy.stats import sem
13
14 "non dimensional plot"
15 ndim_plot = 'Y'
16
17 "what to plot?"
18 plot_exp_correlation = 'Y' #Y/N
19 plot_only_mes_range = 'N' #Y/N
20 plot_sim_planes = 'N' #Y/N
21 plot_sim_avg_plane = 'Y' #Y/N
22 plot_half_rR = 'N'
23 plot_t_step = 'N'
24 plot_err_bar = 'N' #a,t,N
25 plot_t_legend = 'N'
26 plot_each_case = 'Y'
27 plot_label = 'Y'
28
29 case = [3] #D = [ 0.15 , 0.4 , 1 , 3 ] m
30 run_num = [11.381,11.382]

```

```

31 run_label      = ['Improved model', 'Improved model with Quick']
32 t_steps       = [10,20,30,40,50] #sec
33
34 case_all      = [3]
35 plot_all      = 'N'
36
37 "limiting graph size" #not yet working
38 graph_lim     = ['N', 'N'] #[ylim,xlim]
39 y_lim        = 0.225,0.33
40 x_lim        = 0,1
41
42 "number of points per line used"
43 ppl          = 15
44 n_pl         = 4
45 r_R          = np.linspace(-0.875,0.875,1000) #mes = -0.875,0.875
46 run_num      = np.array(run_num)
47
48 """reactor set up"""
49 R_set = np.array([[0.15,0.4,1.0,3.0], #diameter
50                  [0.15,0.16,0.16,0.2], #vel
51                  [0.6,1.6,4,9]]) #H0
52
53 "compiling answers"
54 show = [plot_sim_planes, plot_sim_avg_plane, plot_half_rR, plot_t_step]
55
56 def cent_vel (case):
57     set_up = case-1
58     CV = 1.35*R_set[1, set_up]**(0.16)*R_set[0, set_up]**(0.4)
59     return CV
60
61 def ax_vel (case, r_R, ndim_plot):
62     a = 2.976
63     b = 0.943
64     c = 1.848
65     gas_h = (cent_vel(case)/(a-c))*(a*np.exp(-b*r_R**2)-c)
66     if ndim_plot == 'Y':
67         gas_h = (1/(a-c))*(a*np.exp(-b*r_R**2)-c)
68     return gas_h
69
70
71 "plot title, axis, legend"
72 "dim"
73 def plt_lvu(set_up):
74     plt.axhline(y=0, linestyle='--', color='grey')
75     plt.axvline(0, color='grey')
76     plt.xlabel('r/R')
77     plt.ylabel('$V_L$ (m/s)')
78     plt.legend(loc=8)
79     return
80
81 def plt_lvu_ndim(set_up):
82     plt.axhline(y=0, linestyle='--', color='grey')
83     plt.axvline(0, color='grey')
84     plt.xlabel('r/R')
85     plt.ylabel('$V_L$/$V_0$')
86     plt.legend(loc=8)
87     return
88
89
90 "takes average and std over t&a data, and last rolling gh data"
91 def gem_std(data, ppl, n_pl):
92     gh = data[-1, 1:ppl*n_pl+1]
93     gh = gh.reshape(-1, ppl)
94     gh_M.append(np.mean(gh, axis=0))
95
96     "determine errors of value's"
97     while plot_err_bar != 'N':
98         if plot_err_bar == 't':

```



```

99     gh_e = data[:,ppl*n_pl+1:-1] #t data
100     else:
101         gh_e = data[:,1:ppl*n_pl+1] #a data
102         gh_e = gh_e.reshape(-1,15)
103         ghe_M.append(sem(gh_e))
104         break
105
106     return()
107
108 "checks if cases requested exist"
109 def check_file_exists(filename):
110     directory = os.getcwd()
111     for path, dirs, files in os.walk(directory):
112         if filename in files:
113             return True
114         print('filename %s doesnt exist' %filename )
115     return False
116
117 "memory"
118 abb = []
119 run_M, gh_M, ghe_M, gha_M,t_M,file_M, M_mean,M_ndim, M_ghr,M_var,M_std,M_err,M_cent_vel,
120     M_sim_pl,cases= [], [], [], [], [], [], [], [], [], [], [], [], [], []
121
122 "checks if data is available"
123 for i in range(len(case)):
124     for ii in range(len(run_num)):
125         file_name = 'd%s_%g_ulv.txt'%(R_set[0,case[i]-1],run_num[ii])
126         exist = check_file_exists(file_name)
127         if exist:
128             cases.append(case[i]-1)
129             file_M.append([file_name,case[i]-1,run_num[ii]])
130
131 "graph labels"
132 if plot_label == 'Y':
133     label = run_label
134     for i in range(int(len(file_M)/len(run_label))-1):
135         print(i)
136         run_label = run_label + label
137
138 "x-value"
139 x_axis = np.zeros(ppl)
140 x = -1 #starting on the left hand side
141 for i in range(ppl):
142     x = x +(2/(ppl+1))
143     x_axis[i] = x
144
145
146 "which size data is being analyzed"
147 cases = set(cases)
148
149 "putting plane data"
150 if len(file_M) == 0:
151     print('there is no data for these settings')
152
153 for i in range(len(file_M)):
154     da_set = file_M[i]
155     data = np.loadtxt(da_set[0])
156     data_avg = data[-1,1:ppl*n_pl+1].reshape(-1,ppl) #roling avg data
157     data_t = data[:,60:-2].reshape(-1,ppl) #measured data
158     M_ghr.append(np.mean(data[-1,1:ppl*n_pl+1].reshape(-1,ppl),axis=0))
159     M_mean.append(np.mean(data_t))
160     M_cent_vel.append(np.mean(data_avg[-1,7]))
161     M_ndim.append(np.mean(data_avg,axis=0)/M_cent_vel[-1])
162     M_sim_pl.append(data[-1,1:ppl*n_pl+1].reshape(-1,ppl))
163     M_var.append(np.var(data[:,60:-2].reshape(-1,ppl),axis=0))
164     M_std.append(np.std(data[:,60:-2].reshape(-1,ppl),axis=0))
165     M_err.append(np.mean(abs((np.mean(data[-1,1:ppl*n_pl+1].reshape(-1,ppl),axis=0)-ax_vel(

```

```

166 da_set[1]+1, x_axis, 'N'))/ax_vel(da_set[1]+1, x_axis, 'N'))))
167
168 E = np.mean(data[-1,1:ppl*n_pl+1].reshape(-1,ppl),axis=0)
169 T = ax_vel(da_set[1]+1, x_axis, 'Y')
170 et = abs(E-T) / (abs(E)+abs(T))
171 RRMSE = np.sqrt(np.mean((E-T)**2))/sum(T)
172 abb.append(et)
173
174 RMS = np.sqrt(sum((E-T)**2)/len(E))
175 print('MREA =\t %.3f \t SMAPE =\t %.3f \t RME =\t %.3f for %s with %s' %(M_err[-1],np.
176 mean(et),RMS,R_set[0,da_set[1]],run_label[i]))
177
178 "time correction"
179 data[:,-1] -= data[0,-1] #all times - t end because t doesnt start at 0
180 t_M.append(data[-1,-1])
181
182 "extracting std,avg"
183 gem_std_data = gem_std(data, ppl, n_pl)
184
185 "take values at other t if asked"
186 if show[3] == 'Y':
187     dt = data[2,-1]-data[1,-1]
188     plot_t_legend = 'Y'
189
190     for i in t_steps:
191         if i > data[-1,-1]:
192             print('no data at t= %.f'%(i))
193         else:
194             plot_t_legend = 'Y'
195             t_M.append(data[int(i/dt),-1])
196             gha_M.append(np.mean(data[int(i/dt),1:ppl*n_pl+1]))
197             data2 = data[:int(i/dt),:]
198             gem_std_data = gem_std(data2, ppl, n_pl)
199
200 "plot each plane data"
201 if plot_sim_planes == 'Y':
202     da_set = file_M[i]
203     for i in range(len(file_M)):
204         sim_pl = M_sim_pl[i] #get 4 planes from current measurement
205         plt.figure('All sim for D=%.2f'%R_set[0,da_set[1]])
206         for ii in range(n_pl):
207             plt.plot(x_axis,sim_pl[ii,:],label='Plane %i'%(ii+1))
208             plt.plot(x_axis,M_ghr[i],label='Avarage profile')
209         plt_lvu(1)
210
211 "now plotten"
212 "plot data per case"
213 if ndim_plot == 'N':
214     for i in range(len(file_M)):
215         da_set = file_M[i]
216         if plot_each_case == 'Y':
217             plt.figure('gh_%s' %(R_set[0,da_set[1]]))
218             plt.title('D=%sm Average liquid velocity'%(R_set[0,da_set[1]]))
219             if plot_label == 'Y':
220                 plt.plot(x_axis,M_ghr[i],label=run_label[i])
221                 plt_lvu(da_set[1])
222                 print('The liquid vel MRAE for D=%s is %.3f for label: %s \\'%(R_set[0,
223 da_set[1]],M_err[i],run_label[i]))
224                 continue
225
226             if plot_err_bar == 'Y':
227                 plt.errorbar(x_axis, M_ghr[i], M_std[i], capsize=5, elinewidth = 1, label='D
228 =%.2fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
229                 print('The error for D=%.2fm r=%.3f is %.3f'%(R_set[0,da_set[1]],da_set[2],
230 M_err[i]))
231                 continue
232                 plt.plot(x_axis,M_ghr[i],label='D=%.2fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
233                 plt_lvu(da_set[1])

```

```

229     print('The error for D=%.2fm r=%.3f is %.3f'%(R_set[0,da_set[1]],da_set[2],M_err
[230     [i]))
231 if plot_exp_correlation == 'Y':
232     for i in cases:
233         if plot_each_case == 'Y':
234             plt.figure('gh_%s'%(R_set[0,i]))
235             plt.plot(r_R,ax_vel(i+1,r_R,'N'),color='black', label = 'Empirical correlation'
%(R_set[0,i]))
236             plt_lvu(da_set[1])
237
238 if ndim_plot== 'Y':
239     plot_each_case = 'N'
240     print('Plot all cases not available for ndim plots, plot all cases changed to No')
241     for i in range(len(file_M)):
242         da_set = file_M[i]
243         if plot_each_case == 'Y':
244             plt.figure('gh_%s'%(R_set[0,da_set[1]]))
245             plt.title('D=%sm n_dim liquid velocity'%(R_set[0,da_set[1]]))
246         else:
247             plt.figure('ndim')
248             plt.title('Normalized liquid velocity')
249
250         if plot_label == 'Y':
251             run_label[i]='D=%sm %s'%(R_set[0,da_set[1]],run_label[i])
252             plt.plot(x_axis,M_ndim[i],label=run_label[i])
253             plt_lvu_ndim(da_set[1])
254             continue
255
256         if plot_err_bar == 'Y':
257             plt.errorbar(x_axis, M_ndim[i], M_std[i], capsize=5, elinewidth = 1, label='D=%.2fm
r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
258             continue
259
260         plt.plot(x_axis,M_ndim[i],label='D=%.2fm r=%.3f'%(R_set[0,da_set[1]],da_set[2]))
261         plt_lvu_ndim(da_set[1])
262         print('The error for D=%.2fm r=%.3f is %.3f'%(R_set[0,da_set[1]],da_set[2],M_err[i]
)
263
264 if plot_exp_correlation == 'Y':
265     for i in cases:
266         if plot_each_case == 'Y':
267             plt.figure('gh_%s'%(R_set[0,i]))
268             plt.plot(r_R,ax_vel(i,r_R,ndim_plot),color='black', label = 'Empirical
correlation'%(R_set[0,i]))
269
270
271         if plot_each_case == 'N':
272             plt.plot(r_R,ax_vel(i,r_R,ndim_plot),color='black', label = 'Empirical
correlation'%(R_set[0,i]))
273
274 for i in range(len(file_M)):
275     da_set = file_M[i]
276     if plot_label == 'Y':
277         print('The liq vel for D=%s with %s is %.3f compared to exp value of %.3f meaning an
%.3f percent increase\\'%(R_set[0,da_set[1]],run_label[i],M_cent_vel[i],cent_vel(
da_set[1]+1),(M_cent_vel[i]/cent_vel(da_set[1]+1)))
278
279 "changing graph limits"
280 if graph_lim[0] == 'Y':
281     plt.ylim(y_lim)
282 if graph_lim[1] == 'Y':
283     plt.xlim(x_lim)

```

Listing 8: Python code to set up the liquid velocity profiles

Measuring points

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Mon Oct 17 12:17:23 2022
4
5 @author: bramb
6 """
7
8 import numpy as np
9 import matplotlib.pyplot as plt
10 from scipy import integrate
11 import scipy
12 import pandas as pd
13 import math as m
14
15 """function input"""
16 HO_D = np.array([0.5,1.25,3.75]) #np.array([0.5,1.25,2.5,3.75]) std 2.5
17 case = 3 #D = R_set([0.15,0.4,1,3])
18 n_vert_planes = 4 #how many vertical planes
19 ppl = 15
20 hor_planes = "Y" #Y/N
21 vert_planes = "N" #Y/N
22
23
24 #making of the horizontal planes data points and putting them in a group per plane
25 def data_hor (case,HO_pl,ppl,sl):
26     file.write("horizontal planes")
27     file.write('\n')
28     X0 = []
29     x = Dia = R_set[0,(case-1)]*0.5
30
31     figure, axes = plt.subplots()
32     plt.title('Horizontal plane data sampling')
33     axes.add_artist( plt.Circle((0,0),Dia,fill=False ) )
34     lim = Dia+0.1*Dia
35     plt.xlim(-lim,lim)
36     plt.ylim(-lim,lim)
37     axes.set_aspect( 1 )
38
39     for i in range (ppl):
40         dx = R_set[0,(case-1)]/(ppl+1)
41         x -= dx
42         X0.append(x)
43     X0 = np.array(X0)
44
45     #making data points and adding to plane
46     for Y in HO_D:
47         for ii in range(sl):
48             X = X0*m.cos(ii*1/sli*m.pi)
49             Z = X0*m.sin(ii*1/sli*m.pi)
50
51             plt.scatter(X,Z,color='black',s=3)
52
53             #make point and put it in a group
54             for iii in range((ppl)):
55                 file.write("surface/point-surface point-%.2f-%02d-%02d %.3f %.3f %.3f"%(Y,(
56 ii),(iii),X[iii],Y,Z[iii]))
57                 file.write('\n')
58             plt.plot()
59             return ()
60
61 #making of the horizontal plane data points and putting them in a group per plane
62 def data_vert (case,n_hor_planes,ppl):
63     HO_D = np.array([0.5,1.25,2.5,3.75])
64     file.write("vertical planes")
65     file.write('\n')
```

```

66 #filling in the x-coordinates
67 X0 = []
68 x = R_set[0,(case-1)]*0.5
69 for i in range (ppl):
70     dx = (R_set[0,(case-1)])/(ppl+1)
71     x -= dx
72     X0.append(x)
73 X0 = np.array(X0)
74 for ii in range(n_hor_planes):
75
76     X      = X0*m.cos(ii*0.25*m.pi)
77     Z      = X0*m.sin(ii*0.25*m.pi)
78
79     for Y in H0_D:
80         for iii in range((ppl)):
81             file.write("surface/point-surface point-%.2f-%02d-%02d %.2f %.2f %.2f"%(Y,(
82 ii),(iii),X[iii],Y,Z[iii]))
83             file.write('\n')
84
85     return ()
86
87 """reactor set up"""
88 R_set = np.array([[0.15,0.4,1,3],
89                 [0.15,0.16,0.16,0.2],
90                 [0.6,1.6,4,9]])
91 file = open("Data_points H0_%.1f.txt" %(R_set[0,(case-1)]) ,'+w' )
92
93 if vert_planes == 'Y':
94     vert_plane_data = data_vert(case,n_vert_planes,ppl)
95
96 if hor_planes == 'Y':
97     hor_planes_data = data_hor(case, H0_D,ppl,n_vert_planes)
98
99 file.close()

```

Listing 9: Python code provided measuring points coordinates

Appendix G: Simulation overview

This overview contains all model settings, as well as the simulation codes required to set up the gas holdup and liquid velocity profiles.

first order	second order	Solver	gas density	Drag model	K-eps model	Drag mod	Turbulent dispersion	BIT	BIT Drag	BIT correction	Tstep	First order				Second order	
												D0,15	D0,4	D1,0	D3,0	D0,4	D1,0
NITA vs Base case																	
11	37	-	cst	Tomi	Std	Disp					0,001	x	x				
11	38	Nita	cst	Tomi	Std	Disp					0,001	x	x			t=95 t=65	
With simmonnet or without																	
11	38	Nita	cst	Tomi	Std	Disp					0,001	x	x			t=95 t=65	
11	68	Nita	cst	Grace	Std	Disp						x	x				
11	45	Nita	cst	Tomi	Std	Disp					0,0001	x	x				
11	45	Nita	cst	Tomi	RNG	Disp					0,001	x	x				
11	45	Nita	cst	Tomi	RNG	Disp					0,001	x	x				
11	46	Nita	cst	Grace	Std	Disp	n=1				0,001	x	x				
11	49	Nita	cst	Grace	Std	Disp	n=4				0,001	x	x			x	
Addition of BIT with and without Burns																	
11	39	NITA	cst	Tomi	Std	Disp	0,12	k-eps V25	Tomi	0,12	0,001	x	x			x	
11	40	NITA	cst	Tomi	Std	Disp	0,12	Burns	Tomi	0,12	0,001	x	x			x	
11	70	Nita	cst	Grace	Std	Disp	n=4		Grace	n=4	0,001	t=11	t=30			t=56 s	
11	71	Nita	cst	Grace	Std	Disp	n=4	Burns	grace	n=4		t=11	x				
11		Nita	cst	Grace	Std	Disp	n=4		Grace	n=4		x	x				
Drag model toyama vs grace																	
11	40	NITA	cst	Tomi	Std	Disp	0,12	Burns	Tomi	0,12	0,001	x	x			x	
11	41	NITA	cst	grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	x	x			x	
Drag modification																	
11	41	NITA	cst	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	x	x			x	
11	42	NITA	cst	Grace	Std	Disp	Sim	Burns	Grace	Sim	0,001	x	x				
11	43	NITA	cst	Grace	Std	Disp	0,12	Burns	Grace	Sim	0,001	x	x				
11	44	NITA	cst	Grace	Std	Disp	n=1	Burns	Grace	Sim	0,001	x	x				
Turbulence modeling with simonnet																	
11	43	NITA	cst	Grace	Std	Disp	0,12	Burns	Grace	Sim		x	x				
11	47	NITA	cst	Grace	RNG	Disp	0,12	Burns	Grace	Sim		x	x				
Turbulence modeling With constant drag mod																	
11	41	NITA	cst	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	x	x			x	
11	48	NITA	cst	Grace	RNG	Disp	0,12	Burns	Grace	0,12	0,001	x	x			x	
Mesh independancy																	
11	41	Nita		Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	0,6s	x	x	resi		
16	41	Nita		Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	t=43	x				
16		NITA	IG	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001					t=47	
16	65	NITA	IG	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001					t=29,5	
Different locations																	
11	413	Nita		Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	x	x				
non constant density and mesh dependency																	
11	72	Nita	IG	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	x	x			resi	
16	72	Nita	IG	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001	x	x			x	
16		Nita	IG	Grace	Std	Disp	0,12	Burns	Grace	0,12	0,001					x	